Journal of Machine Learning Research 7 (2006) 31–54 Submitted 3/05; Revised 11/05; Published 1/06

Incremental Algorithms for Hierarchical Classification

Nicolò Cesa-Bianchi CESA-BIANCHI@DSI.UNIMI.IT
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
via Comelico 39
20135 Milano, Italy

Claudio Gentile CLAUDIO.GENTILE@UNINSUBRIA.IT
Dipartimento di Informatica e Comunicazione
Università dell'Insubria
via Mazzini 5
21100 Varese, Italy

Luca Zaniboni ZANIBONI@DTI.UNIMI.IT
Dipartimento di Tecnologie dell'Informazione
Università degli Studi di Milano
via Bramante 65
26013 Crema (CR), Italy

Editor: Michael Collins

Abstract

We study the problem of classifying data in a given taxonomy when classifications associated with multiple and/or partial paths are allowed. We introduce a new algorithm that incrementally learns a linear-threshold classifier for each node of the taxonomy. A hierarchical classification is obtained by evaluating the trained node classifiers in a top-down fashion. To evaluate classifiers in our multipath framework, we define a new hierarchical loss function, the H-loss, capturing the intuition that whenever a classification mistake is made on a node of the taxonomy, then no loss should be charged for any additional mistake occurring in the subtree of that node.

Making no assumptions on the mechanism generating the data instances, and assuming a linear noise model for the labels, we bound the H-loss of our on-line algorithm in terms of the H-loss of a reference classifier knowing the true parameters of the label-generating process. We show that, in expectation, the excess cumulative H-loss grows at most logarithmically in the length of the data sequence. Furthermore, our analysis reveals the precise dependence of the rate of convergence on the eigenstructure of the data each node observes.

Our theoretical results are complemented by a number of experiments on texual corpora. In these experiments we show that, after only one epoch of training, our algorithm performs much better than Perceptron-based hierarchical classifiers, and reasonably close to a hierarchical support vector machine.

Keywords: incremental algorithms, online learning, hierarchical classification, second order perceptron, support vector machines, regret bound, loss function

1. Introduction

In this paper, we investigate the problem of classifying data based on the knowledge that the graph of dependencies between the classes is a tree forest. The trees in this forest are collectively interpreted

©2006 Nicolò Cesa-Bianchi, Claudio Gentile and Luca Zaniboni.

as a taxonomy. That is, we assume that every data instance is labelled with a (possibly empty) set of class nodes and, whenever an instance is labelled with a certain node *i*, then it is also labelled with all the nodes on the path from the root of the tree where *i* occurs down to node *i*. A distinctive feature of our framework is that we also allow multiple-path labellings (instances can be labelled with nodes belonging to more than one path in the forest), and partial-path labellings (instances can be labelled with nodes belonging to a path that does not end on a leaf).

We introduce a new algorithm that incrementally learns a linear-threshold classifier for each node of the taxonomy. A hierarchical classification is then obtained by evaluating the node classifiers in a top-down fashion, so that the final labelling is consistent with the taxonomy.

The problem of hierarchical classification, especially of textual information, has been extensively investigated in past years (see, e.g., Dumais and Chen, 2000; Dekel et al., 2004, 2005; Granitzer, 2003; Hofmann et al., 2003; Koller and Sahami, 1997; McCallum et al., 1998; Mladenic, 1998; Ruiz and Srinivasan, 2002; Sun and Lim, 2001, and references therein). The on-line approach to hierarchical classification, which we analyze here, seems well suited when dealing with scenarios in which new data are produced frequently and in large amounts (e.g., data produced by newsfeeds—considered in this paper, or the speech data considered in Dekel et al., 2005).

An important ingredient in a hierarchical classification problem is the loss function used to evaluate the classifier's performance. In pattern classification the zero-one loss is traditionally used. In a hierarchical setting this loss would simply count one mistake each time, on a given data instance, the set of class labels output by the hierarchical classifier is not perfectly identical to the set of true labels associated to that instance. Loss functions able to reflect the taxonomy structure have been proposed in the past (e.g., Dekel et al., 2004; Hofmann et al., 2003; Sun and Lim, 2001), but none of these losses works well in our framework where multiple and partial paths are allowed. In this paper we define a new loss function, the H-loss (hierarchical loss), whose simple definition captures the following intuition: "if a mistake is made at node *i* of the taxonomy, then further mistakes made in the subtree rooted at *i* are unimportant". In other words, we do not require the algorithm be able to make fine-grained distinctions on tasks where it is unable to make coarse-grained ones. For example, if an algorithm failed to label a document with the class SPORTS, then the algorithm should not be charged more loss because it also failed to label the same document with the subclass SOCCER and the sub-subclass CHAMPIONS LEAGUE.

We bound the theoretical performance of our algorithm using the H-loss. In our analysis, we make no assumptions on the mechanism generating the data instances; that is, we bound the H-loss of the algorithm for any arbitrary sequence of data instances. The hierarchical labellings associated to the instances, instead, are assumed to be independently generated according to a parametric stochastic process defined on the taxonomy.

Following a standard approach in the analysis of on-line algorithms, we measure the predictive performance using the cumulative regret, a quantity measuring the difference between the cumulative H-loss of the classifiers incrementally generated by the on-line algorithm during its run and the cumulative H-loss of a fixed reference classifier. Our main theoretical result is a bound on the regret of our hierarchical learning algorithm with respect to a reference hierarchical classifier based on the true parameters of the label-generating process. More specifically, we bound the contribution to the cumulative regret of each node classifier in terms of quantities related to the position of the node in the taxonomy and the data process parameters. This interaction between node position and data process parameters captures the hierarchical nature of the classification problem since the contribution of each node to the overall cumulative regret decreases as we proceed downward from a root in the forest. In general, the overall cumulative regret is seen to grow at most logarithmically in the length T of the data sequence.

From the theoretical point of view, the novelty of this line of research is twofold:

1. The use of hierarchically trained linear-threshold classifiers is common to several of the previous approaches to hierarchical classification (e.g., Dumais and Chen, 2000; Dekel et al., 2004, 2005; Granitzer, 2003; Hofmann et al., 2003; Koller and Sahami, 1997; McCallum et al., 1998; Mladenic, 1998; Ruiz and Srinivasan, 2002; Sun and Lim, 2001). However, to our knowledge, this research is the first one to provide a rigorous performance analysis of hierarchical classification algorithms in the presence of multiple and partial path classifications.

2. The core of our analysis is a local cumulative regret bound showing that the instantaneous regret of each node classifier vanishes at a rate $1/T$. The precise dependence of the rate of convergence on the eigenstructure of the data each node observes is a major contribution of this paper. This turns out to be similar in spirit to early (and classical) work in least-squares linear regression (e.g., Lai et al., 1979; Lai and Wei, 1982). But unlike these previous investigations, our analysis is not asymptotic in nature and studies a specific classification setting, instead of a regression one.

To support our theoretical findings we also describe some experiments concerning a more practical variant of the algorithm we actually analyze. These experiments use large corpora of textual data on which we test different batch and incremental classifiers. The experiments show that our on-line algorithm performs significantly better than Perceptron-based hierarchical classifiers. Furthermore, after only one epoch of training, our algorithm achieves a performance close to that of a hierarchical support vector machine, the popular batch learning algorithm for which, to the best of our knowledge, no theoretical performance bounds are known in hierarchical classification frameworks.

The paper is organized as follows. Section 2 defines the notation used throughout the paper. In Section 3 we introduce the H-loss function. Our hierarchical algorithm is described in Section 4. In Section 5 and 6 we define the data model, the learning model, and our theoretical performance measure: the cumulative regret. The analysis of our algorithm is carried out in Section 7, while in Section 8 we report on the experiments. Finally, in Section 9 we summarize our results and mention a few open questions.

2. Notation

We assume data elements are encoded as unit-norm vectors $x \in \mathbb{R}^d$, which we call *instances*. A *multilabel* for an instance x is any subset of the set $\{1,\dots,N\}$ of all labels, including the empty set. We represent the multilabel of x with a vector $v = (v_1,\dots,v_N) \in \{0,1\}^N$, where $i \in \{1,\dots,N\}$ belongs to the multilabel of x if and only if $v_i = 1$.

A *taxonomy* G is a forest whose trees are defined over the set of labels. A multilabel $v \in \{0,1\}^N$ is said to *respect* a taxonomy G if and only if v is the union of one or more paths in G, where each path starts from a root but need not terminate on a leaf, see Figure 1. We assume the data-generating mechanism produces examples (x,v) such that v respects some fixed underlying taxonomy G with N nodes (see Section 5). The set of roots in G is denoted by $\text{ROOT}(G)$. We use $\text{PAR}(i)$ to denote the unique parent of node i, $\text{ANC}(i)$ to denote the set of ancestors of i, $\text{SUB}(i)$ to denote the set of nodes in the subtree rooted at i (including i), and $\text{CHILD}(i)$ to denote the set of children of node i.

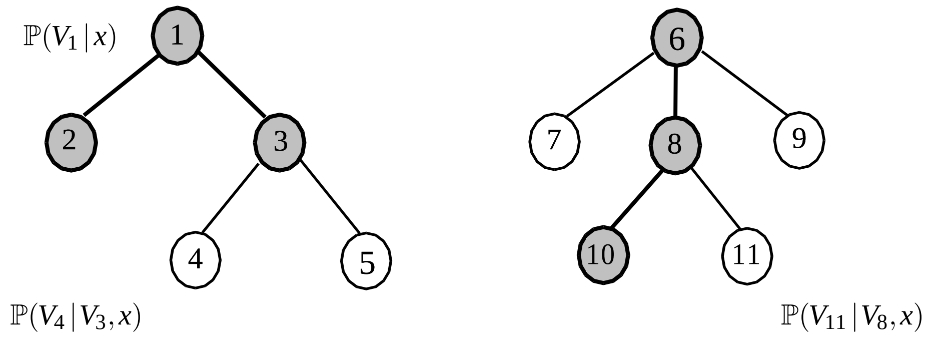

$\mathbb{P}(V_1 \,|\, x)$
1
2
3
4
5
6
7
8
9
10
11
$\mathbb{P}(V_4 \,|\, V_3, x)$
$\mathbb{P}(V_{11} \,|\, V_8, x)$

Figure 1: A forest made of two disjoint trees. The nodes are tagged with the name of the labels, so that in this case $N = 11$. According to our definition, the multilabel $v = (1,1,1,0,0,1,0,1,0,1,0)$ respects this taxonomy (since it is the union of paths $1 \rightarrow 2$, $1 \rightarrow 3$ and $6 \rightarrow 8 \rightarrow 10$), while the multilabel $v = (1,1,0,1,0,0,0,0,0,0,0)$ does not, since $1 \rightarrow 2 \rightarrow 4$ is not a path in the forest. Associated with each node i is a $\{0,1\}$-valued random variable V_i distributed according to a conditional probability function $\mathbb{P}(V_i \mid V_{\text{PAR}(i)}, x)$ —see Section 5.

