Regularized Structured Output Learning with Partial Labels

Sundararajan Sellamanickam[*] Charu Tiwari[†] Sathiya Keerthi Selvaraj[‡]

Abstract

We consider the problem of learning structured output probabilistic models with training examples having partial labels. Partial label scenarios arise commonly in web applications such as taxonomy (hierarchical) classification, multi-label classification and information extraction from web pages. For example, label information may be available only at the internal node level (*not at the leaf level*) for some pages in a taxonomy classification problem. In a multi-label classification problem, it may be available only for some of the classes (in each example). Similarly, in a sequence learning problem, we may have label information only for some nodes in the training sequences. Conventionally, marginal likelihood maximization technique has been used to solve these problems. In such a solution unlabeled examples and any side information like expected label distribution (or correlation in a multi-label setting) of the unlabeled part are not used. We solve these problems by incorporating entropy and label distribution or correlation regularizations along with marginal likelihood. Entropy and label distribution regularizations have been used previously in semi-supervised learning with *fully* unlabeled examples. In this paper we develop probabilistic taxonomy and multi-label classifier models, and provide the ideas needed for expanding their usage to the *partial* label scenario. Experiments on real-life taxonomy and multi-label learning problems show that significant improvements in accuracy are achieved by incorporating these regularizations, when most of the examples are only partially labeled.

1 Introduction

Supervised learning algorithms require fully labeled training examples to learn classifiers. In many structured output applications such as taxonomy classification, multi-label classification, information extraction or sequence labeling in bio-informatics, getting labeled examples is expensive. Thus, semi-supervised learning algorithms ([18],[13]) that make use of a few labeled examples together with a large number of unlabeled examples have become popular. However, all of these semi-supervised learning algorithms assume that label information for each labeled example is *fully* available. We consider the semi-supervised learning problem with *weak* supervision in which only partial label information is available for the examples. We illustrate the notion of partial label using some motivating applications below.

1.1 Motivating Applications In hierarchical (taxonomy) classification of web pages, we may have label information only at the internal node level (*not at the leaf node level*) for many pages. For example, we may only know that a particular web page belongs to SCIENCE category but not whether the page belongs to the *leaf nodes* PHYSICS or CHEMISTRY. In practice, such a situation can arise from the requirement to expand the taxonomy by splitting leaf nodes further. Then, to reduce annotation cost, we may label only a few pages from each such leaf nodes.

In a multi-label classification problem, a web page or image is associated with multiple categories represented by a multi-label output vector (with each bit taking a value in $\{+1,-1\}$). For example, a particular web page may belong to FINANCE and POLITICS categories. Partial labeling in multi-label output means that labels are known only for a subset of categories in the multi-label output vector. Following is a practical scenario in which such partial data arise. Suppose there are multiple annotators who label the examples, and each annotator judges the label only for a subset of categories (non-overlapping with other annotators); such a way of labelling may become a necessity when the number of categories is large. Furthermore, the set of examples given to each annotator may overlap. Clearly, when an example overlaps with all the annotators (thus, covering all the categories) we have a fully labeled example; otherwise, we have a partially labeled example. One advantage of having non-overlapping examples across the annotators is that we can get more number of labeled examples (though partial) for the same annotation cost[1].

In a web information extraction application, a record (a sequence of nodes) from a list page has multiple attributes. For example, a BOOK record will have attributes like Name, Author, Publisher etc. Similarly, a RESTAURANT record will have Name, Address,

[*]Yahoo! Labs, Bangalore, India. ssrajan@yahoo-inc.com
[†]Yahoo! Labs, Bangalore, India. charu@yahoo-inc.com
[‡]Yahoo! Labs, Santa Clara, CA. selvarak@yahoo-inc.com

[1]We assume that the annotator's cost is directly proportional to the total number of class labels he/she assigns for the objects (e.g., documents or images).

Phone etc, as the attributes. Now assume that we have some *incomplete* databases of RESTAURANT and BOOK names. Then, we can use these databases to annotate the attribute Name in extracted records from list pages[2]. Similarly, we may use PERSONNAME, PUBLISHERNAME dictionaries and *address*, *phone* regular expressions to annotate other attributes like Author, Publisher, Address and Phone. In such a situation, the records will only have *partial* information since the databases and dictionaries are often *incomplete.* But, the quality of annotation (*precision*) will be very high. It is clear from these applications that there is a need to design learning algorithms that can learn with partial labels.

1.2 Use of side information and Regularizations in Semi-supervised Learning Side information or constraints (on the labels of unlabeled data) that come from domain knowledge play a key role in getting significantly improved performance in semi-supervised learning. Joachims [13] uses label distribution of the unlabeled data as a hard constraint, to avoid the trivial (bad) solution of labeling the entire unlabeled data with same label. In [16], a soft constraint (regularization) to match the model predicted label distribution with a prior label distribution is used. Entropy regularization is another form of regularization that prefers a solution providing predicted label distribution with less entropy (see [9],[16],[11]). In [16], experiments conducted on multi-class problems show that label regularization is crucial to get good performance; entropy regularization can further enhance the performance when included with label regularization, but it gives poor performance when used alone. While all the above mentioned works deal with semi-supervised learning where the unlabeled examples are *fully* unlabeled, our focus is on the problem of learning with partial labels that often arise in structured output problems.

1.3 Our Contributions The main contributions of this paper are:

- we provide the ideas needed for expanding the usage of entropy and side information based label regularizations to the *partial label* scenario for structured output problems.
- we propose a probabilistic taxonomy classifier model using Cai and Hoffmann [7] attribute vector taxonomy representation (where each class is characterized by an attribute vector). In this representation, we introduce the notion of conditional class distribution that is useful in solving the problem of learning with partial labels.
- we introduce a probabilistic multi-label classifier model based on the linear (label) correlation model proposed in [10], and show how different regularizations can be used in the partial label scenario.
- we conduct extensive experiments on real-life taxonomy and multi-label learning problems and demonstrate that significant improvements in accuracy are achieved by incorporating entropy and label (distribution or correlation) regularizations, when most of the examples are only partially labeled.

The paper is organized as follows. In Section 2, we present related work. Section 3 covers background on likelihood and marginal likelihood maximization approaches. In Section 4, we present our approach of using entropy and label regularizations in the partial label scenario; we take up two structured output problems, viz., taxonomy (hierarchical) and multi-label classification, and develop necessary probabilistic classifier models for these problems, and show how various regularizations can be incorporated. Experimental results on real-life benchmark datasets of these two problems are presented in Section 5. We briefly discuss in Section 6, how the proposed approach can be used in a more complex sequence learning problem. Section 7 concludes the paper.

2 Related Work

As annotation is an expensive process due to the human effort involved, significant attention has been given to the problem of learning with limited or no labeled examples by researchers from various communities viz., artificial intelligence, machine learning, statistics and data mining. Semi-supervised learning [18] refers to the problem of learning when both labeled and unlabeled examples are available. In practice, depending on the problem and application at hand, different *degree of labeling* (that is, *fully* and *partial*) arises. For instance, in a binary or multi-class problem, an example is *fully* labeled when the label is exactly known; in a multi-label classification problem, an example is *fully* labeled when labels of all the outputs (classes) are known. Our interest in this paper is in learning with *partial* labels. Next we discuss works related to such learning.

