On Maximum Margin Hierarchical Multilabel Classification

Juho Rousu*
Department of Computer Science
Royal Holloway University of London
TW20 0EX, United Kingdom
juho@cs.rhul.ac.uk

Craig Saunders
Electronics and Computer Science
University of Southampton
SO17 1BJ, United Kingdom
cjs@ecs.soton.ac.uk

Sandor Szedmak
Electronics and Computer Science
University of Southampton
SO17 1BJ, United Kingdom
ss03v@ecs.soton.ac.uk

John Shawe-Taylor
Electronics and Computer Science
University of Southampton
SO17 1BJ, United Kingdom
jst@ecs.soton.ac.uk

Abstract

We present work in progress towards maximum margin hierarchical classification where the objects are allowed to belong to more than one category at a time. The classification hierarchy is represented as a Markov network equipped with an exponential family defined on the edges. We present a variation of the maximum margin multilabel learning framework, suited to the hierarchical classification task and allows efficient implementation via gradient-based methods. We compare the behaviour of the proposed method to the recently introduced hierarchical regularized least squares classifier as well as two SVM variants in Reuter's news article classification.

Often in hierarchical classification, the object to be classified is assumed to belong to exactly one (leaf) node in the hierarchy (c.f. [**?**, **?**, **?**]). Following [1], in this paper we consider the more general case where a single object can be classified into several categories in the hierarchy, to be specific, the multilabel is a *union of partial paths* in the hierarchy. For example, a news article about David and Victoria Beckham could belong to partial paths SPORT, FOOTBALL and ENTERTAINMENT, MUSIC but might not belong to any leaf categories such as CHAMPIONS LEAGUE or JAZZ.

In our setting the training data $((\mathbf{x}_i, \mathbf{y}(\mathbf{x}_i)))_{i=1}^m$ consists of pairs $(\mathbf{x}, \mathbf{y})$ of vector $\mathbf{x} \in \mathbb{R}^n$ and a multilabel $\mathbf{y} \in \{+1, -1\}^k$ consisting of k microlabels. As the model class we use the exponential family

$$P(\mathbf{y}|\mathbf{x}) \propto \prod_{e \in E} \exp\left(\mathbf{w}_e^T \boldsymbol{\phi}_e(\mathbf{x}, \mathbf{y}_e)\right) = \exp\left(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}, \mathbf{y})\right) \tag{1}$$

defined on the edges of a Markov network $G = (V, E)$, where node $j \in V$ corresponds to the j'th component of the multilabel and the edges $e = (j, j') \in E$ correspond

*Corresponding author.

to the classification hierarchy given as input. By $\mathbf{y}_e = (y_j, y'_j)$ we denote the restriction of the multilabel $\mathbf{y} = (y_1, \ldots, y_k)$ to the edge $e = (j, j')$. The *edge-feature* vector $\boldsymbol{\phi}_e$, in turn, is a concatenation of 'class-sensitive' feature vectors $\boldsymbol{\phi}_e^{\mathbf{u}}(\mathbf{x}, \mathbf{y}_e) = [\![\mathbf{y}_e = \mathbf{u}]\!]\boldsymbol{\phi}(\mathbf{x})$, where $[\![\cdot]\!]$ denotes an indicator function (c.f. [**?**]), and $\mathbf{w}_e$ is the weight vector for edge e. The vector $\boldsymbol{\phi}(\mathbf{x})$ could be a bag of words—as in the experiments reported here—or any other feature representation of the document $\mathbf{x}$. Also, note that although the same feature vector $\boldsymbol{\phi}(\mathbf{x})$ is duplicated for each edge and edge-labeling, in the weight vector $\mathbf{w} = (\mathbf{w}_e^{\mathbf{u}_e})_{e \in E, \mathbf{u}_e}$ we still have a separate weights to represent importance differences of a given feature in different contexts.