We denote by $\{\phi\}$ the Bernoulli random variable which is 1 if and only if predicate ϕ is true. In our analysis, we repeatedly use simple facts such as $\{\phi \vee \psi\} = \{\phi\} + \{\psi \wedge \neg\phi\} \leq \{\phi\} + \{\psi\}$ and $\{\phi\} = \{\phi \wedge \psi\} + \{\phi \wedge \neg\psi\} \leq \{\phi \wedge \psi\} + \{\neg\psi\}$, where ψ is another predicate.

3. The H-Loss

Two very simple loss functions, measuring the discrepancy between the prediction multilabel $\widehat{y} = (\widehat{y}_1, \ldots, \widehat{y}_N)$ and the true multilabel $v = (v_1, \ldots, v_N)$, are the zero-one loss $\ell_{0/1}(\widehat{y}, v) = \{\exists i : \widehat{y}_i \neq v_i\}$ and the symmetric difference loss $\ell_\Delta(\widehat{y}, v) = \{\widehat{y}_1 \neq v_1\} + \ldots + \{\widehat{y}_N \neq v_N\}$. Note that the definition of these losses is based on the set $\{1, \ldots, N\}$ of labels without any additional structure. A loss function that takes into account a taxonomical structure defined over the set of labels is

$$\ell_H(\widehat{y}, v) = \sum_{i=1}^{N} \{\widehat{y}_i \neq v_i \wedge \widehat{y}_j = v_j,\ j \in \text{ANC}(i)\}\ .$$

This loss, which we call H-loss (hierarchical loss), can also be defined as follows: all paths in G from a root down to a leaf are examined and, whenever a node i is encountered such that $\widehat{y}_i \neq v_i$, then 1 is added to the loss, while all the loss contributions in the subtree rooted at i are discarded. Note that, with this definition, $\ell_{0/1} \leq \ell_H \leq \ell_\Delta$. A graphical representation of the H-loss and related concepts is given in Figure 2.

In the next lemma we show an important (and intuitive) property of the H-loss: when the multilabel v to be predicted respects a taxonomy G then there is no loss of generality in restricting to predictions which respect G. Formally, given a multilabel $\widehat{y} \in \{0,1\}^N$, we define the *$G$-truncation* of $\widehat{y}$ as the multilabel $\widehat{y}' = (\widehat{y}'_1, \ldots, \widehat{y}'_N) \in \{0,1\}^N$ where, for each $i = 1, \ldots, N$, $\widehat{y}'_i = 1$ if and only if $\widehat{y}_i = 1$ and $\widehat{y}_j = 1$ for all $j \in \text{ANC}(i)$. Note that the G-truncation of any multilabel always respects G. The next lemma states that if v respects G, then $\ell_H(\widehat{y}, v)$ cannot be smaller than $\ell_H(\widehat{y}', v)$.

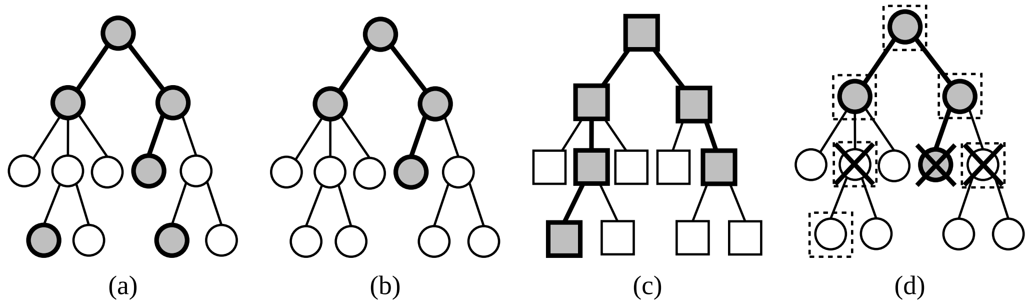


Figure 2: A one-tree forest (repeated four times). Each node corresponds to a class in the taxonomy G, hence in this case $N = 12$. Gray nodes are included in the multilabel under consideration, white nodes are not. (a) A generic multilabel which *does not* respect G; (b) its G-truncation. (c) A second multilabel that respects G. (d) Superposition of multilabel (b) on multilabel (c): Only the checked nodes contribute to the H-loss between (b) and (c). Hence the H-loss between multilabel (b) and multilabel (c) is 3. Here the zero-one loss between (b) and (c) is 1, while the symmetric difference loss equals 4.

Lemma 1 *Let G be a taxonomy, $v, \widehat{y} \in \{0,1\}^N$ be two multilabels such that v respects G, and $\widehat{y}'$ be the G-truncation of $\widehat{y}$. Then*

$$\ell_H(\widehat{y}', v) \le \ell_H(\widehat{y}, v) \ .$$

Proof. Since $\ell_H(\widehat{y}', v) = \sum_{i=1}^N \{\widehat{y}'_i \neq v_i \wedge \widehat{y}'_j = v_j,\ j \in \mathrm{ANC}(i)\}$ and $\ell_H(\widehat{y}, v) = \sum_{i=1}^N \{\widehat{y}_i \neq v_i \wedge \widehat{y}_j = v_j,\ j \in \mathrm{ANC}(i)\}$, it suffices to show that, for each $i = 1, \ldots, N$, $\widehat{y}'_i \neq v_i$ and $\widehat{y}'_j = v_j$ for all $j \in \mathrm{ANC}(i)$ implies $\widehat{y}_i \neq v_i$ and $\widehat{y}_j = v_j$ for all $j \in \mathrm{ANC}(i)$.

Pick some i and suppose $\widehat{y}'_i \neq v_i$ and $\widehat{y}'_j = v_j$ for all $j \in \mathrm{ANC}(i)$. Now suppose $\widehat{y}'_j = 0$ (and thus $v_j = 0$) for some $j \in \mathrm{ANC}(i)$. Then $v_i = 0$ since v respects G. But this implies $\widehat{y}'_i = 1$, contradicting the fact that the G-truncation $\widehat{y}'$ respects G. Therefore, it must be the case that $\widehat{y}'_j = v_j = 1$ for all $j \in \mathrm{ANC}(i)$. Hence the G-truncation of $\widehat{y}$ left each node $j \in \mathrm{ANC}(i)$ unchanged, implying $\widehat{y}_j = v_j$ for all $j \in \mathrm{ANC}(i)$. But, since the G-truncation of $\widehat{y}$ does not change the value of a node i whose ancestors j are such that $\widehat{y}_j = 1$, this also implies $\widehat{y}_i = \widehat{y}'_i$. Therefore $\widehat{y}_i \neq v_i$ and the proof is concluded. $\square$

4. A New Hierarchical Learning Algorithm

In this section we describe our on-line algorithm for hierarchical classification. Its theoretical performance is analyzed in Section 7.

The on-line learning model we consider is the following. In the generic time step $t = 1, 2, \ldots$ instance x_t is revealed to the algorithm which outputs the prediction $\widehat{y}_t = (\hat{y}_{1,t}, \ldots, \hat{y}_{N,t}) \in \{0,1\}^N$. This is viewed as a guess for the multilabel $v_t = (v_{1,t}, v_{2,t}, \ldots, v_{N,t})$ associated with the current instance x_t. After each prediction, the algorithm observes the true multilabel v_t and adjusts its parameters for the next prediction.

Our algorithm computes $\hat{y}_{1,t}, \ldots, \hat{y}_{N,t}$ using N linear-threshold classifiers, one for each node in the taxonomy. These node classifiers are evaluated, starting from each root, in the following top-down fashion: the root is labelled by evaluating its node classifier; if a node has been labelled 1, then each child is labelled by evaluating its node classifier. On the other hand, if a node is labelled

```
Algorithm H-RLS.
Initialization: Weight vectors $w_{i,1} = (0,\dots,0)$, $i = 1,\dots,N$.
For $t = 1,2,\dots$ do
    1. Observe instance $x_t \in \{x \in \mathbb{R}^d : ||x|| = 1\}$;
    2. For each $i = 1,\dots,N$ compute predictions $\hat{y}_{i,t} \in \{0,1\}$ as follows:
       $\hat{y}_{i,t} = \begin{cases} \{w_{i,t}^\top x_t \ge 0\} & \text{if } i \text{ is a root node,} \\ \{w_{i,t}^\top x_t \ge 0\} & \text{if } i \text{ is not a root node and } \hat{y}_{j,t} = 1 \text{ for } j = \textsc{par}(i), \\ 0 & \text{if } i \text{ is not a root node and } \hat{y}_{j,t} = 0 \text{ for } j = \textsc{par}(i), \end{cases}$
       where
       $w_{i,t} = (I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^\top + x_t x_t^\top)^{-1} \times S_{i,Q(i,t-1)} (v_{i,i_1}, v_{i,i_2}, \dots, v_{i,i_{Q(i,t-1)}})^\top$
       $S_{i,Q(i,t-1)} = [x_{i_1}\, x_{i_2} \dots x_{i_{Q(i,t-1)}}] \qquad i = 1,\dots,N.$
    3. Observe multilabel $v_t$ and update weights.
```

Figure 3: The hierarchical learning algorithm H-RLS.

0 then *all* of its descendants are labelled 0. Note that this evaluation scheme can only generate multilabels that respect the underlying taxonomy.

Let $w_1,\dots,w_N$ be the weight vectors defining the linear-threshold classifiers used by the algorithm. A feature of the learning process, which is also important for its theoretical analysis, is that the classifier at node i is only trained on the examples that are positive for its parent node. In other words, w_i is considered for update only on those instances x_t such that $v_{\textsc{par}(i),t} = 1$.