Grandvalet and Bengio [9], and Jin and Ghahramani [12] consider the partial labeling problem in the multi-class probabilistic modeling setting, where each training input is associated with a subset of class labels. It is assumed that the *true* label is a member of this set. Grandvalet and Bengio [9] propose a min-

[2]Any standard unsupervised technique such as [1] can be used to extract records from list pages.

imum entropy regularization based solution; however, there are no detailed experimental studies specific to the partial labeling problem. Jin and Ghahramani [12] solve the problem using the expectation-maximization (EM) approach. Nguyen and Caruana [17] propose a large margin approach for the same problem setting. Our work differs from the above mentioned works in two ways. First, our work is focused on structured output problems. Second, none of these works make use of label regularization which is crucial in getting significant improvement in performance - particularly, when the degree of labeling is low (for example, when the percentage examples for which labeling information is available at the leaf node level in the taxonomy classification problem is, say, less than 5%).

Liu et al. [6] proposed a semi-supervised learning method for the multi-label classification problem where the class label assignments for the fully unlabeled data are set such that the Frobenius norm between input similarity matrix and class similarity matrix[3] is minimized. Chen et al. [3] presented a graph regularized semi-supervised multi-label learning method that make use of input similarity matrix and category similarity matrix that captures the correlation information between the labels. In their method, the labels for the fully unlabeled data were obtained by solving a Sylvester equation. However, these methods do not solve the problem of learning with partial labels.

Mann and McCallum [16] devised a method called *generalized expectation criteria (GE)* to make use of prior or side information in semi-supervised learning. The GE criteria method essentially fits model parameters $\mathbf{W}$ in such a way that the model's predictions match certain expectation constraints. An important type of expectation constraint is to match the model predicted marginal label distribution on the unlabeled data with the expected label distribution. Such constraints play a very important role in semi-supervised learning to get improved generalization performance; this has also been demonstrated in applications like text classification [13]. Mann and McCallum [16] explored the above mentioned label distribution constraint by adding a *label regularization* function to the objective function. They demonstrate its usefulness on both multi-class and sequence labeling problems. Jiao et al., [11] propose to use entropy regularization for structured prediction problems. However, all these works consider the problem in the *fully* labeled setting whereas our focus and approach is for the *partially* labeled setting.

Kedar and McCallum [2] propose marginal likelihood maximization for the sequence learning problem with partially labeled nodes. Carlson et al., [8] propose partial perceptron learning for the same problem. Neither use entropy and label regularizations as proposed in this paper. Our experimental results show that these regularizations are extremely useful in getting significant performance improvements.

3 Background

In this work, we are interested in learning discriminative probabilistic models specified by the class conditional probability distribution function $p(\mathbf{y}|\mathbf{x};\mathbf{W})$, where $\mathbf{x}$ and $\mathbf{y}$ respectively denote the input and output representations of an example, and $\mathbf{W}$ denotes the model parameter vector.

Suppose we are given a set of *fully* labeled examples $\{\mathbf{x}_i, \mathbf{y}_i : i = 1, \ldots, n\}$ which are independently chosen from the underlying distribution. The model parameter $\mathbf{W}$ is learned by maximizing the regularized log likelihood function given by:

$$(3.1)\quad \mathcal{L}(\mathbf{W}) = -\frac{1}{2\sigma^2}||\mathbf{W}||^2 + \frac{\lambda_{ML}}{n}\sum_{i=1}^{n}\log p(\mathbf{y}_i|\mathbf{x}_i;\mathbf{W})$$

where the first term is a weight regularization term and λ_{ML} is a regularization constant that trades-off data fitting with function smoothness for a fixed σ^2. In structured output problems, $\mathbf{y}_i$ involves *multiple nodes* and each node (j) is assigned a label $y_{i,j} \in \mathcal{Y}$ where $\mathcal{Y}$ is the output label space.

3.1 Marginal Likelihood In the partial labeling scenario, labels are known only for a subset of nodes; that is, $\mathbf{y}_i^{(o)} \in \mathbf{y}_i = [\mathbf{y}_i^{(o)}\mathbf{y}_i^{(u)}]$ where the superscripts o and u respectively denote *observed* and *unobserved* nature of the node labels. When the examples are *partially* labeled, the likelihood $p(\mathbf{y}_i|\mathbf{x}_i;\mathbf{W})$ is *replaced* with the marginal likelihood function $p(\mathbf{y}_i^{(o)}|\mathbf{x}_i;\mathbf{W}) = \sum_{\mathbf{y}_i^{(u)}} p(\mathbf{y}_i|\mathbf{x}_i;\mathbf{W})$ in equation (3.1) [2]. This results in:
(3.2)

$$\mathcal{L}_{ML}(\mathbf{W}) = -\frac{1}{2\sigma^2}||\mathbf{W}||^2 + \frac{\lambda_{ML}}{n}\sum_{i=1}^{n}\log(p(\mathbf{y}_i^{(o)}|\mathbf{x}_i;\mathbf{W})).$$

Unlike full likelihood, marginal likelihood is not necessarily convex; hence $\mathcal{L}_{ML}(\mathbf{W})$ in equation (3.2) may not be convex. This results in the problem of *local* minima.

As pointed out by Grandvalet and Bengio [9], semi-supervised learning is a special case of learning with partial labels. For example, when $\mathbf{y}_i^{(o)} = \phi$ (that is, no labels are observed) we have the *fully unlabeled* scenario

[3]For example, the input similarity matrix is computed using cosine similarity measure between the term frequency vectors documents and the class similarity matrix is computed as the inner product of the confidence scores of the output (class) vector between the documents in a transformed space that captures the label correlation.

for i-th sequence. Then, it is seen that such an example does not contribute to the likelihood term in equation (3.2) with the marginal probability summing to 1 (since $\mathbf{y}_i^{(u)} = \mathbf{y}_i$). When $\mathbf{y}_i^{(u)} = \phi$ (that is, all the labels are observed), we have the *fully labeled* scenario. Then, the summation within the log term involves only one term, resulting in the standard likelihood function.

4 Our Approach

We are interested in getting improved generalization performance while learning with partial labels. For this purpose, we enhance equation (3.2) by providing additional regularizations. In particular, we propose to extend and use two types of regularizations commonly known as *entropy* and *label* regularizations (see [9] and [16]). The key idea in our work is to use these regularizations along with marginal likelihood for structured output learning with *partial* labels, when partial labeling is available for some or all of the training examples. As we will see, our experimental results show that significant improvement in generalization performance is achieved in the partial labeled scenario with these regularizations. Although a big chunk of the improvement comes from using only label regularization, further improvement is achieved when *label* regularization is combined with *entropy* regularization.