There are many ways to define loss functions for hierarchical classification setting (c.f [**?**, **?**, **?**, 1]). *Zero-one loss* $\ell_{0/1}(\mathbf{y}, \mathbf{u}) = [\![\mathbf{y} \neq \mathbf{u}]\!]$ is not very well suited to the task as it ignores the *severity* of the discrepancy between $\mathbf{y}$ and $\mathbf{u}$. *Symmetric difference loss* $\ell_\Delta(\mathbf{y}, \mathbf{u}) = \sum_j [\![y_i \neq u_i]\!]$ does not suffer from this deficiency. However, it fails to take the dependency structure of the microlabels into account. A more appealling choice is the hierarchical loss function of [1]. It penalizes the first mistake along a path, $\ell_{PATH}(\mathbf{y}, \mathbf{u}) = \sum_j c_j [\![y_j \neq u_j \ \ \& \ \ y_k = u_k \forall k \in anc(j)]\!]$, where the coefficients $c_{root} = 1, c_j = c_{pa(j)}/|sibl(j)|$ down-weight mistakes made deeper in the hierarchy. Here we denoted by $anc(j)$ an ancestor, by $pa(j)$ the immediate parent, and by $sibl(j)$ the set of siblings of node j. In this paper, we consider a simplified version of ℓ_{PATH}, namely

$$\ell_{EDGE}(\mathbf{y}, \mathbf{u}) = \sum_j c_j [\![y_j \neq u_j \ \ \& \ \ y_{pa(j)} = u_{pa(j)}]\!],$$

that penalizes a mistake in child if the label of the parent was correct. This choice lets the loss function to capture some of the hierarchical dependencies (between the parent and the child) but allows us define the loss in terms of edges, which is crucial for the efficiency of our learning algorithm. This is achieved by dividing the microlabel loss of each node among the edges adjacent to it.

As in [**?**, **?**], our goal is to learn a weight vector $\mathbf{w}$ that maximizes the minimum margin on training data the between the correct multilabel $\mathbf{y}(\mathbf{x}_i)$ and the incorrect multilabels $\mathbf{y} \neq \mathbf{y}(\mathbf{x}_i)$. Also, we would like the margin to scale as a function of the loss. Alloting a single slack variable for each training example results in the following soft-margin optimization problem:

$$\min_{\mathbf{w}} \frac{1}{2} ||\mathbf{w}||^2 + C \sum_{i=1}^{m} \xi_i \text{ s.t. } \mathbf{w}^T \Delta\boldsymbol{\phi}(\mathbf{x}_i, \mathbf{y}) \geq \boldsymbol{\ell}(\mathbf{y}_i, \mathbf{y}) - \xi_i, \forall i, \mathbf{y} \tag{2}$$

where $\Delta\boldsymbol{\phi}(\mathbf{x}_i, \mathbf{y}) = \boldsymbol{\phi}(\mathbf{x}_i, \mathbf{y}_i) - \boldsymbol{\phi}(\mathbf{x}_i, \mathbf{y})$. This optimization problem suffers from the possible high-dimensionality of the feature vectors. A dual problem [**?**, **?**]

$$\max_{\boldsymbol{\alpha} > 0} \boldsymbol{\alpha}^T \boldsymbol{\ell} - \frac{1}{2} \boldsymbol{\alpha}^T K \boldsymbol{\alpha}, \text{ s.t.} \sum_{\mathbf{y}} \alpha(\mathbf{x}_i, \mathbf{y}) \leq C$$

where $K = \Delta\Phi^T \Delta\Phi$ is the kernel matrix and $\boldsymbol{\ell} = (\ell(\mathbf{x}_i, \mathbf{y}))_{i,\mathbf{y}}$ is the loss vector, allows us to circumvent the problem with feature vectors. However, the number of dual variables $\alpha(\mathbf{x}_i, \mathbf{y})$ is exponential (in number of microlabels) due to the exponential number of constraints in (2), although a polynomial number of dual variables suffices for an approximate solution [**?**].