Let $Q(i,t)$ denote the number of times the *parent* of node i observes a positive label up to time t, i.e., $Q(i,t) = |\{1 \le s \le t : v_{\textsc{par}(i),s} = 1\}|$. The weight vector $w_{i,t}$ stored at time t in node i is a (conditional) regularized least squares estimator given by

$$w_{i,t} = \left(I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^\top + x_t x_t^\top\right)^{-1} S_{i,Q(i,t-1)} (v_{i,i_1},\dots,v_{i,i_{Q(i,t-1)}})^\top, \qquad (1)$$

where I is the $d \times d$ identity matrix, $S_{i,Q(i,t-1)}$ is the $d \times Q(i,t-1)$ matrix whose columns are the instances $x_{i_1},\dots,x_{i_{Q(i,t-1)}}$, and $(v_{i,i_1},\dots,v_{i,i_{Q(i,t-1)}})^\top$ is the $Q(i,t-1)$-dimensional (column) vector of the corresponding labels observed by node i.

The estimator in (1) is a slight variant of the regularized least squares estimator for classification (Cesa-Bianchi et al., 2002; Rifkin et al., 2003) where we include the current instance x_t in the computation of $w_{i,t}$ (see, e.g., Azoury and Warmuth, 2001; Vovk, 2001, for analyses of similar algorithms in different contexts). Efficient incremental computations of the inverse matrix and dual variable formulations of the algorithm are extensively discussed by Cesa-Bianchi et al. (2002) and Rifkin et al. (2003).

The pseudocode of our algorithm, which we call H-RLS (Hierarchical Regularized Least Squares) is given in Figure 3.

5. A Stochastic Model for Generating Labels

While no assumptions are made on the mechanism generating the sequence $x_1, x_2, \dots$ of instances, we base our analysis on the following stochastic model for generating the multilabel associated to an instance x_t.

A probability distribution f_G over the set of multilabels is associated to a taxonomy G as follows. Each node i of G is tagged with a $\{0,1\}$-valued random variable V_i distributed according to a conditional probability function $\mathbb{P}(V_i \mid V_{\mathrm{PAR}(i)}, x)$. To model the dependency between the labels of nodes i and $j = \mathrm{PAR}(i)$ we assume

$$\mathbb{P}\left(V_i = 1 \mid V_j = 0, x\right) = 0 \tag{2}$$

for all nonroot nodes i and all instances x. For example, in the taxonomy of Figure 1 we have $\mathbb{P}\left(V_4 = 1 \mid V_3 = 0, x\right) = 0$ for all $x \in \mathbb{R}^d$. The quantity

$$f_G(v \mid x) = \prod_{i=1}^{N} \mathbb{P}\left(V_i = v_i \mid V_j = v_j,\, j = \mathrm{PAR}(i), x\right)$$

thus defines a joint probability distribution on $V_1, \dots, V_N$ conditioned on x being the current instance. This joint distribution puts zero probability on all multilabels $v \in \{0,1\}^N$ which do not respect G.

Through f_G we specify an i.i.d. process $\{\boldsymbol{V}_1, \boldsymbol{V}_2, \dots\}$ as follows. We assume that an arbitrary and unknown sequence of instance vectors $x_1, x_2, \dots$ is fixed in advance, where $\|x_t\| = 1$ for all t. The multilabel $\boldsymbol{V}_t$ is distributed according to the joint distribution $f_G(\cdot \mid x_t)$. We call each pair (x_t, v_t), where v_t is a realization of $\boldsymbol{V}_t$, an *example*.

Let us now introduce a parametric model for f_G. With each node i in the taxonomy, we associate a unit-norm weight vector $u_i \in \mathbb{R}^d$. Then, we define the conditional probabilities for a nonroot node i with parent j as follows:

$$\mathbb{P}\left(V_i = 1 \mid V_j = 1, x\right) = \frac{1 + u_i^\top x}{2}\,. \tag{3}$$

If i is a root node, the above simplifies to

$$\mathbb{P}\left(V_i = 1 \mid x\right) = \frac{1 + u_i^\top x}{2}\,.$$

Our choice of a linear model for Bernoulli random variables, as opposed to a more standard log-linear model, is mainly motivated by our intention of proving regret bounds with no assumptions on the way the sequence of instances is generated. Indeed, we are not aware of any analysis of logistic regression holding in a similar classification setup.

Note also that, in this model, the labels of the children of any given node are independent random variables. This is motivated by the fact that, unlike previous investigations, we are expliciteply modelling labellings involving multiple paths. A more sophisticated analysis could introduce arbitrary negative correlations among the labels of the children nodes. In this paper, however, we do not follow this route.

6. Regret and the Reference Classifier

Assuming the stochastic model described in Section 5, we compare the performance of our algorithm to the performance of the fixed hierarchical classifier built on the true parameters $u_1,\dots,u_N$ governing the label-generating process. This reference hierarchical classifier has the same form as the classifiers generated by H-RLS. More precisely, let the multilabel $y=(y_1,\dots,y_N)$ for an instance x be computed as follows:

$$y_i = \begin{cases} \{u_i^\top x \ge 0\} & \text{if } i \text{ is a root node,} \\ \{u_i^\top x \ge 0\} & \text{if } i \text{ is not a root and } y_j = 1 \text{ for } j = \text{PAR}(i), \\ 0 & \text{if } i \text{ is not a root and } y_j = 0 \text{ for } j = \text{PAR}(i). \end{cases} \tag{4}$$

To evaluate our algorithm against the reference hierarchical classifier defined in (4), we use the cumulative regret. Given any loss function ℓ (such as one of the three defined in Section 3), we define the (instantaneous) *regret* of a classifier assigning label $\widehat{y}_t$ to instance x_t as

$$\mathbb{E}\,\ell(\widehat{y}_t, V_t) - \mathbb{E}\,\ell(y_t, V_t)\ ,$$

where y_t is the multilabel assigned by classifier (4), and the expectation is with respect the random draw of V_t (as specified in Section 5). We measure the performance of H-RLS through its cumulative regret on a sequence of T examples:

$$\sum_{t=1}^{T} \bigl(\mathbb{E}\,\ell(\widehat{y}_t, V_t) - \mathbb{E}\,\ell(y_t, V_t)\bigr)\ . \tag{5}$$

The regret bound we prove in Section 7 holds when $\ell = \ell_H$, and is shown to depend on the interaction between the spectral structure of the data generating process and the structure of the taxonomy on which the process is applied.

7. Analysis

We now prove a bound on the cumulative regret of H-RLS with respect to the H-loss function ℓ_H. Our analysis hinges on proving that for any node i, the estimated margin $w_{i,t}^\top x_t$ is an asymptotically unbiased estimator of the true margin $u_i^\top x_t$, and then on using known large deviation arguments to obtain the stated bound. For this purpose, we bound the variance of the margin estimator at each node and prove a bound on the rate at which the bias vanishes.

Theorem 2 *Consider a taxonomy G with N nodes. Pick any set of model parameters $u_1,\dots,u_N \in \mathbb{R}^d$ such that $\|u_i\| = 1$ for $i = 1,\dots,N$, and pick any sequence of instance vectors $x_1, x_2, \dots \in \mathbb{R}^d$ such that $\|x_t\| = 1$ for all t. Then the cumulative regret of the* H-RLS *algorithm (described in Figure 3) satisfies, for each $T \ge 1$,*

$$\sum_{t=1}^{T} \bigl(\mathbb{E}\,\ell_H(\widehat{y}_t, V_t) - \mathbb{E}\,\ell_H(y_t, V_t)\bigr) \le 16(1+1/e) \sum_{i=1}^{N} \frac{C_i}{\Delta_i^2}\, \mathbb{E}\left[\sum_{j=1}^{d} \log(1+\lambda_{i,j})\right]\ ,$$

where

$$\Delta_{i,t} = u_i^\top x_t, \qquad \Delta_i^2 = \min_{t=1,\dots,T} \Delta_{i,t}^2, \qquad C_i = |\text{SUB}(i)|,$$

$\lambda_{i,1},\dots,\lambda_{i,d}$ are the eigenvalues of matrix $S_{i,Q(i,T)} S_{i,Q(i,T)}^\top$, and e is the base of natural logarithms.

Before delving into the proof, it is worth making a few comments.

Remark 3 Since H-RLS can be cast in dual variables, we can run it in any reproducing kernel Hilbert space (e.g., Schölkopf and Smola, 2002). The regret bound contained in Theorem 2 remains true once we observe that the nonzero eigenvalues of $S_{i,Q(i,T)}S_{i,Q(i,T)}^\top$ coincide with the nonzero eigenvalues of the Gram matrix $S_{i,Q(i,T)}^\top S_{i,Q(i,T)}$, and we replace the sum over all input dimensions d with the sum over the (at most T) nonzero eigenvalues of $S_{i,Q(i,T)}^\top S_{i,Q(i,T)}$. We refer the reader to the work by Cesa-Bianchi et al. (2002) for additional details.