4.1 Entropy Regularization We define the *entropy regularization* (ER) function $\mathcal{H}_i(\mathbf{W})$ for the partial labels scenario as:

$$(4.3) \quad -\sum_{\mathbf{y}_i^{(u)}} p(\mathbf{y}_i^{(u)}|\mathbf{y}_i^{(o)}, \mathbf{x}_i; \mathbf{W}) \log p(\mathbf{y}_i^{(u)}|\mathbf{y}_i^{(o)}, \mathbf{x}_i; \mathbf{W})$$

where $p(\mathbf{y}_i^{(u)}|\mathbf{y}_i^{(o)}, \mathbf{x}_i; \mathbf{W})$ is the class probability distribution of the unobserved variable $\mathbf{y}_i^{(u)}$ *conditional* on the observed variable $\mathbf{y}_i^{(o)}$. This function is a generalized version of the function used in the *fully unlabeled* example scenario [16] where $\mathbf{y}_i^{(u)} = \mathbf{y}_i$; so, the summation in equation (4.3) runs over all possible assignments of $\mathbf{y}_i$ (i.e. $\mathbf{y}_i^{(o)} = \phi$ and equation (4.3) becomes $-\sum_{\mathbf{y}_i} p(\mathbf{y}_i|\mathbf{x}_i; \mathbf{W}) \log p(\mathbf{y}_i|\mathbf{x}_i; \mathbf{W})$). When we add the entropy regularization function to equation (3.2), we get:

$$(4.4) \quad \mathcal{L}_{ML-ER}(\mathbf{W}) = \mathcal{L}_{ML}(\mathbf{W}) - \frac{\lambda_{ER}}{n} \sum_{i=1}^{n} \mathcal{H}_i(\mathbf{W})$$

where λ_{ER} is an entropy regularization constant which controls the amount of regularization. Note that maximizing entropy regularized marginal likelihood function (equation (4.4)) tries to decrease the conditional entropy function $\sum_{i=1}^{n} \mathcal{H}_i(\mathbf{W})$. Of course, this happens jointly with maximizing the model prediction probability on the observed part (i.e., the first term). Essentially, decreasing the conditional entropy function drives the prediction on the unlabeled part of the structure to be more confident.

4.2 Label Regularization Label regularization is a simple and powerful idea that has been found to be very useful in semi-supervised learning with fully labeled and fully unlabeled examples [16]. Here, we propose to use *label regularization* in the partial labeled scenario. We add a label regularization term $\mathcal{L}_{LR}(\mathbf{W})$ to equation (4.4) and get:

$$(4.5) \quad \mathcal{L}_{ML-LR-ER}(\mathbf{W}) = \mathcal{L}_{ML-ER}(\mathbf{W}) - \mathcal{L}_{LR}(\mathbf{W}).$$

The label regularization term can be defined in different ways depending on the side information available. For illustrative purpose, we present a simple label distribution based regularization function here; later, we show how label correlation information in a multi-label classification setting can be used to define the regularization function.

4.2.1 Label Distribution Regularization (LDR) Let $\mathbf{q}$ denote the expected label distribution in the partial labeled scenario where the distribution is defined over the label space $\mathcal{Y}$. Let $\mathbf{p}(\mathbf{W})$ denote the model predicted marginal label distribution[4] in the unobserved part of the structured outputs; that is, $\mathbf{p} = \frac{1}{|U|}\sum_{i\in U} \mathbf{p}_i$ where U and $|U|$ respectively denote the set and size of the unobserved nodes, and $\mathbf{p}_i$ denotes the model predicted marginal probability for i-th node. For notational simplicity, we drop the dependence on $\mathbf{W}$. Then, we define[5]:

$$(4.6) \quad \mathcal{L}_{LR}(\mathbf{W}) = \mathcal{L}_{LDR}(\mathbf{W}) = \lambda_{LDR}\mathcal{G}(\mathbf{q}; \mathbf{p})$$

where λ_{LDR} is a regularization constant, and $\mathcal{G}(\mathbf{q}; \mathbf{p})$ measures the difference between the expected and predicted distributions. For instance, $\mathcal{G}(\cdot)$ may be a Kullback-Leibler divergence or squared error loss measure; e.g., $\mathcal{G}(\mathbf{q}, \mathbf{p}) = ||\mathbf{q} - \mathbf{p}||^2$. Note that when $\lambda_{ER} = 0$ in equation (4.5) (via equation (4.4)), we refer the objective function as $\mathcal{L}_{ML-LDR}$ (with label distribution regularization).

[4]Note that if we have expected label distribution information for the entire training data then the expected label distribution in the unlabeled part is specified as conditional distribution.

[5]We use the abbreviation LR to refer to *general* label regularization setting. Whenever we use the abbreviation LDR, we specifically refer to label distribution regularization. We have introduced the function $\mathcal{L}_{LDR}(\mathbf{W})$ to distinguish from other types of label regularizations.

Illustrative Examples In the taxonomy classification problem, we may specify the expected label distribution over the classes represented by the leaf nodes. For instance, this distribution may be just uniform; this is the case in one of the datasets News20 used in our experiment explained later. Alternately, one could also define constraints at the internal node levels, if only such information is available. For example, we may just specify the expected fraction of documents in some of the internal nodes. Thus, in a general setting, $\mathbf{p}$ *need not be* a distribution.

In the multi-label classification problem, the expected fraction of documents may be specified for each output (class) *independently*. In this case, the label regularization term can be written as the sum of the label regularization term for each output: $\sum_{k=1}^{K} \mathcal{G}(\mathbf{q}_k; \mathbf{p}_k)$ where K denotes the number of outputs, $\mathbf{q}_k$ and $\mathbf{p}_k$ denote the expected and predicted label distribution respectively for k^{th} output in the label space $\{-1, +1\}$.

In the sequence learning problem, one can define label constraints in several ways. For example, the expected label distribution can be defined over the label space $\mathcal{Y}$ for the entire collection of n sequences. As an illustration, we may have information, such as the average number of label type Author is 2 per publication sequence. In this case, we expect $2n$ nodes to be labeled as Author and if there are total M nodes then $p(\text{Author}) = \frac{2n}{M}$. Alternately, we may also impose constraint for *each* partially labeled sequence. In this case, $\mathcal{G}(\mathbf{q}; \mathbf{p})$ in (4.5) would be $\sum_{i=1}^{n} g(\mathbf{q}_i; \mathbf{p}_i)$ where $g(\cdot)$ is again a divergence measure and i indexes the sequences.

4.3 Taxonomy Classifier Model

Consider a hierarchical tree structure with root node R and each leaf node representing a class. Following [7], we associate a weight vector $\mathbf{w}_r$, $r = 1, \ldots, s$ where s denotes the number of nodes in the tree (with a specific order to index each node), and use an attribute vector representation model for each leaf node $v \in \mathbf{L}$ ($\mathbf{L}$ denotes the set of leaf nodes). Let $\Lambda(\mathbf{y}_v) = [\lambda_1(\mathbf{y}_v) \cdots \lambda_s(\mathbf{y}_v)]$ denote the attribute vector representation for v^{th} node where $\mathbf{y}_v$ denotes the path from R, and λ_rs take binary value (1 if a node lies in the path and 0 otherwise). We define a scoring function[6] for v^{th} (leaf) node as:

$$F(\mathbf{x}, \mathbf{y}_v; \mathbf{W}) = \sum_{r=1}^{s} \lambda_r(\mathbf{y}_v) \mathbf{w}_r^T \mathbf{x}.$$

Then, inference is made as:

$$\mathbf{y}^* = \underset{\mathbf{y}_v : v \in \mathbf{L}}{\text{argmax}}\, F(\mathbf{x}, \mathbf{y}_v; \mathbf{W}).$$

While [7] studies the support vector machine model, we are interested in building a probabilistic model. Therefore, we work with an extended multinomial logistic regression model, and define the class probability of a leaf node $v \in \mathbf{L}$ as:

$$p(\mathbf{y}_v|\mathbf{x}; \mathbf{W}) = \frac{\exp(F(\mathbf{x}, \mathbf{y}_v; \mathbf{W}))}{\sum_{\bar{v} \in \mathbf{L}} \exp(F(\mathbf{x}, \mathbf{y}_{\bar{v}}; \mathbf{W}))}. \tag{4.7}$$