For the loss functions ℓ_Δ and ℓ_{EDGE} we can use a marginalization trick (c.f. [**?**]) to obtain a polynomial-sized dual optimization problem with dual variables $\mu_e(\mathbf{x}_i, \mathbf{y}_e) = \sum_{\{\mathbf{u} | \mathbf{u}_e = \mathbf{y}_e\}} \alpha(\mathbf{x}_i, \mathbf{u})$ that can be seen as edge-marginals of the original

dual variables. After some arithmetic manipulation the optimization problem takes the form

$$\begin{aligned}
\max_{\boldsymbol{\mu}>0} & \sum_{e\in E}\sum_{i=1}^{m}\sum_{\mathbf{u}_e}\mu_e(x_i,\mathbf{u}_e)\ell_e(x_i,\mathbf{u}_e) \\
& -\frac{1}{2}\sum_{e\in E}\sum_{i,j=1}^{m}\sum_{\mathbf{u}_e,\mathbf{v}_e}\mu_e(x_i,\mathbf{u}_e)K_e(x_i,\mathbf{u}_e;x_j,\mathbf{v}_e)\mu_e(x_j,\mathbf{v}_e) \\
\text{s.t} & \sum_{\mathbf{u}_e}\mu_e(x_i,\mathbf{u}_e)\leq C,\forall x_i,e\in E, \\
& \sum_{\hat{y}}\mu_e(\mathbf{x},y,\hat{y})-\sum_{\hat{y}}\mu_{e'}(\mathbf{x},\hat{y},y)=0,\ \forall e\ \exists e':e'(2)=e(1), \\
& \sum_{\hat{y}}\mu_e(\mathbf{x},\hat{y},y)-\sum_{\hat{y}}\mu_{e''}(\mathbf{x},\hat{y},y)=0,\ \forall e\ \exists e'':e''(2)=e(2). \qquad (3)
\end{aligned}$$

where K_e is an edge-kernel, defined by $K_e(x_i,\mathbf{u}_e;x_j,\mathbf{v}_e)=\Delta\boldsymbol{\phi}_e(x_i,\mathbf{u}_e)^T\Delta\boldsymbol{\phi}_e(x_j,\mathbf{v}_e)$. While this formulation is closely related to that described in [**?**], there are a few differences to be pointed out. Firstly, as we assign the loss to the edges rather than the microlabels, we are able to use richer loss functions than the simple ℓ_Δ. Secondly, single-node marginal dual variables (the μ_i's in [**?**]) become redundant when the box constraints (the first constraint set above) and marginal consistency constraints[1] (the last two constraint sets) are given in terms of the edges. Finally, we utilize the fact that the 'cross-edge' kernel values $\Delta\boldsymbol{\phi}_e(x,\mathbf{y})^T\Delta\boldsymbol{\phi}_{e'}(x',\mathbf{y}')$ are zero when using the feature vectors described above. However, even this considerably smaller optimization problem is a challenging one to solve.

Using our efficient reformulation of the problem (3), we are able to arrive at a more tractable problem. Notice that the box constraints and the marginal consistency constraints are defined for each $\mathbf{x}$ separately, which suggest a possible decomposition. Unfortunately, the objective does not decompose similarly as there certainly exists non-zero kernel values $k(e,x,\mathbf{y}_e,e,x',\mathbf{y}'_e)$ for most $\mathbf{x}\neq\mathbf{x}'$. However, the gradient of the objective $\mathbf{g}=(\boldsymbol{\ell}_e)_{e\in E}-K_E\boldsymbol{\mu}$ can be decomposed by training examples. This gives us the possibility to conduct iterative constrained gradient ascent in subspaces of the examples, where the gradient update for single $\mathbf{x}$ is relatively cheap and the more expensive gradient update only needs to be performed when moving onto the next example. A working set approach, where a chunk of worst predicted training exmaples are added in to the current working set (c.f [**?**, **?**]) also immediately suggests itself. These algorithms can be implemented with relatively low memory footprint by taking into account the special structure of the feature vectors, that repeat $\boldsymbol{\phi}(x)$ many times. We omit details due to lack of space.