Remark 4 It is important to emphasize the interplay between the taxonomy structure and the process generating the examples, as expressed by the above regret bound. Recall that we denote by $\lambda_{i,1},\ldots,\lambda_{i,d}$ the eigenvalues of matrix $S_{i,Q(i,T)}S_{i,Q(i,T)}^\top$. From the previous remark we have $\sum_{j=1}^d \lambda_{i,j} = \mathrm{trace}\big(S_{i,Q(i,T)}^\top S_{i,Q(i,T)}\big) = Q(i,T)$ since $\|x_t\| = 1\ \forall t$, and

$$\sum_{j=1}^d \log(1+\lambda_{i,j}) \le \max\left\{\sum_{j=1}^d \log(1+\mu_j)\ :\ \sum_{j=1}^d \mu_j = Q(i,T)\right\} = d\log\left(1+\frac{Q(i,T)}{d}\right).$$

Moreover, $Q(i,T)$ is the sum of T Bernoulli random variables, where the t-th variable takes value 1 when the parent of the i-th node in the taxonomy observes label $V_{\mathrm{PAR}(i),t} = 1$ at time t. The probability of this event clearly equals

$$\prod_{j\in\mathrm{ANC}(i)}\left(\frac{1+\Delta_{j,t}}{2}\right).$$

Thus

$$\begin{aligned}
\mathbb{E}\left[\sum_{j=1}^d \log(1+\lambda_{i,j})\right] &\le d\,\mathbb{E}\left[\log\left(1+\frac{Q(i,T)}{d}\right)\right] && (6)\\
&\le d\log\left(1+\frac{\mathbb{E}\,Q(i,T)}{d}\right)\\
&\quad\text{(from Jensen's inequality)}\\
&= d\log\left(1+\frac{\sum_{t=1}^T \prod_{j\in\mathrm{ANC}(i)}\left(\frac{1+\Delta_{j,t}}{2}\right)}{d}\right). && (7)
\end{aligned}$$

Bound (6) is obviously a $\log T$ cumulative regret bound, since $Q(i,T) \le T$ anyway. It is important, however, to see how the regret bound depends on the taxonomy structure. Let us focus on (7). If i is a root node then $\mathbb{E}\,Q(i,T) = Q(i,T) = T$ (since a root node observes all labels). As we descend along a path, $\mathbb{E}\,Q(i,T)$ tends to decrease with a rate depending on the margins achieved by the ancestors of node i. Bound (7) thus makes explicit the contribution of node i to the overall regret. If i is a root node, then its contribution to the overall regret is roughly $\log T$. On the other hand, the deeper is node i within the taxonomy the smaller is the contribution of node i to the overall regret. A very deep leaf node observes a possibly small subset of the instances, but it is also required to produce only a small subset of linear-threshold predictions, i.e., the associated weight vector $w_{i,t}$ might be an unreliable estimator, but is also used less often. Therefore, the contribution of leaf node i is smaller than $\log T$ because the hierarchical nature of the problem (as expressed by the H-loss) lowers the relative importance of the accuracy of estimator $w_{i,t}$ when computing the overall regret.

Remark 5 Nothing prevents us from generalizing the H-loss by associating fixed cost coefficients to each taxonomy node:

$$\ell_H(\widehat{y}, v) = \sum_{i=1}^{N} c_i \{\widehat{y}_i \neq v_i \wedge \widehat{y}_j = v_j,\ j \in \text{ANC}(i)\} ,$$

where the cost coefficients c_i are positive real numbers. It is straightforward to see that with this definition of H-loss, the statement of Theorem 2 still holds, once we generalize the regret factors C_i as $C_i = \sum_{k \in \text{SUB}(i)} c_k$. Note that this would involve changes neither in our learning algorithm nor in our reference predictor. In fact, we are measuring regret against a reference predictor that is not Bayes optimal for the data model at hand. This is not immediate to see when the cost coefficients c_i defining the H-loss are all set to 1 but, as we mentioned, it is generally evinced by the fact that both the reference predictor (4) and our learning algorithm do not depend on the c_i.

Remark 6 From the proof of Theorem 2 below, the reader can see that there are several ways one can improve the bounds. In fact, we made no special effort to minimize the main constant $16(1+1/e)$ and, in general, we disregarded quite a lot of constant factors throughout. Moreover, though we decided to cast the bounds in terms of the worst-case margin $\Delta_i^2 = \min_{t=1,\dots,T} \Delta_{i,t}^2$, it is straighforward to modify the proof to obtain a bound depending on some sort of average squared margin. Since this sharper bound would hide the clean dependence on the eigenstructure of the data, we decided not to pursue this optimization any further.

We are now ready to prove Theorem 2.

Proof of Theorem 2. We fix a node i and upper bound its contribution to the total instantaneous regret. Since for any four predicates ϕ, ψ, χ, ζ we have $\{\phi \wedge \psi\} - \{\chi \wedge \zeta\} \leq \{\phi \wedge \psi \wedge \neg\chi\} + \{\phi \wedge \psi \wedge \chi \wedge \neg\zeta\}$, we see that

$$\begin{aligned}
&\{\widehat{y}_{i,t} \neq V_{i,t}, \forall j \in \text{ANC}(i) : \widehat{y}_{j,t} = V_{j,t}\} - \{y_{i,t} \neq V_{i,t}, \forall j \in \text{ANC}(i) : y_{j,t} = V_{j,t}\} \\
&\quad \leq \left\{\widehat{y}_{i,t} \neq V_{i,t}, y_{i,t} = V_{i,t}, \forall j \in \text{ANC}(i) : \widehat{y}_{j,t} = V_{j,t}\right\} && (8)\\
&\qquad + \left\{\widehat{y}_{i,t} \neq V_{i,t}, y_{i,t} \neq V_{i,t}, \forall j \in \text{ANC}(i) : \widehat{y}_{j,t} = V_{j,t}, \exists j \in \text{ANC}(i) : y_{j,t} \neq V_{j,t}\right\} . && (9)
\end{aligned}$$

We bound the two terms (8) and (9) separately. We can write:

$$\begin{aligned}
(8) &= \{\widehat{y}_{i,t} \neq V_{i,t}, y_{i,t} = V_{i,t}, \forall j \in \text{ANC}(i) : \widehat{y}_{j,t} = V_{j,t} = 1\} \\
&\quad \text{(since } \widehat{y}_{j,t} = V_{j,t} = 0 \text{ for some ancestor } j \text{ implies } \widehat{y}_{i,t} = V_{i,t} = 0) \\
&\leq \{\widehat{y}_{i,t} \neq y_{i,t},\ \mathcal{K}_{i,t}\} ,
\end{aligned}$$

where we have introduced the short-hand $\mathcal{K}_{i,t}$ = "$\forall j \in \text{ANC}(i) : V_{j,t} = 1$". By the same token, we have

$$\begin{aligned}
(9) &= \{\widehat{y}_{i,t} \neq V_{i,t}, y_{i,t} \neq V_{i,t}, \forall j \in \text{ANC}(i) : \widehat{y}_{j,t} = V_{j,t} = 1, \exists j \in \text{ANC}(i) : y_{j,t} \neq V_{j,t}\} \\
&= \{\widehat{y}_{i,t} \neq V_{i,t}, y_{i,t} \neq V_{i,t}, \forall j \in \text{ANC}(i) : \widehat{y}_{j,t} = V_{j,t} = 1, \exists j \in \text{ANC}(i) : \widehat{y}_{j,t} \neq y_{j,t}\} \\
&\leq \{\exists j \in \text{ANC}(i) : \widehat{y}_{j,t} \neq y_{j,t},\ \mathcal{K}_{i,t}\} \\
&\leq \sum_{j \in \text{ANC}(i)} \{\widehat{y}_{j,t} \neq y_{j,t},\ \mathcal{K}_{i,t}\} \\
&\leq \sum_{j \in \text{ANC}(i)} \{\widehat{y}_{j,t} \neq y_{j,t},\ \mathcal{K}_{j,t}\} ,
\end{aligned}$$

where the last inequality holds because $\mathcal{K}_{i,t}$ implies $\mathcal{K}_{j,t}$ for all $j \in \mathrm{ANC}(i)$. Using our bounds for (8) and (9), and summing over i yields

$$\begin{aligned}
&\ell_H(\widehat{y}_t, V_t) - \ell_H(y_t, V_t) \\
&= \sum_{i=1}^{N} \Big(\{\widehat{y}_{i,t} \neq V_{i,t}, \forall j \in \mathrm{ANC}(i) : \widehat{y}_{j,t} = V_{j,t}\} - \{y_{i,t} \neq V_{i,t}, \forall j \in \mathrm{ANC}(i) : y_{j,t} = V_{j,t}\} \Big) \\
&\le \sum_{i=1}^{N} \sum_{j \in \mathrm{ANC}(i) \cup \{i\}} \{\widehat{y}_{j,t} \neq y_{j,t},\, \mathcal{K}_{j,t}\} \\
&= \sum_{i=1}^{N} \{\widehat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t}\} \sum_{j \in \mathrm{SUB}(i)} 1 \\
&= \sum_{i=1}^{N} C_i \{\widehat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t}\}\ .
\end{aligned}$$

We then take expectations and sum over t:

$$\begin{aligned}
\sum_{t=1}^{T} \big(\mathbb{E}\ell_H(\widehat{y}_t, V_t) - \mathbb{E}\ell_H(y_t, V_t) \big) &\le \sum_{t=1}^{T} \sum_{i=1}^{N} C_i \,\mathbb{P}(\widehat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t}) \\
&= \sum_{i=1}^{N} C_i \sum_{t=1}^{T} \mathbb{P}(\widehat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t})\ . \qquad (10)
\end{aligned}$$

Equation (10) is a conveniently simple upper bound on the cumulative regret. This allows us to focus on bounding from above the one-node cumulative expectation $\sum_{t=1}^{T} \mathbb{P}(\widehat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t})$.

For brevity, in the rest of this proof we use the notations $\Delta_{i,t} = u_i^\top x_t$ (the target margin on x_t) and $\widehat{\Delta}_{i,t} = w_{i,t}^\top x_t$ (the algorithm margin on x_t). As we said earlier, our argument centers on proving that for any node i, $\widehat{\Delta}_{i,t}$ is an asymptotically unbiased estimator of $\Delta_{i,t}$, and then on using known large deviation techniques to obtain the stated bound. For this purpose, we need to study both the conditional bias and the conditional variance of $\widehat{\Delta}_{i,t}$.

Recall Figure 3. Since the sequence $x_1, x_2, \ldots$ is fixed, the multilabel vectors V_t are statistically independent. Also, for any $t = 1, 2, \ldots$ and for any node i with parent j, the child's labels $V_{i,i_1}, \ldots, V_{i,i_{Q(i,t-1)}}$ are independent when conditioned on the parent's labels $V_{j,1}, \ldots, V_{j,t-1}$. We use the notation

$$\mathbb{E}_{i,t} = \mathbb{E}[\,\cdot \mid V_{j,1}, \ldots, V_{j,t-1}]\ .$$

By definition of our parametric model (3) we have $\mathbb{E}_{i,t}[(V_{i,i_1}, \ldots, V_{i,i_{Q(i,t-1)}})^\top] = S_{i,Q(i,t-1)}^\top u_i$. Recalling the definition (1) of $w_{i,t}$, this implies (for conciseness we write Q instead of $Q(i,t-1)$)

$$\mathbb{E}_{i,t}[\widehat{\Delta}_{i,t}] = u_i^\top S_{i,Q} S_{i,Q}^\top (I + S_{i,Q} S_{i,Q}^\top + x_t x_t^\top)^{-1} x_t\ .$$

Note that

$$\Delta_{i,t} = \mathbb{E}_{i,t}[\widehat{\Delta}_{i,t}] + u_i^\top (I + x_t x_t^\top)(I + S_{i,Q} S_{i,Q}^\top + x_t x_t^\top)^{-1} x_t = \mathbb{E}_{i,t}[\widehat{\Delta}_{i,t}] + B_{i,t},$$

where $B_{i,t} = u_i^\top (I + x_t x_t^\top)(I + S_{i,Q} S_{i,Q}^\top + x_t x_t^\top)^{-1} x_t$ is the conditional bias of $w_{i,t}$. It is useful to introduce the short-hand notation

$$r_{i,t} = x_t^\top (I + S_{i,Q} S_{i,Q}^\top + x_t x_t^\top)^{-1} x_t\ .$$

Also, in order to stress the dependence[1] of $r_{i,t}$ on $Q = Q(i,t-1)$, we denote it by $r_{i,t,Q}$.