We see that the conditional probability of a document belonging to a child node (v) given that it belongs to a parent node r is:

$$p(\mathbf{y}_v|\mathbf{y}_r, \mathbf{x}; \mathbf{W}) = \frac{p(\mathbf{y}_r|\mathbf{y}_v, \mathbf{x}; \mathbf{W}) p(\mathbf{y}_v|\mathbf{x}; \mathbf{W})}{p(\mathbf{y}_r|\mathbf{x}; \mathbf{W})}. \tag{4.8}$$

From the observations that a child has only one parent and a parent has more than one child[7], the following properties hold:

1. $p(\mathbf{y}_r|\mathbf{y}_v, \mathbf{x}; \mathbf{W}) = 1$
2. Probability of a document belonging to a parent (i.e., internal node) is nothing but the sum of the class probabilities of its children given by:

   $$p(\mathbf{y}_r|\mathbf{x}; \mathbf{W}) = \sum_{\bar{v} \in child(r)} p(\mathbf{y}_{\bar{v}}|\mathbf{x}; \mathbf{W}). \tag{4.9}$$

   As we propagate the probabilities upward to the root, the probability at the root node R becomes 1.

Using these properties we can rewrite equation (4.8) as:

$$p(\mathbf{y}_v|\mathbf{y}_r, \mathbf{x}; \mathbf{W}) = \frac{\exp(F(\mathbf{x}, \mathbf{y}_v; \mathbf{W}))}{\sum_{\bar{v} \in child(r)} \exp(F(\mathbf{x}, \mathbf{y}_{\bar{v}}; \mathbf{W}))}. \tag{4.10}$$

Furthermore, the conditional probability is not dependent on the weight vectors shared by the children (as the common terms in the numerator and denominator cancels out).

4.3.1 Likelihood and Regularization

We use equation (4.7) in the likelihood term (equation (4.4)) when leaf node label is available for an example. When an example has only partial label (i.e., label information only at an internal node level), we use equations (4.9) and (4.10) in the (marginal) likelihood and

[6]In document categorization problems linear kernel is often sufficient.

[7]A parent with single child can be collapsed to a single child node.

entropy regularization terms of equation (4.4) respectively. To compute the label distribution regularization (LDR) term in equation (4.6), we use $p(\mathbf{y}_v|\mathbf{x};\mathbf{W}) = \frac{\exp(\frac{1}{T}F(\mathbf{x},\mathbf{y}_v;\mathbf{W}))}{\sum_{\bar{v}\in\mathbf{L}}\exp(\frac{1}{T}F(\mathbf{x},\mathbf{y}_{\bar{v}};\mathbf{W}))}$. The scaling factor $(\frac{1}{T})$ is useful in preventing the degenerate solution of all the examples having same probability score [16]. In all our experiments we used squared error loss (between the expected and model predicted label distributions) for $\mathcal{G}(\cdot)$ in equation (4.5), and set $T = 0.1$. Note that the gradients of the marginal probability and entropy regularization term are straight-forward to compute for this problem.

4.4 Multi-Label Model In this section we show how entropy and side information based label regularizations can be incorporated in a partial label, multi-label classification setting. Before that, we first describe the multi-label setting that we consider here; we make use of the approach suggested in [10] to define our probabilistic multi-label classifier model.

Let $f(\mathbf{x},\mathbf{y})$ denote the scoring function for a multi-label vector assignment $\mathbf{y}$ defined as:

$$f(\mathbf{x},\mathbf{y};\mathbf{w}) = \mathbf{w}^T(\mathbf{x}\otimes\psi(\mathbf{y}))$$

where $\otimes$ denotes the Kronecker product, $\mathbf{w}$ and $\psi(\mathbf{y})$ denote the model weight vector, and attribute vector (for $\mathbf{y}$) respectively. Then, inference is made as:

$$\mathbf{y}^* = \underset{\mathbf{y}\in\{-1,+1\}^K}{\operatorname{argmax}} f(\mathbf{x},\mathbf{y})$$

where K denotes the number of classes (i.e., dimension of the multi-label output). Note that inference becomes intractable as the label space is exponentially large. While treating each class as independent can make the problem simple, it does not make use of vital dependency information across the classes. Hariharan et al. [10] considered a problem setup where the output variables are linearly correlated; i.e., they assumed that $\psi(\mathbf{y}) = \mathbf{P}\mathbf{y}$ where $\mathbf{P}$ is an invertible matrix that captures all the *label correlation* information. With this assumption they proposed an approach where the labels are densely correlated, but does not incorporate pairwise label terms in the prediction function. They showed that (1) the complexity (prediction as well as training in terms of the number of constraints) can still be reduced from exponential to linear while modeling dense pairwise correlations, and (2) incorporating correlation priors can result in improved prediction accuracy. While in [10], Hariharan et al. worked in a large margin setting with support vector machines, we are interested in probabilistic models. Therefore, we propose a probabilistic multi-label classifier model with their *linear (label) correlation* assumption. It is easy to show that the *model decoupling* simplification (from prediction and likelihood perspectives) obtained in their formulation holds in probabilistic formulation as well.

4.4.1 Likelihood The joint probability of the multi-label output $\mathbf{y}$ is given by:

$$p(\mathbf{y}|\mathbf{x};\mathbf{W},\mathbf{P}) = \frac{\exp(f(\mathbf{x},\mathbf{y}))}{\sum_{\mathbf{y}}\exp(f(\mathbf{x},\mathbf{y}))}$$

where $\mathbf{W} = [\mathbf{w}_1\mathbf{w}_2\ldots\mathbf{w}_K]$ and $\mathbf{w}_k$ denotes k-th classifier weight vector. Note that the model weight (column) vector ($\mathbf{w}$) is rearranged as a matrix of weight vectors and we have omitted the dependencies of $\mathbf{P}$ and $\mathbf{W}$ in the scoring function. Denoting $\mathbf{Z} = \mathbf{W}\mathbf{P}$ and $\mathbf{z}(\mathbf{x}) = \mathbf{Z}^T\mathbf{x}$, we rewrite $p(\mathbf{y}|\mathbf{x};\mathbf{W},\mathbf{P})$ as:

$$p(\mathbf{y}|\mathbf{x};\mathbf{Z}) = \frac{\exp(\mathbf{y}^T\mathbf{z}(\mathbf{x}))}{\sum_{\mathbf{y}}\exp(\mathbf{y}^T\mathbf{z}(\mathbf{x}))}.$$

Due to additive nature of $\mathbf{y}^T\mathbf{z}(\mathbf{x})$ and exponentiation property, the sum in the denominator factors into a product of K sums; therefore, $p(\mathbf{y}|\mathbf{x};\mathbf{Z})$ factors into K independent distributions, i.e.,

$$p(\mathbf{y}|\mathbf{x};\mathbf{Z}) = \prod_{k=1}^{K} p(y_k|\mathbf{x},\mathbf{z}_k)$$

where $p(y_k|\mathbf{x},\mathbf{z}_k) = \frac{\exp(y_k z_k(\mathbf{x}))}{\sum_{y_k\in\{-1,+1\}}\exp(y_k z_k(\mathbf{x}))}$, $z_k(\mathbf{x}) = \mathbf{z}_k^T\mathbf{x}$ and $\mathbf{z}_k$ is k-th column in $\mathbf{Z}$. Using this likelihood model, we write the basic weight regularized log likelihood function (equation (3.1) in the transformed space $\mathbf{Z}$) as:

(4.11)

$$\mathcal{L}(\mathbf{Z}) = -\frac{1}{2\sigma^2}trace(\mathbf{Z}\mathbf{R}^{-1}\mathbf{Z}^T) + \frac{\lambda_{ML}}{n}\sum_{i=1}^{n}\log p(\mathbf{y}_i|\mathbf{x}_i;\mathbf{Z})$$

where the first term is obtained from the regularization term:

$$\sum_{k=1}^{K}||\mathbf{w}_k||^2 = trace(\mathbf{W}^T\mathbf{W}) = trace(\mathbf{W}\mathbf{W}^T)$$

and $\mathbf{R} = \mathbf{P}^T\mathbf{P}$. Note that the transformed weight matrix $\mathbf{Z}$ can only be learned *jointly* due to the coupling through $\mathbf{R}^{-1}$. Therefore, although the joint probability distribution factors into independent distributions, interdependencies among the transformed weight vectors $\{\mathbf{z}_k : k = 1,\ldots,K\}$ imply *joint* learning of K classifiers. But, the key benefit due to factorization comes during testing where prediction can be done independently for each class k using $z_k(\mathbf{x})$ (i.e., $y_{i,k} = 1$ when $z_k(\mathbf{x}_i) \geq 0$ and -1 otherwise; here, $y_{i,k}$ denotes the label of i^{th} example for k^{th} class). Furthermore, factorization results in simpler log marginal likelihood and entropy regularization terms as explained next.

4.4.2 Marginal Likelihood

Recall that we compute the marginal likelihood for i-th example in the partial label scenario as $p(\mathbf{y}_i^{(o)}|\mathbf{x}_i;\mathbf{Z}) = \sum_{\mathbf{y}_i^{(u)}} p(\mathbf{y}_i|\mathbf{x}_i;\mathbf{Z})$ (after replacing $\mathbf{W}$ with $\mathbf{Z}$). Since the joint distribution factors into K independent distributions, the marginal likelihood function for i^{th} example simplifies to

$$p(\mathbf{y}_i^{(o)}|\mathbf{x}_i;\mathbf{Z}) = \prod_{k\in\mathbf{L}_i} p(y_{i,k}|\mathbf{x}_i,\mathbf{z}_k) \tag{4.12}$$

where $\mathbf{L}_i$ denotes the subset of classes in $\mathcal{K} = \{1,\ldots,K\}$ for which the labels are known. Then, we get the expression for $\mathcal{L}_{ML}(\mathbf{Z})$ (by substituting equation (4.12) in equation (4.11)) as

$$-\frac{1}{2\sigma^2}trace(\mathbf{Z}\mathbf{R}^{-1}\mathbf{Z}^T) + \frac{\lambda_{ML}}{n}\sum_{i=1}^{n}\sum_{k\in\mathbf{L}_i}\log p(y_{i,k}|\mathbf{x}_i,\mathbf{z}_k). \tag{4.13}$$

4.4.3 Entropy Regularization

Using the factorization property above and denoting $U_i = \mathcal{K}\setminus L_i$, the conditional entropy function (4.3) reduces to:

$$\mathcal{H}_i(\mathbf{Z}) = -\sum_{k\in U_i} p(y_{i,k}|\mathbf{x}_i,\mathbf{z}_k)\log p(y_{i,k}|\mathbf{x}_i,\mathbf{z}_k)$$

and (4.4) becomes $\mathcal{L}_{ML-ER}(\mathbf{Z}) = \mathcal{L}_{ML}(\mathbf{Z}) - \frac{\lambda_{ER}}{n}\sum_{i=1}^{n}\mathcal{H}_i(\mathbf{Z})$.

4.4.4 Label Regularizations

We consider two types of label regularizations, namely, label distribution regularization (LDR) and label correlation regularization (LCR).

In LDR, we assume that the expected label (binary) distribution for each output (i.e., $\{\mathbf{q}_k : k = 1,\ldots,K\}$) is available, and use the label distribution regularization function mentioned earlier: $\mathcal{L}_{LDR}(\mathbf{Z}) = \frac{\lambda_{LDR}}{K}\sum_{k=1}^{K}\mathcal{G}(\mathbf{q}_k;\mathbf{p}_k)$ where λ_{LDR} and $\{\mathbf{p}_k : k = 1,\ldots,K\}$ denote the regularization constant and predicted label (binary) distributions respectively. In all our experiments, we used squared error loss for $\mathcal{G}(\cdot)$.

In LCR, we use the label correlation matrix $\mathbf{R}$ to regularize the classifier weights such that the model predictions of the labels (including the hidden labels) result in the desired $\mathbf{R}$. Thus, we define the LCR regularization function as: $\mathcal{L}_{LCR}(\mathbf{Z}) = \frac{\lambda_{LCR}}{K^2}||\hat{\mathbf{R}} - \mathbf{R}||_F^2$ where $||.||_F^2$ and λ_{LCR} denote the Frobenius norm and regularization constant respectively, $\hat{\mathbf{R}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{u}_i\mathbf{u}_i^T$, $\mathbf{u}_i = 2 * \mathbf{p}_i - \mathbf{1}$ and $\mathbf{p}_i$ is a column vector $(K \times 1)$ representing K classifier output probabilities. Note that when $p_{i,k} = p(y_{i,k}|\mathbf{x}_i,\mathbf{z}_k) = 1$ we have $u_{i,k} = 1$, and, when $p_{i,k} = 0$ we have $u_{i,k} = -1$. Therefore, both positive and negative label correlations are possible.

Finally, combining marginal likelihood, entropy and label regularizations, we get

$$\mathcal{L}_{ML-LR-ER}(\mathbf{Z}) = \mathcal{L}_{ML-ER}(\mathbf{Z}) - \mathcal{L}_{LDR}(\mathbf{Z}) - \mathcal{L}_{LCR}(\mathbf{Z}).$$

In our experiments, we evaluated the usefulness of each of the regularization terms separately as well as in combination with others.

5 Experimental Evaluation

To demonstrate the effectiveness of the proposed methods we conducted experiments on two structured output problems viz., taxonomy and multi-label classification problems using the models proposed in sections 4.3 and 4.4 respectively. We compared the performance with the marginal likelihood method. We give details of the experimental setup and results for taxonomy and multi-label classification experiments in separate sections.

5.1 Taxonomy Classification Experiments

5.1.1 Datasets

We evaluated the performance of the various methods on two popular web taxonomy datasets viz., News20 and RCV-MCAT.
The News20 dataset[8] consists of 19928 examples belonging to 20 *leaf* level categories. The taxonomy tree has a depth of 3 and has 28 nodes including the *root*. There are six nodes (belonging to categories COMPUTERS, RECREATION, SCIENCE, RELIGION, POLITICS and MISCELLANEOUS) at the first level and one of them (MISCELLANEOUS) is a leaf node. There are seventeen nodes at the second level and fifteen of them are leaf nodes. The remaining four leaf nodes are at the third level. The number of features is 62061. The examples are uniformly distributed across the classes. We used a train/test split of 65% and 35%.
The RCV[9] dataset (with single label) consists of 547655 examples belonging to 101 classes. For our experiments we considered a sub-tree belonging to a high level category MCAT with seven leaf nodes. This tree has a depth of 2 with a total of 10 nodes. There are four nodes in the first level and five nodes in the second level. The seven market related categories (leaf level nodes) are EQUITY, BOND, FOREX, COMMODITY, SOFT, METAL and ENERGY. The RCV-MCAT dataset has 154706 examples and the number of features is 11429. Since this dataset is relatively easy, we used a train and test split of 3% and 97% respectively.