We report on an initial experiment with hierarchical classification of Reuter's newswire articles. We used the Reuters Corpus Volume 1, RCV1 [2]. The 2500 documents were used for training and 5000 for testing. Each document in the corpus is associated with one or more topic codes, which are arranged as a forest of 4 trees. For our experiments we used the tree corresponding to the 'CCAT' family of categories, which had a total of 34 nodes We compared the performance of the presented learning approach—below denoted by H-M^3—to three algorithms: SVM denotes an SVM trained for each microlabel separately, H-SVM denotes the case where the SVM for a microlabel is trained only with examples for which the ancestor labels are positive, and H-RLS is the algorithm described in [1]. For training

[1] The notation $\forall e\ \exists e'$ is meant to indicate that to ensure consistency, it suffices to pair up each edge with another, all pairs do not need to be considered

H-M^3 we use both the symmetric difference loss ℓ_Δ and the edge-loss ℓ_{EDGE}. For training SVM and H-SVM, these losses produce the same learned model. The value $C = 1$ was used in all experiments. It should also be noted that, due to time constraints, H-M^3 models were not trained to convergence but until the duality gap was $5 - 10\%$ of the objective value. This took around 4 hours of CPU time on a high-end PC running MATLAB 7.0.

For H-SVM and H-RLS to obtain a reasonable prediction accuracy it was necessary to postprocess the predicted labeling as follows: start at the root and traverse the tree in a breadth-first fashion. If the label of a node is predicted as -1 then all descendants of that node are also labelled negatively, reasoning that if the algorithm cannot make coarse-grained decisions we should not punish it for inability to make fine ones (c.f. [1]). For the other algorithms such postprocessing had only a minor effect so for them the results are shown without this postprocessing.

The results of our comparison are shown in the table below. Flat SVM is expectedly inferior to the competing algorithms, as it cannot utilize the dependencies between the microlabels in any way. In terms of zero-one loss H-M^3-ℓ_Δ performs the best, while H-RLS has the lowest ℓ_Δ and also has slightly lower ℓ_{PATH} loss than the competitors. In this experiment ℓ_{EDGE} and ℓ_{PATH} were equal in all cases. This is mostly due to the shallowness (3 levels) of the hierarchy.

Algorithm	$\ell_{0/1}/m$	ℓ_Δ/m	ℓ_{PATH}	ℓ_{EDGE}
SVM	0.329	0.611	0.099	0.099
H-SVM	0.298	0.570	0.097	0.097
H-RLS	0.281	0.554	0.095	0.095
H-M^3-ℓ_Δ	0.271	0.627	0.144	0.144
H-M^3-ℓ_{EDGE}	0.294	0.643	0.102	0.102

Based on these initial experiments, we conclude the presented H-M^3 method seems promising, although the prediction accuracies do not differ very much between the algorithms.

We plan to extend the experimental studies by the time of the workshop, taking on board other document hierarchicies. Also, the good choice of loss function is far from clear from these experiments alone. For example, scaling of the ℓ_{PATH} and ℓ_{EDGE} losses can be done in several ways only one of which was examined here.

References

[1] Y. Altun and T. Hofmann I. Tsochantaridis. Hidden markov support vector machines. In *ICML'03*, pages 3–10.

[2] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *13 ACM CIKM*, 2004.

[3] N. Cesa-Bianchi, C. Gentile, A. Tironi, and L. Zaniboni. Incremental algorithms for hierarchical classification. In *Neural Information Processing Systems*, 2004.

[4] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML'04*, pages 209–216.

[5] T. Hofmann, L. Cai., and M. Ciaramita. Learning with taxonomies: Classifying documents and words. In *NIPS Workshop on Syntax, Semantics, and Statistics*, 2003.

[6] David D. Lewis, Y. Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397, Apr 2004.

[7] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Neural Information Processing Systems*, 2003.

[8] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML'04*, pages 823–830.