The conditional bias is bounded in the following lemma (proven in the appendix).

Lemma 7 *With the notation introduced so far, we have*

$$B_{i,t} \le \sqrt{r_{i,t,Q}} + |\Delta_{i,t}|\, r_{i,t,Q} \ .$$

As far as the conditional variance of $\widehat{\Delta}_{i,t}$ is concerned, from Figure 3 we see that

$$\widehat{\Delta}_{i,t} = \sum_{k=1}^{Q} V_{i,i_k} Z_{i,t,k} \, ,$$

where

$$Z_{i,t}^\top = (Z_{i,t,1}, \ldots, Z_{i,t,Q}) = S_{i,Q}^\top \left(I + S_{i,Q} S_{i,Q}^\top + x_t x_t^\top \right)^{-1} x_t \ . \tag{11}$$

The next lemma (proven in the appendix) handles the conditional variance $\|Z_{i,t}\|^2$.

Lemma 8 *With the notation introduced so far, we have*

$$\|Z_{i,t}\|^2 \le r_{i,t,Q} \ .$$

Armed with these two lemmas, we proceed through our large deviation argument.

We can write

$$\begin{aligned}
&\{\hat{y}_{i,t} \ne y_{i,t}, \mathcal{K}_{i,t}\} \\
&\quad \le \ \left\{ \widehat{\Delta}_{i,t}\, \Delta_{i,t} \le 0, \ \mathcal{K}_{i,t} \right\} \\
&\quad \le \ \left\{ |\widehat{\Delta}_{i,t} - \Delta_{i,t}| \ge |\Delta_{i,t}|, \ \mathcal{K}_{i,t} \right\} \\
&\quad \le \ \left\{ |\widehat{\Delta}_{i,t} + B_{i,t} - \Delta_{i,t}| \ge |\Delta_{i,t}| - |B_{i,t}|, \ \mathcal{K}_{i,t} \right\} \\
&\quad \le \ \left\{ |\widehat{\Delta}_{i,t} + B_{i,t} - \Delta_{i,t}| \ge |\Delta_{i,t}|/2, \ \mathcal{K}_{i,t} \right\} + \{ |B_{i,t}| \ge |\Delta_{i,t}|/2, \ \mathcal{K}_{i,t} \} \, .
\end{aligned} \tag{12}$$

We can further bound the second term of (12) by using Lemma 7. We obtain

$$\begin{aligned}
\left\{ |B_{i,t}| \ge |\Delta_{i,t}|/2, \ \mathcal{K}_{i,t} \right\} &\le \ \{ \sqrt{r_{i,t,Q}} + |\Delta_{i,t}|\, r_{i,t,Q} \ge |\Delta_{i,t}|/2, \ \mathcal{K}_{i,t} \} \\
&\le \ \left\{ \left(r_{i,t,Q} \ge |\Delta_{i,t}|^2/16 \vee r_{i,t,Q} \ge 1/4 \right), \ \mathcal{K}_{i,t} \right\} \\
&= \ \left\{ r_{i,t,Q} \ge |\Delta_{i,t}|^2/16, \ \mathcal{K}_{i,t} \right\}
\end{aligned}$$

1. As it turns out, many of the quantities appearing in the present proof, including the bias term $B_{i,t}$ and the variance vector $Z_{i,t}$ defined later on, are algorithm-dependent, hence they do actually depend on $Q = Q(i,t-1)$. However, this dependence is made notationally explicit only for the quantity $r_{i,t} = r_{i,t,Q}$ since, we believe, this specific dependence is key to the proof.

the equality following from the fact that $|\Delta_{i,t}|^2/16 \le 1/16 < 1/4$. We plug back into (12), take expectations, and sum over t. We have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{T}\{\hat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t}\}\right] \\
&\le \ \mathbb{E}\left[\sum_{t=1}^{T}\left(\left\{|\widehat{\Delta}_{i,t}+B_{i,t}-\Delta_{i,t}| \ge |\Delta_{i,t}|/2,\, \mathcal{K}_{i,t}\right\}+\left\{r_{i,t,Q} \ge |\Delta_{i,t}|^2/16,\, \mathcal{K}_{i,t}\right\}\right)\right] \\
&= \ \mathbb{E}\left[\sum_{t=1}^{T}\{\mathcal{K}_{i,t}\}\,\mathbb{E}_{i,t}\left\{|\widehat{\Delta}_{i,t}+B_{i,t}-\Delta_{i,t}| \ge |\Delta_{i,t}|/2\right\}\right] \qquad (13) \\
&\quad + \mathbb{E}\left[\sum_{t=1}^{T}\left\{r_{i,t,Q} \ge |\Delta_{i,t}|^2/16,\, \mathcal{K}_{i,t}\right\}\right], \qquad (14)
\end{aligned}$$

where in (13) we used the fact that $\mathcal{K}_{i,t}$ is determined given $V_{\mathrm{PAR}(i),1},\dots,V_{\mathrm{PAR}(i),t-1}$.

We now bound the two expectations (13) and (14) separately. Let $j = \mathrm{PAR}(i)$. To bound the first expectation, we exploit the fact that $V_{i,i_1},\dots,V_{i,i_Q}$ are independent under the law $P_{i,t} = \mathbb{P}\big(\cdot \mid V_{j,1},\dots,V_{j,t-1}\big)$, and $Z_{i,t,1},\dots,Z_{i,t,Q}$ defined in (11) are determined given $V_{j,1},\dots,V_{j,t-1}$. Hence, we can apply Chernoff-Hoeffding inequality (Hoeffding, 1963) to the sum $\widehat{\Delta}_{i,t} = V_{i,i_1}Z_{i,t,1}+\dots+V_{i,i_Q}Z_{i,t,Q}$ of independent random variables, where $\mathbb{E}_{i,t}[\widehat{\Delta}_{i,t}] = \Delta_{i,t}-B_{i,t}$ and $(V_{i,i_1}Z_{i,t,1})^2+\dots+(V_{i,i_Q}Z_{i,t,Q})^2 \le r_{i,t,Q}$ by Lemma 8. Recalling that $\Delta_i^2 = \min_{t=1,\dots,T}\Delta_{i,t}^2$, we can write

$$\sum_{t=1}^{T}\{\mathcal{K}_{i,t}\}\mathbb{P}_{i,t}\Big(|\widehat{\Delta}_{i,t}+B_{i,t}-\Delta_{i,t}| \ge |\Delta_{i,t}|/2\Big) \le 2\sum_{t=1}^{T}\{\mathcal{K}_{i,t}\}\exp\left(-\frac{\Delta_i^2}{8\,r_{i,t,Q}}\right).$$

This quantity can be further upper bounded using the following lemma (proven in the appendix).

Lemma 9 *Let α, M be positive constants. Then*

$$\max\left\{\sum_{t=1}^{n} e^{-\alpha/a_t} \,:\, a_1 \ge 0,\dots,a_n \ge 0, \sum_{t=1}^{n} a_t = M\right\} \le \frac{M}{e\alpha}.$$

If we let

$$M_i = \sum_{t=1}^{T}\{\mathcal{K}_{i,t}\}r_{i,t,Q} = \sum_{t\,:\,\{\mathcal{K}_{i,t}\}=1} r_{i,t,Q}$$

we immediately see that Lemma 9 implies

$$\sum_{t=1}^{T}\{\mathcal{K}_{i,t}\}\exp\left(-\frac{\Delta_i^2}{8\,r_{i,t,Q}}\right) = \sum_{t\,:\,\{\mathcal{K}_{i,t}\}=1}\exp\left(-\frac{\Delta_i^2}{8\,r_{i,t,Q}}\right) \le \frac{8}{e\Delta_i^2}M_i.$$

Therefore,

$$(13) \le \frac{16}{e\Delta_i^2}\,\mathbb{E}M_i.$$

To bound (14) we can argue as follows (note that, by definition, $r_{i,t,Q} \geq 0$, since it is the value of a quadratic form with a positive definite matrix):

$$
\begin{aligned}
M_i &= \sum_{t=1}^{T} \{\mathcal{K}_{i,t}\}\, r_{i,t,Q} \\
&= \sum_{t=1}^{T} \{r_{i,t,Q} \geq \Delta_i^2/16,\, \mathcal{K}_{i,t}\}\, r_{i,t,Q} + \sum_{t=1}^{T} \{r_{i,t,Q} < \Delta_i^2/16,\, \mathcal{K}_{i,t}\} r_{i,t,Q} \\
&\geq \sum_{t=1}^{T} \{r_{i,t,Q} \geq \Delta_i^2/16,\, \mathcal{K}_{i,t}\}\, \Delta_i^2/16\ .
\end{aligned}
$$

Hence

$$
(14) = \mathbb{E}\left[\sum_{t=1}^{T} \{r_{i,t,Q} \geq \Delta_i^2/16,\, \mathcal{K}_{i,t}\}\right] \leq \frac{16}{\Delta_i^2}\, \mathbb{E}\, M_i\ .
$$

We have thus obtained the following bound

$$
\sum_{t=1}^{T} \mathbb{P}(\hat{y}_{i,t} \neq y_{i,t},\, \mathcal{K}_{i,t}) \leq \frac{16(1+1/e)}{\Delta_i^2}\, \mathbb{E}\, M_i\ .
$$