5.1.2 Training

We set the regularization constants by computing 3-fold cross-validation log likelihood on

[8] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html`

[9] `http://www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm`

the training data. To minimize computational complexity, we set the regularization constants for the different methods as follows. First, we set the regularization parameter $\lambda_{ML} \in \{2.5, 12.5, \cdots, 250\}$ for the ML method. Then, keeping λ_{ML} fixed, we tuned $\lambda_{ER} \in \{0.02, 0.1, \cdots, 62.5\}$ for the ML-ER method. Similarly, we set $\lambda_{LDR} \in \{12.5, 25, \cdots, 800\}$ for the ML-LDR method. Finally, keeping λ_{ML} and λ_{LDR} fixed, we set λ_{ER} for the ML-LDR-ER method. We set $\sigma^2 = 10000$ and initialized the weights for the ML method with small random initial weights. For ML-LDR and ML-ER we initialized with the solution obtained from ML. Similarly, we initialized the weights for ML-LDR-ER to the solution obtained from ML-LDR. We set $\mathbf{q}$ to be the true label distribution. We used the L-BFGS quasi-Newton method for optimizing the weights.

5.1.3 Evaluation Metrics We varied the degree of labeling (d) by controlling the percentage of examples in the interval [1-90] for which labels are available at the leaf level. When there were more than one level of internal node for a leaf node we randomly selected one of the internal nodes. We constructed 10 random train/test partitions and evaluated Macro-F1 score and accuracy on the test partitions. The Macro-F1 score is given by: $\frac{2P_aR_a}{P_a+R_a}$ where the average recall ($R_a = \frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}} R_y$) and precision ($P_a = \frac{1}{|\mathcal{Y}|}\sum_{y\in\mathcal{Y}} P_y$) are computed from the recall and precision for each category (R_y and P_y). We conduct Wilcoxon sign-rank test and make comparisons using the statistical significance results at the significance level of 0.05.

5.1.4 Experimental Results The Macro-F1 mean and standard deviation performance over 10 partitions for the two datasets are given in Table 1 and Table 3. The accuracy mean and standard deviation performance are given in Table 2 and Table 4. The results are given mainly when d is small. This is because noticeable differences between the various methods were mainly observed for small d values. In these tables, the behaviors of the methods on the Macro-F1 score and accuracy were similar. Therefore, the following observations hold for both the measures.

The performance degrades as d decreases as expected. But, the degradation is more gradual for the methods ML-LDR and ML-LDR-ER. The standard deviation is higher for these methods when $d = 1\%$. This is because on a couple of partitions, the performance improvements were around 5% (absolute) as compared to $20\% - 30\%$ (absolute) in other partitions.

ML-ER gives some improvement over ML when d is not very small. Though this improvement is not high, we observed the difference to be still statistically significant. ML-ER is inferior compared to ML-LDR. ML-LDR gives significant improvement over ML and the improvement increases as d decreases. This improvement is statistically significant. This gain is *further enhanced* by adding *entropy regularization*. This is clearly seen by comparing the performance of ML-LDR and ML-LDR-ER on the News20 dataset, for which the difference is quite significant. Although the performance difference between these methods may not be visibly high in the RCV-MCAT dataset, the results are still statistically significant. Thus, ML-LDR-ER gives the best performance.

5.2 Multi-label Classification Experiments

5.2.1 Datasets We evaluated the performance of the various methods on two popular multi-label datasets, viz., siam-competition2007 and RCV-Topics-Subsets datasets[10]. The original siam-competition2007 dataset has 21519 training and 7077 test examples. It has 30438 features and 22 outputs. The average and maximum number of (positive) labels per example were 2.16 and 10 respectively. There are five different sets of RCV-Topics-Subsets dataset. We used the first set in our experiment, and this dataset has 3000 training and 3000 test examples. It has 47236 features and 101 outputs. The average and maximum number of (positive) labels per example were 2.95 and 12 respectively. For both the datasets, to evaluate the performance on different train and test partitions, we combined the original training and test examples, and constructed 10 random train and test splits of 70% and 30% respectively.

5.2.2 Training We constructed a random train and validation split of 70% and 30%. The number of positive examples was significantly lesser compared to the negative examples for many outputs in both the datasets. So, to give same importance, we divided the marginal likelihood term for the positive and negative examples separately by their respective numbers. We set the regularization constants by computing the log likelihood of the validation set. To minimize computational complexity, we set the regularization constants for the different methods as follows. First, we set the regularization parameter $\lambda_{ML} \in \{1, 4, \cdots, 16384\}$ for the ML method. Then, keeping λ_{ML} fixed, we tuned $\lambda_{ER} \in \{0.01, 0.04, \cdots, 163.84\}$ for the ML-ER method. Similarly, we set $\lambda_{LDR} \in \{5, 25, \cdots, 78125\}$ for the ML-

[10]`http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multilabel.html`

d	ML	ML-ER	ML-LDR	ML-LDR-ER
30	97.13 (0.14)	97.26 (0.15)	97.23 (0.12)	**97.35** (0.13)
20	96.95 (0.18)	97.08 (0.17)	97.11 (0.17)	**97.22** (0.14)
15	96.54 (0.29)	96.71 (0.30)	96.96 (0.17)	**97.16** (0.14)
10	95.62 (0.70)	95.96 (0.59)	96.57 (0.27)	**96.88** (0.30)
5	91.21 (2.23)	91.33 (2.70)	94.76 (1.70)	**95.19** (1.96)
3	86.00 (3.79)	86.05 (4.29)	93.20 (1.58)	**93.56** (1.65)
1	66.53 (4.90)	65.96 (4.61)	82.47 (11.00)	**82.98** (11.57)

Table 1: Macro-F1 (multiplied by 100) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (d) on the RCV-MCAT taxonomy dataset. The best mean performance in each row is highlighted.

d	ML	ML-ER	ML-LDR	ML-LDR-ER
30	97.61 (0.10)	97.70 (0.12)	97.68 (0.08)	**97.76** (0.09)
20	97.48 (0.14)	97.58 (0.12)	97.59 (0.13)	**97.67** (0.11)
15	97.17 (0.19)	97.30 (0.20)	97.46 (0.14)	**97.60** (0.12)
10	96.56 (0.43)	96.82 (0.35)	97.19 (0.20)	**97.41** (0.21)
5	93.19 (1.28)	93.44 (1.63)	95.76 (1.22)	**96.08** (1.45)
3	88.87 (3.19)	89.11 (3.72)	94.52 (1.23)	**94.84** (1.31)
1	72.81 (1.85)	72.23 (1.66)	87.54 (6.68)	**88.50** (7.12)

Table 2: Accuracy (percentage) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (d) on the RCV-MCAT taxonomy dataset. The best mean performance in each row is highlighted.