To conclude, we need to upper bound $\mathbb{E}\, M_i$. Observe that M_i is a sum only over time steps t such that $\{\mathcal{K}_{i,t}\} = 1$; i.e., over those t such that the weight vector $w_{i,t}$ gets actually updated. Therefore, since we would like to relate M_i to the spectral structure of the data correlation matrices $S_{i,Q(i,T)} S_{i,Q(i,T)}^\top$, we can proceed through the standard upper bounding argument (Azoury and Warmuth, 2001; Cesa-Bianchi et al., 2002) given below.

$$
\begin{aligned}
M_i &= \sum_{t=1}^{T} \{\mathcal{K}_{i,t}\} r_{i,t,Q} \\
&= \sum_{t=1}^{T} \left(1 - \frac{\det(I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^\top)}{\det(I + S_{i,Q(i,t)} S_{i,Q(i,t)}^\top)}\right) \\
&\quad \text{(using Lemma 2, part 1, in Lai and Wei, 1982)} \\
&\leq \sum_{t=1}^{T} \log \frac{\det(I + S_{i,Q(i,t)} S_{i,Q(i,t)}^\top)}{\det(I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^\top)} \qquad \text{(since } 1 - x \leq -\log x \text{ for all } x > 0\text{)} \\
&= \log \frac{\det(I + S_{i,Q(i,T)} S_{i,Q(i,T)}^\top)}{\det(I)} \\
&= \sum_{j=1}^{d} \log(1 + \lambda_{i,j})\ .
\end{aligned}
$$

Putting together as in (10) concludes the proof. □

Our analysis of Theorem 2 is similar in spirit to the work of Lai et al. (1979) on least-squares regression. In particular, they also assume the sequence $x_1, x_2, \ldots$ be arbitrary while the real-valued labels y_t are defined as $y_t = u^\top x_t + \varepsilon_t$, where ε_t are i.i.d. random variables with finite variance.

A regret bound similar to the one established by Theorem 2 can be proven for the zero-one loss using the fact that this loss can be crudely upper bounded by the H-loss (with all cost coefficients set to 1). Indeed, a more direct (and sharper) analysis could be performed for the zero-one loss, following the same lines as the proof of Theorem 2. As far as the symmetric difference loss ℓ_Δ is concerned, a regret analysis might be obtained through a method we developed in earlier work (Cesa-Bianchi et al., 2004). As a matter of fact, the analysis by Cesa-Bianchi et al. (2004) rests on several side assumptions about the way data $x_1, \ldots, x_T$ are generated. We have been unable to apply the theoretical arguments employed in the present paper to ℓ_Δ. In any case, since these two loss functions are unable to capture the hierarchical nature of our classification problem, we believe the resulting bounds are less relevant to this paper.

8. Experimental Results

We tested the empirical performance of our on-line algorithm on data sets extracted from two popular corpora of free-text documents. The first data set consists of the first (in chronological order) 100,000 newswire stories from the Reuters Corpus Volume 1 (Reuters, 2000). The associated taxonomy of labels, which are the document topics, contains 101 nodes organized in a forest of 4 trees. The forest is shallow: the longest path has length 3 and the distribution of nodes, sorted by increasing path length, is $\{0.04, 0.53, 0.42, 0.01\}$. The average number of paths in the multilabel of an instance is 1.5. For this data set we used the bag-of-words vectorization performed by Xerox Research Center Europe within the EC project KerMIT (see Cesa-Bianchi et al., 2003, for details). The 100,000 documents were divided into 5 equally sized groups of chronologically consecutive documents. We then used each adjacent pair of groups as training and test set for an experiment (here the fifth and first group are considered adjacent), and then averaged the test set performance over the 5 experiments.

The second data set includes the documents classified in the nodes of the subtree rooted in "Quality of Health Care" (MeSH code N05.715) of the OHSUMED corpus of medical abstracts (Hersh, 1994). Since OHSUMED is not quite a tree but a directed acyclic graph, and since the H-loss is defined for trees only, we removed from this OHSUMED fragment the few nodes that did not have a unique path to the root. This produced a hierarchy with 94 classes and a data set with 55,503 documents. The choice of this specific subtree was motivated by its structure only; in particular: the subtree depth is 4, the distribution of nodes (sorted by increasing path length) is $\{0.26, 0.37, 0.22, 0.12, 0.03\}$, and there is a reasonable number of partial and multiple path multilabels (the average number of paths per instance is 1.53). The vectorization of the documents was carried out similarly to RCV1. After tokenization, we removed all stopwords and also those words that did not occur at least 3 times in the corpus. Then, we vectorized the documents using the BOW library (McCallum, 2004) with a $\log(1+\text{TF})\log(\text{IDF})$ encoding. We ran 5 experiments by randomly splitting the corpus in a training set of 40,000 documents and a test set of 15,503 documents. Test set performances are averages over these 5 experiments. In the training set we kept more documents than in the RCV1 splits since the OHSUMED corpus turned out to be a harder classification problem than RCV1. In both data sets instances have been normalized to unit length.

Since the space complexity of H-RLS grows linearly with training time, due to the need of storing each training instance in the matrices $S_{i,t}$ —see (1), we had to make some modifications to the algorithm in order to be able to carry out experiments on data sets of this size. For this purpose, we have developed SH-RLS, a space-efficient variant of H-RLS that we used in all of our experiments.

The performance of SH-RLS is compared against five baseline algorithms: a flat and a hierarchical version of the Perceptron algorithm (Novikov, 1962; Rosenblatt, 1958), a flat and a hierarchical version of Vapnik's support vector machine (see, e.g., Vapnik, 1998; Schölkopf and Smola, 2002), and a flat version of SH-RLS. Note that support vector machines are not trained incrementally; we include them in our pool of baseline algorithms to show that on-line learners, processing each training example only once, can have a performance level close to that of batch learners.

Note also that, unlike our theoretical analysis based on cumulative regret, in the experiments we distinguish a training phase, where the hierarchical classifiers are built, and a test phase, where the performance of the hierarchical classifiers obtained in the training phase is measured on fresh data. This allows us to use a single measure, the test error, to compare both batch and incremental learners.

The first algorithm we consider, H-PERC, is a simple hierarchical version of the Perceptron. Its functioning differs from H-RLS described in Figure 3 only in the way weights are updated. In particular, H-PERC learns a hierarchical classifier by training a linear-threshold classifier at each node via the Perceptron algorithm. At the beginning, the weight vector of each node classifier is set to the zero vector, $w_{i,1} = (0,\dots,0)$ for $i = 1,\dots,N$. Upon receiving an example (x_t, v_t), H-PERC considers for an update only those classifiers sitting at nodes i satisfying either $i \in \text{ROOT}(G)$ or $v_{\text{PAR}(i),t} = 1$. If $\{w_{i,t}^\top x_t \geq 0\} \neq v_{i,t}$ for such a node i, then the weight vector $w_{i,t}$ is updated using the Perceptron rule $w_{i,t+1} = w_{i,t} + v_{i,t} x_t$; on the other hand, if $\{w_{i,t}^\top x_t \geq 0\} = v_{i,t}$, then $w_{i,t+1} = w_{i,t}$ (no update takes place at node i).

During the test phase, H-PERC computes the multilabel $\widehat{y} = (\widehat{y}_1,\dots,\widehat{y}_N)$ of a test instance x using the same top-down process described in Figure 3,

$$\widehat{y}_i = \begin{cases} \{w_i^\top x \geq 0\} & \text{if } i \text{ is a root node,} \\ \{w_i^\top x \geq 0\} & \text{if } i \text{ is not a root node and } \hat{y}_j = 1 \text{ for } j = \text{PAR}(i), \\ 0 & \text{if } i \text{ is not a root node and } \hat{y}_j = 0 \text{ for } j = \text{PAR}(i). \end{cases} \tag{15}$$

The second incremental algorithm considered is SH-RLS, our sparse variant of H-RLS. The two algorithms, H-RLS and SH-RLS operate in the same way (see Figure 3) with the only difference that SH-RLS performs fewer updates in the training phase. In particular, given a training example (x_t, v_t), both algorithms consider for an update only those classifiers sitting at nodes i satisfying either $i \in \text{ROOT}(G)$ or $v_{\text{PAR}(i),t} = 1$. However, whereas H-RLS would update the weight $w_{i,t}$ of all such nodes i, SH-RLS also requires the margin condition $|w_{i,t}^\top x_t| \leq \sqrt{(5\ln t)/N_{i,t}}$, where $N_{i,t}$ is the number of instances stored at node i up to time $t-1$. The choice of the margin threshold $\sqrt{(5\ln t)/N_{i,t}}$ is motivated by Cesa-Bianchi et al. (2003) via a large deviation analysis.

We also tested a hierarchical version of SVM (denoted by H-SVM) in which each node is an SVM classifier trained using a batch version of our hierarchical learning protocol. More precisely, each node i was trained only on those examples (x_t, v_t) such that $v_{\text{PAR}(i),t} = 1$. The resulting set of linear-threshold functions was then evaluated on the test set using the hierarchical classification scheme (15). We tried both the C and ν parametrizations (Schölkopf et al., 2000) for SVM and found the setting $C = 1$ to work best[2] for our data (recall that all instances x_t are normalized, $\|x_t\| = 1$).

We finally tested the "flat" variants of H-PERC, SH-RLS and H-SVM, denoted by PERC, S-RLS and SVM, respectively. In these variants, each node is trained and evaluated independently of the others, disregarding all taxonomical information. All SVM experiments were carried out using the libSVM implementation (Chang and Lin, 2004) and all the algorithms ran with a linear kernel. The

2. It should be emphasized that this tuning of C was actually chosen in hindsight across the interval [0.1,10] with no cross-validation.