d	ML	ML-ER	ML-LDR	ML-LDR-ER
30	84.42 (0.42)	84.46 (0.42)	**84.55** (0.44)	**84.55** (0.43)
20	83.16 (0.44)	83.66 (0.40)	83.45 (0.40)	**83.70** (0.34)
15	81.40 (0.48)	82.14 (0.61)	82.53 (0.27)	**83.18** (0.37)
10	73.86 (1.39)	72.48 (1.87)	79.41 (0.74)	**81.36** (0.53)
5	54.65 (5.99)	48.86 (5.66)	74.34 (2.33)	**77.63** (2.66)
3	35.35 (1.83)	34.63 (2.55)	75.65 (0.82)	**76.47** (0.55)
1	22.45 (2.20)	22.91 (3.30)	50.30 (13.60)	**55.62** (14.17)

Table 3: Macro-F1 (multiplied by 100) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (d) on the News20 taxonomy dataset. The best mean performance in each row is highlighted.

d	ML	ML-ER	ML-LDR	ML-LDR-ER
30	84.44 (0.38)	84.47 (0.41)	**84.59** (0.40)	84.58 (0.40)
20	83.04 (0.51)	83.51 (0.47)	83.42 (0.41)	**83.68** (0.34)
15	80.77 (0.68)	81.44 (0.98)	82.31 (0.34)	**82.99** (0.41)
10	68.93 (2.34)	65.62 (2.79)	78.35 (1.01)	**80.78** (0.75)
5	42.89 (6.58)	36.24 (5.55)	71.71 (3.47)	**76.05** (3.70)
3	25.46 (1.01)	25.07 (1.14)	75.05 (0.87)	**75.59** (0.68)
1	25.56 (1.28)	25.55 (1.29)	51.35 (12.75)	**55.68** (14.75)

Table 4: Accuracy (percentage) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (d) on the News20 taxonomy dataset. The best mean performance in each row is highlighted.

d	ML	ML-ER	ML-LDR	ML-LDR-ER	ML-LCR	ML-LCR-ER	ML-LR	ML-LR-ER
3	**58.61** (0.59)	56.96 (0.43)	57.81 (0.73)	57.80 (0.68)	58.02 (0.82)	58.09 (0.72)	58.12 (0.72)	58.15 (0.76)
1	50.62 (1.10)	49.99 (1.17)	49.50 (1.21)	49.91 (1.14)	50.74 (0.96)	51.28 (0.85)	51.07 (0.88)	**51.80** (0.93)
0.5	45.51 (1.64)	45.24 (1.59)	44.65 (0.89)	46.34 (1.07)	46.63 (0.92)	48.43 (0.80)	46.90 (0.94)	**48.73** (1.02)
0.25	40.38 (1.26)	40.02 (1.36)	41.38 (1.25)	44.26 (2.44)	43.90 (1.15)	46.41 (1.97)	44.10 (1.64)	**46.50** (2.10)
0.1	35.87 (1.80)	35.71 (1.67)	36.74 (1.34)	39.92 (3.14)	39.38 (1.73)	42.88 (1.83)	40.85 (1.22)	**43.25** (1.34)
0.05	32.13 (2.57)	32.13 (2.56)	34.66 (1.14)	36.91 (2.97)	34.01 (1.79)	36.36 (4.21)	38.99 (1.33)	**41.79** (1.30)

Table 5: Macro-F1 (multiplied by 100) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (*d*) on the siamcompetition2007 multi-label dataset. The best mean performance in each row is highlighted. Note that we use the abbreviation LR when both LDR and LCR are used.

d	ML	ML-ER	ML-LDR	ML-LDR-ER	ML-LCR	ML-LCR-ER	ML-LR	ML-LR-ER
3	19.29 (0.43)	18.48 (0.35)	19.18 (0.50)	19.16 (0.46)	19.26 (0.62)	19.27 (0.56)	**19.48** (0.57)	19.47 (0.58)
1	14.21 (0.75)	13.95 (0.70)	14.22 (0.72)	14.32 (0.71)	14.48 (0.66)	14.68 (0.63)	14.80 (0.70)	**15.06** (0.76)
0.5	11.53 (0.82)	11.50 (0.84)	11.74 (0.90)	12.15 (0.59)	11.88 (0.81)	12.54 (0.62)	12.11 (0.83)	**12.63** (0.95)
0.25	9.37 (0.96)	9.27 (0.93)	9.98 (1.07)	10.79 (1.16)	10.20 (1.05)	**11.03** (1.37)	10.41 (1.01)	10.95 (1.17)
0.1	7.54 (0.78)	7.43 (0.75)	8.32 (1.12)	9.19 (1.47)	8.81 (0.92)	9.93 (0.97)	9.10 (0.89)	**9.64** (0.89)
0.05	6.42 (0.65)	6.41 (0.65)	7.47 (0.82)	8.33 (1.13)	7.82 (0.72)	**8.41** (0.82)	8.04 (0.63)	8.35 (0.74)

Table 6: Accuracy (FULL) (percentage) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (*d*) on the siamcompetition2007 multi-label dataset. The best mean performance in each row is highlighted. Note that we use the abbreviation LR when both LDR and LCR are used.

d	ML	ML-ER	ML-LDR	ML-LDR-ER	ML-LCR	ML-LCR-ER	ML-LR	ML-LR-ER
20	70.58 (0.73)	69.59 (0.84)	70.76 (0.71)	70.46 (0.64)	**71.31** (0.64)	71.12 (0.69)	70.84 (0.70)	70.70 (0.70)
15	68.03 (0.70)	66.84 (0.67)	68.96 (0.73)	68.60 (0.68)	**69.78** (0.68)	69.60 (0.68)	69.25 (0.74)	69.14 (0.72)
10	63.73 (0.60)	62.58 (0.56)	66.18 (0.58)	65.75 (0.53)	**67.51** (0.65)	67.37 (0.63)	66.92 (0.62)	66.88 (0.68)
5	55.21 (0.66)	53.97 (1.10)	61.12 (1.00)	60.91 (0.97)	**63.14** (1.01)	62.88 (0.94)	62.68 (1.10)	62.63 (1.14)
3	48.39 (0.87)	47.04 (1.09)	55.80 (1.38)	55.68 (1.31)	**57.31** (0.91)	57.00 (0.86)	57.08 (1.27)	56.79 (1.17)
1	34.56 (1.24)	32.70 (1.55)	44.12 (1.73)	44.26 (1.68)	43.02 (2.55)	43.22 (1.63)	**46.75** (1.68)	46.65 (1.56)

Table 7: Macro-F1 (multiplied by 100) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (*d*) on the RCV-Topics-Subsets multi-label dataset. The best mean performance in each row is highlighted. Note that we use the abbreviation LR when both LDR and LCR are used.

d	ML	ML-ER	ML-LDR	ML-LDR-ER	ML-LCR	ML-LCR-ER	ML-LR	ML-LR-ER
20	32.76 (1.58)	30.86 (1.62)	32.19 (1.66)	31.78 (1.65)	**33.11** (1.50)	32.81 (1.56)	32.35 (1.64)	32.11 (1.66)
15	29.25 (1.50)	27.17 (1.44)	29.62 (1.59)	28.98 (1.48)	**30.71** (1.46)	30.48 (1.49)	29.68 (1.64)	29.59 (1.53)
10	23.68 (1.07)	21.48 (0.96)	25.72 (1.15)	25.06 (1.08)	**27.51** (1.26)	27.18 (1.29)	26.35 (1.29)	26.19 (1.39)
5	14.27 (0.90)	12.37 (1.08)	19.24 (1.19)	19.02 (1.17)	**22.19** (1.34)	21.94 (1.20)	21.10 (1.34)	21.08 (1.43)
3	9.18 (0.67)	7.90 (0.72)	14.69 (1.29)	14.52 (1.17)	**17.27** (1.36)	16.99 (1.26)	15.98 (1.43)	15.55 (1.35)
1	3.52 (0.71)	2.72 (0.63)	8.08 (1.00)	8.05 (0.99)	9.21 (1.01)	**9.26** (1.08)	9.08 (1.12)	8.69 (0.91)

Table 8: Accuracy (FULL) (percentage) performance (mean and standard deviation (in brackets)) as a function of degree of labeling (*d*) on the RCV-Topics-Subsets multi-label dataset. The best mean performance in each row is highlighted. Note that we use the abbreviation LR when both LDR and LCR are used.