RCV1			
Algorithm	zero-one loss	uniform H-loss	Δ-loss
PERC	0.702(±0.045)	1.196(±0.127)	1.695(±0.182)
H-PERC	0.655(±0.040)	1.224(±0.114)	1.861(±0.172)
S-RLS	0.559(±0.005)	0.981(±0.020)	1.413(±0.033)
SH-RLS	**0.456**(±**0.010**)	**0.743**(±**0.026**)	**1.086**(±**0.036**)
SVM	0.482(±0.009)	0.790(±0.023)	1.173(±0.051)
H-SVM	0.440(±0.008)	0.712(±0.021)	1.050(±0.027)

OHSUMED			
Algorithm	zero-one loss	uniform H-loss	Δ-loss
PERC	0.899(±0.024)	1.938(±0.219)	2.639(±0.226)
H-PERC	0.846(±0.024)	1.560(±0.155)	2.528(±0.251)
S-RLS	0.873(±0.004)	1.814(±0.024)	2.627(±0.027)
SH-RLS	**0.769**(±**0.004**)	**1.200**(±**0.007**)	**1.957**(±**0.011**)
SVM	0.784(±0.003)	1.206(±0.003)	1.872(±0.005)
H-SVM	0.759(±0.002)	1.170(±0.005)	1.910(±0.007)

Table 1: Experimental results on two hierarchical text classification tasks under various loss functions. We report average test errors along with standard deviations (in parentheses). In bold are the best performance figures among the incremental algorithms (all incremental algorithms were run for one epoch over the training data).

performance of these algorithms was evaluated against three different loss measures (see Table 1). The first two losses are the zero-one loss and the H-loss with cost coefficients set to 1 (denoted by uniform H-loss in Table 1). The third loss is the symmetric difference loss (Δ-loss in Table 1).

A few remarks on Table 1 are in order at this point. As expected, H-SVM performs best, but the good performance of SVM (flat support vector machine) is surprising. As for the incremental algorithms, SH-RLS performs better than its flat variant SH-RLS, and far better than both H-PERC and PERC. In addition, and perhaps surprisingly, after a single epoch of training SH-RLS performs generally better than SVM and comes reasonably close to the performance of H-SVM. Finally, note that the running times of both S-RLS and SH-RLS scale quadratically in the number of stored instances, whereas the running time of Perceptrons scales only linearly. Thus, as usual, the performance benefit has to be traded-off against computational cost.

To give an idea of how flat and hierarchical algorithms compare in terms of running times, we mention that hierarchical algorithms turned out to be roughly four times faster than the corresponding flat algorithms running on the same data sets.

The (uniform) H-loss does not provide any information on the distribution of mistakes across the different hierarchy levels. Therefore, we counted the "H-loss mistakes" made at each level, distinguishing between false positive (FP) and false negative (FN) mistakes. Fix an example (x,v) and let $\widehat{y}$ be the guessed multilabel. Then node i makes an H-loss mistake on (x,v) if

$$\widehat{y}_i \neq v_i \wedge \widehat{y}_j = v_j = 1,\ j \in \text{ANC}(i)\ .$$

RCV1				
Depth		H-PERC	SH-RLS	H-SVM
0	FP	4144(±2431)	1449(±79)	1769(±163)
	FN	2690(±851)	2436(±112)	2513(±148)
1	FP	6769(±2509)	1361(±108)	1317(±81)
	FN	7961(±838)	8135(±476)	7260(±450)
2	FP	1161(±261)	413(±32)	380(±28)
	FN	1513(±833)	937(±51)	624(±23)
3	FP	161(±314)	14(±16)	20(±26)
	FN	88(±44)	115(±31)	94(±24)

OHSUMED				
Depth		H-PERC	SH-RLS	H-SVM
0	FP	7916(±2638)	3192(±88)	3062(±60)
	FN	12639(±1418)	12888(±64)	12587(±49)
1	FP	1816(±730)	828(±14)	839(±11)
	FN	1606(±373)	1594(±33)	1542(±25)
2	FP	88(±20)	30(±6)	37(±7)
	FN	86(±31)	54(±4)	55(±2)
3	FP	10(±5)	2(±1)	3(±1)
	FN	16(±11)	13(±3)	14(±1)
4	FP	3(±2)	1(±1)	4(±1)
	FN	5(±6)	1(±1)	2(±1)

Table 2: Distribution across the hierarchy levels of false positive (FP) and false negative (FN) H-loss mistakes on the two hierarchical text classification tasks RCV1 and OHSUMED. We report the average number of mistakes at each level of the hierarchy trees with standard deviation in parentheses (recall that we made 5 experiments on different splits of the two data set).

Thus, node *i* makes a false positive mistake if

$$\widehat{y}_i = 1 \land v_i = 0 \land \widehat{y}_j = v_j = 1,\ j \in \text{ANC}(i)$$

and makes a false negative mistake if

$$\widehat{y}_i = 0 \land v_i = 1 \land \widehat{y}_j = v_j = 1,\ j \in \text{ANC}(i)\ .$$

Table 2 shows the H-loss mistake distribution for RCV1 and OHSUMED over hierarchy levels.

The average values contained in Table 2 are also plotted in Figure 4. A quick visual comparison reveals the close similarity between the distributions obtained by SH-RLS and H-SVM, whereas the behavior of H-PERC looks quite different.

9. Conclusions, Ongoing Research, and Open Problems

We have introduced H-RLS, a new on-line algorithm for hierarchical classification that maintains and updates regularized least-squares estimators on the nodes of a taxonomy. The linear-threshold classifications, obtained from the estimators, are combined to produce a single hierarchical multilabel through a simple top-down evaluation model.

Our algorithm is suitable for learning multilabels that include multiple and/or partial paths on the taxonomy. To properly evaluate hierarchical classifiers in this framework we have defined the H-loss, a new hierarchical loss function, with cost coefficients possibly associated to each taxonomy node—see Remark 5.

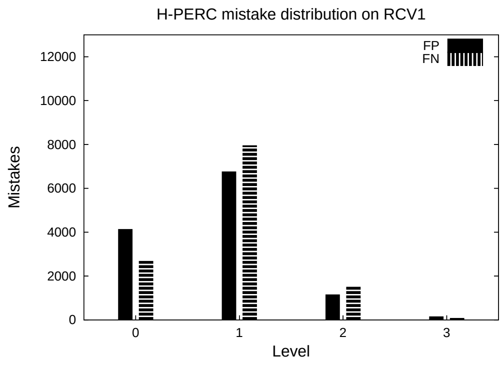
Our main theoretical result states that, on any sequence of instances, the cumulative H-loss of H-RLS is never much bigger than the cumulative H-loss of a reference classifier tuned with the parameters of the stochastic process generating the multilabels for the given sequence of instances. Our theoretical findings are complemented by experiments on the hierarchical classification of textual data, in which we compare the performance of a sparsified variant of H-RLS to that of standard batch and incremental learners, such as simple hierarchical versions of the Perceptron algorithm and the SVM. The experiments show that one epoch of training of our algorithm is enough to achieve a performance close to that of the hierarchical SVM.

Our investigation leaves a number of open questions. The first open question is the derivation of a hierarchical algorithm especially designed to minimize the H-loss. We are currently exploring efficient ways to approximate the Bayes optimal classifier for the H-loss, given our data model. Since such optimal classifier turns out to be remarkably different from the hierarchical classifiers produced by H-RLS, a related theoretical question is to prove any reasonable bound on the regret with respect to the Bayes optimal classifier.

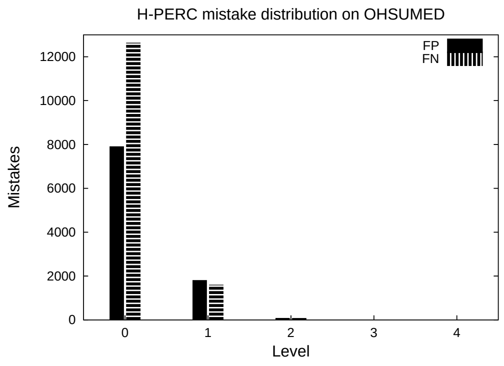
Additional open problems concern the data model. First, it would be useful to modify the label-generating model to introduce dependencies among the children's labels. This could allow a better fitting of data sets when the rate of multiple paths in multilabels is limited. Second, further investigation, both of empirical and theoretical nature, might be devoted to the issue of using regularized logistic regressors at each node.

Acknowledgments

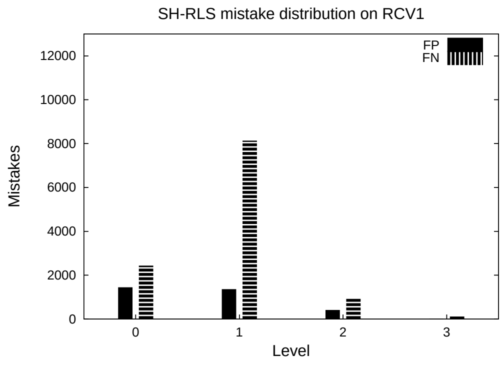
The authors would like to thank Michael Collins for his timely editorial work, as well as the anonymous reviewers, whose comments and suggestions greatly improved the presentation of this paper.



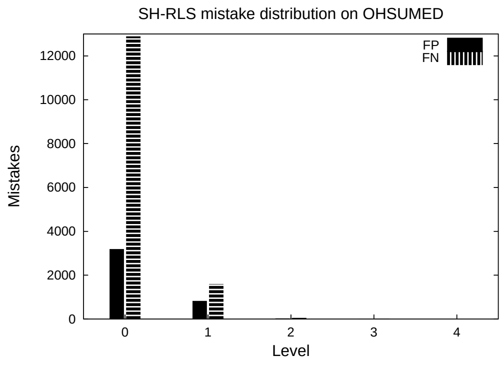
(a)



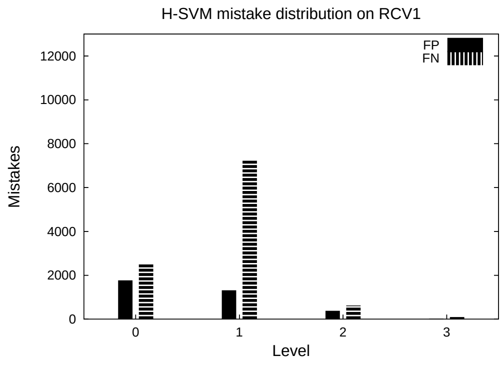
(b)



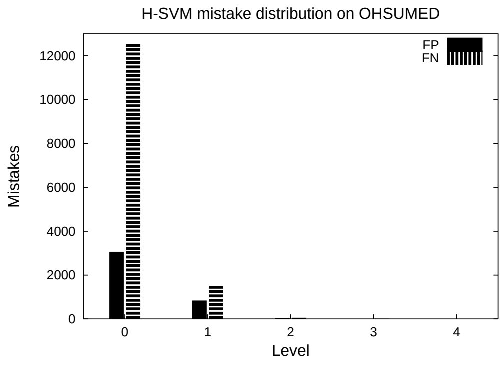
(c)



(d)



(e)



(f)

Figure 4: Plot of the average values contained in Table 2 for the H-loss mistake distribution over hierarchy levels.

This work was supported in part by the IST Programme of the European Community under the PASCAL Network of Excellence IST-2002-506778. This publication only reflects the authors' views.

Appendix A

This appendix contains the proofs of Lemmas 7, 8, and 9 mentioned in the main text. Throughout this appendix A denotes the positive definite matrix $I + S_{i,Q} S_{i,Q}^\top$, while r denotes the quadratic form $x_t^\top (A + x_t x_t^\top)^{-1} x_t$.

Proof of Lemma 7

We have

$$\begin{aligned} B_{i,t} &= u_i^\top (I + x_t x_t^\top)(A + x_t x_t^\top)^{-1} x_t \\ &= u_i^\top (A + x_t x_t^\top)^{-1} x_t + \Delta_{i,t}\, r \\ &\le \sqrt{x_t^\top (A + x_t x_t^\top)^{-2} x_t} + |\Delta_{i,t}|\, r \\ &\le \sqrt{r} + |\Delta_{i,t}|\, r \end{aligned}$$

where the first inequality follows from $u_i^\top z \le \max_{\|u_i\|=1} u_i^\top z = \|z\|$, with $z = (A + x_t x_t^\top)^{-1} x_t$, and the second inequality follows from $x^\top (A + xx^\top)^{-2} x \le x^\top (A + xx^\top)^{-1} x$, holding for any x and for any positive definite matrix A whose eigenvalues are not smaller than 1 (notice that this condition makes $(A + xx^\top)^{-1} - (A + xx^\top)^{-2}$ a positive semidefinite matrix). □

Proof of Lemma 8

Setting for brevity $H = S_{i,Q}^\top A^{-1} x_t$ and $a = x_t^\top A^{-1} x_t$ we can write

$$\begin{aligned} \|Z_{i,t}\|^2 &= x_t^\top \left(A + x_t x_t^\top\right)^{-1} S_{i,Q} S_{i,Q}^\top \left(A + x_t x_t^\top\right)^{-1} x_t \\ &= x_t^\top \left(A^{-1} - \frac{A^{-1} x_t x_t^\top A^{-1}}{1 + x_t^\top A^{-1} x_t}\right) S_{i,Q} S_{i,Q}^\top \left(A^{-1} - \frac{A^{-1} x_t x_t^\top A^{-1}}{1 + x_t^\top A^{-1} x_t}\right) x_t \\ &\quad \text{(by the Sherman-Morrison formula—e.g., Horn and Johnson, 1985, chap. 0)} \\ &= H^\top H - \frac{a}{1+a} H^\top H - \frac{a}{1+a} H^\top H + \frac{a^2}{(1+a)^2} H^\top H \\ &= \frac{H^\top H}{(1+a)^2} \\ &= \frac{x_t^\top A^{-1} S_{i,Q} S_{i,Q}^\top A^{-1} x_t}{(1+a)^2} \\ &= \frac{x_t^\top A^{-1/2} A^{-1/2} S_{i,Q} S_{i,Q}^\top A^{-1/2} A^{-1/2} x_t}{(1+a)^2} \\ &\le \frac{\left\|A^{-1/2} x_t\right\| \left\|A^{-1/2} S_{i,Q} S_{i,Q}^\top A^{-1/2}\right\| \left\|x_t^\top A^{-1/2}\right\|}{(1+a)^2} \\ &= \frac{a}{(1+a)^2} \left\|A^{-1/2} S_{i,Q} S_{i,Q}^\top A^{-1/2}\right\|, \qquad (16) \end{aligned}$$

where $\left\|A^{-1/2}S_{i,Q}S_{i,Q}^{\top}A^{-1/2}\right\|$ is the spectral norm of matrix $A^{-1/2}S_{i,Q}S_{i,Q}^{\top}A^{-1/2}$.

We continue by bounding the two factors in (16). Observe that

$$\frac{a}{(1+a)^2} \le \frac{a}{1+a} = r$$

where the equality derives again from the Sherman-Morrison formula. As far as the second factor is concerned, we just note that the two matrices $A^{-1/2}$ and $S_{i,Q}S_{i,Q}^{\top}$ have the same eigenvectors. Furthermore, if λ_j is an eigenvalue of $S_{i,Q}S_{i,Q}^{\top}$, then $1/\sqrt{1+\lambda_j}$ is an eigenvalue of $A^{-1/2}$. Therefore

$$\left\|A^{-1/2}S_{i,Q}S_{i,Q}^{\top}A^{-1/2}\right\| = \max_j \frac{1}{\sqrt{1+\lambda_j}} \times \lambda_j \times \frac{1}{\sqrt{1+\lambda_j}} \le 1 \ .$$

Substituting into (16) yields $\|Z_{i,t}\|^2 \le r$, as desired. □

Proof of Lemma 9

From a simple Kuhn-Tucker analysis[3] it follows that if a_t is larger than 0 at the maximum, then a_t takes some constant value $\beta > 0$ (independent of t). Hence the maximizing vector $(a_1, a_2, \ldots, a_n)$ has components which can take only two possible values: $a_t = 0$ or $a_t = \beta$. Let us denote by N^+ the number of $t : a_t = \beta$. At the maximum we can write

$$M = \sum_{t=1}^{n} a_t = \sum_{t\,:\,a_t=\beta} a_t + \sum_{t\,:\,a_t\to 0^+} a_t = \beta N^+$$

i.e., $\beta = M/N^+$. Hence, at the maximum

$$\begin{aligned}
\sum_{t=1}^{n} e^{-\alpha/a_t} &= \sum_{t\,:\,a_t=\beta} e^{-\alpha/a_t} + \sum_{t\,:\,a_t=0^+} e^{-\alpha/a_t} \\
&= \sum_{t\,:\,a_t=\beta} e^{-\alpha/\beta} \\
&= N^+ e^{-\alpha/\beta} \\
&= N^+ e^{-\alpha N^+/M} \ .
\end{aligned}$$

Since N^+ is not determined by this argument, we can write

$$\max\left\{\sum_{t=1}^{n} e^{-\alpha/a_t} \,:\, a_1 \ge 0, \ldots, a_n \ge 0, \sum_{t=1}^{n} a_t = M\right\} \le \max_{x\ge 0} x e^{-\alpha x/M} = \frac{M}{e\alpha}$$

thereby concluding the proof. □

3. The function $f(a) = e^{-\alpha/a}$ is not defined when $a = 0$. However, it is clearly possible to extend f by defining $f(0) = 0$, preserving (one-sided) differentiability.

References

K. S. Azoury and M. K. Warmuth. Relative loss bounds for on-line density estimation with the exponential familiy of distributions. *Machine Learning*, 43(3):211–246, 2001.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order perceptron algorithm. *SIAM Journal of Computing.*, 43(3):640–668, 2005.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. Learning probabilistic linear-threshold classifiers via selective sampling. In *Proceedings of the 16th Annual Conference on Computational Learning Theory*, pages 373–386. LNAI 2777, Springer, 2003.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. Regret bounds for hierarchical classification with linear-threshold functions. In *Proceedings of the 17th Annual Conference on Computational Learning Theory*, pages 93–108. LNAI 3120, Springer, 2004.

C. C. Chang and C. J. Lin. Libsvm: a library for support vector machines, 2004.
Available electronically at `http://www.csie.ntu.edu.tw/∼cjlin/libsvm/`.

O. Dekel, J. Keshet, and Y. Singer. An efficient online algorithm for hierarchical phoneme classification. In *Proceedings of the 1st International Workshop on Machine Learning for Multimodal Interaction*, pages 146-158. Springer LNAI 3361, 2005.

O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *Proceedings of the 21st International Conference on Machine Learning*. Omnipress, 2004.

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*, pages 256–263. ACM Press, 2000.

M. Granitzer. *Hierarchical Text Classification using Methods from Machine Learning*. PhD thesis, Graz University of Technology, 2003.

W. R. Hersh. The OHSUMED test collection, 1994.
Available electronically at `http://medir.ohsu.edu/pub/ohsumed/`.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

T. Hofmann, L. Cai, and M. Ciaramita. Learning with taxonomies: classifying documents and words. In *NIPS 2003: Workshop on syntax, semantics, and statistics*, 2003.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the 14th International Conference on Machine Learning*, pages 170–178, 1997.

T. L. Lai, H. Robbins, and C. Z. Wei. Strong consistency of least squares estimates in multiple regression. *Proceedings of the National Academy of Sciences USA*, 75(7):3034–3036, 1979.

T. L. Lai and C. Z. Wei. Least squares estimates in stochastic regression models with applications to identification and control of dynamic systems. *The Annals of Statistics*, 10(1):154–166, 1982.

A. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering, 2004.
Available electronically at `http://www-2.cs.cmu.edu/∼mccallum/bow/`.

A. K. McCallum, R. Rosenfeld, T. M. Mitchell, and A. Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the 15th International Conference on Machine Learning*, pages 359–367. Morgan Kaufmann Publishers, 1998.

D. Mladenic. Turning yahoo into an automatic web-page classifier. In *Proceedings of the 13th European Conference on Artificial Intelligence*, pages 473–474, 1998.

A. B. J. Novikov. On convergence proofs on Perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata, vol. XII*, pages 615–622, 1962.

Reuters. Reuters corpus volume 1, 2000. Available electronically at
`http://about.reuters.com/researchandstandards/corpus/`.

R. Rifkin, G. Yeo, and T. Poggio. Regularized least squares classification. *Advances in Learning Theory: Methods, Model and Applications. NATO Science Series III: Computer and Systems Sciences*, 190:131–153, 2003.

F. Rosenblatt. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

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

B. Schölkopf and A. Smola. *Learning with kernels*. MIT Press, 2002.

B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.

A. Sun and E. P. Lim. Hierarchical text classification and evaluation. In *Proceedings of the 2001 International Conference on Data Mining*, pages 521–528. IEEE Press, 2001.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

V. Vovk. Competitive on-line statistics. *International Statistical Review*, 69:213–248, 2001.