LDR method. Finally, keeping λ_{ML} and λ_{LDR} fixed, we set λ_{ER} for the ML-LDR-ER method. We followed a similar strategy while doing experiments with LCR methods. In experiments involving LDR and LCR, we fixed λ_{LDR} first; then, optimized for λ_{LCR} in a scaled regularization constant set of λ_{LDR} (given above); we used $\frac{1}{2K}$ as the scaling factor. We set $\sigma^2 = 10000$ and initialized the weights for different methods following a similar strategy used in the taxonomy classification experiments. We set $\mathbf{q}$ to be the true label distribution for LDR, and set $\mathbf{R}$ to be the averaged outer product of label vectors of the training examples. As in the taxonomy classification experiments, we used the L-BFGS quasi-Newton method for optimizing the weights.

5.2.3 Evaluation Metrics We varied the degree of labeling (d) by controlling the percentage of outputs for which labels are available. For the siam-competition2007 dataset, we varied the percentage in the interval [0.05-3], and for the RCV-Topics-Subsets dataset, we varied in the interval [1-20]. We chose lower percentage values for the siam-competition2007 dataset because the number of training examples is significantly higher. The real benefits of label regularizations and performance difference between the methods are clearly seen only when the number of labeled outputs is small (as expected). Therefore, we report results (mean and standard deviation from 10 test partitions) only for percentage values in the intervals mentioned. We compare the methods on two multi-label classification performance metrics: (1) Accuracy (FULL) [5], and (2) Macro-F1 [4]. Accuracy (FULL) is given by the percentage of examples whose predicted labels match true labels on *all* outputs, i.e., with zero Hamming loss; this is a stringent metric compared to the conventional accuracy computed by treating each label prediction, as in a binary classification problem. Macro-F1 of the multi-label classifier is defined as: $\frac{2P_M R_M}{P_M+R_M}$ where P_M and R_M denote precision and recall respectively, given by, $P_M = \frac{\sum_{k=1}^{K} TP(k)}{\sum_{k=1}^{K} TP(k)+FP(k)}$ and $R_M = \frac{\sum_{k=1}^{K} TP(k)}{\sum_{k=1}^{K} TP(k)+FN(k)}$; $TP(k)$, $FP(k)$ and $FN(k)$ denote true positive, false positive and false negative of k^{th} classifier. We conduct Wilcoxon sign-rank test and make comparisons using the statistical significance results at the significance level of 0.05.

5.2.4 Experimental Results The Macro-F1 mean and standard deviation performance over 10 partitions for the two datasets are given in Table 5 and Table 7. The Accuracy (FULL) mean and standard deviation performance are given in Table 6 and Table 8.

Let us first consider siamcompetition2007 dataset. On Macro-F1, we see from Table 5 that performance improvement of 3-9% (absolute) is achieved at lower d values with regularizations, compared to the ML method. We also observe that entropy regularization *alone* is not helpful; it improves the performance by 2-5% (absolute) at lower d values when combined with label regularizations, namely, LDR, LCR and LR (recall that LR has both LDR and LCR). LCR performs slightly better than LDR, and LR performs better than both LDR and LCR. In most cases, the observed differences are statistically significant. Similar observations can be made on the Accuracy (FULL) performance from Table 6. Again, regularizations based methods perform better than the ML method by 0.2-2% (absolute). Note that zero Hamming loss is a stringent requirement; therefore, these improvements are significant.

Now consider RCV-Topics-Subsets dataset. We observe from Table 7 that Macro-F1 performance improvement of 3-12% (absolute) is achieved with regularizations at lower d values, compared to the ML method. Unlike in the siamcompetition2007 dataset, the main contribution comes from label regularizations, and entropy regularization is not useful in this dataset. The performance with LCR is better than LDR. The behaviors of the methods on the Accuracy (FULL) performance are also similar. Regularizations based methods perform better than the ML method by 3-6% (absolute) (at lower d values). As in the siamcompetition2007 dataset, the observed differences are statistically significant in most cases.

6 Other Structured Output Models

In the previous sections, we demonstrated our approach on two popular structured output problems. The basic ideas that were used for handling partial labels in these problems are quite general and can be applied to other problems. For example, there are several structured output problems with applications in information extraction [2] where a sequence model is used to label words in text sequences (e.g., text contents in a web page). Below, we illustrate how our approach can be applied in one such model. We only describe the ideas; their implementation and experimental evaluation are left for future work.

For the sequence learning problem, we use conditional random field [14]. CRF is a Markov random field that defines a conditional distribution of the labels $\mathbf{y}$ of a sequence conditioned on the input $\mathbf{x}$ and this conditional distribution has the form:

$$p(\mathbf{y}|\mathbf{x};\mathbf{W}) = \frac{1}{Z_{\mathbf{x}}} \exp\big(\sum_{c\in\mathcal{C}} \sum_{k} w_k f_k(\mathbf{x}_c, \mathbf{y}_c)\big) \tag{6.14}$$

where $\mathbf{W}$ denotes the model weights, $\mathcal{C}$ is the set of cliques present in the sequence and $Z_{\mathbf{x}}$ is a normalizing constant; $\mathbf{x}_c$, $\mathbf{y}_c$ denote the components of $\mathbf{x}$ and $\mathbf{y}$ present in a clique c, and $f_k(\mathbf{x}_c, \mathbf{y}_c)$s are feature functions. In linear chain sequence labeling problems [14], common feature functions are represented as $f_k(\mathbf{x}_c, \mathbf{y}_c)$ where c is either a node clique c with $\mathbf{y}_c \in \mathcal{Y}$ or an edge clique c connecting two adjacent nodes in a sequence, with $\mathbf{y}_c \in \mathcal{Y}^2$. Thus, we have two types of feature functions, namely, *node* and *edge* feature functions. In general the individual feature functions take real values; in many cases they are boolean.

For efficient optimization of the objective function in equation (4.4), it is important to compute the gradient $\frac{\partial \mathcal{H}_i(\mathbf{W})}{\partial w_k}$ efficiently and this quantity can be written as: $\sum_{\mathbf{y}_i^{(u)}} p(\mathbf{y}_i^{(u)}|\mathbf{y}_i^{(o)};\mathbf{W}) \log p(\mathbf{y}_i^{(u)}|\mathbf{y}_i^{(o)};\mathbf{W}) F_k([\mathbf{y}_i^{(o)} \;\; \mathbf{y}_i^{(u)}])$ $+\ \mathcal{H}_i(\mathbf{W})(\sum_{\mathbf{y}_i^{(u)}} p(\mathbf{y}_i^{(u)}|\mathbf{y}_i^{(o)};\mathbf{W}) F_k([\mathbf{y}_i^{(o)} \;\; \mathbf{y}_i^{(u)}])$; for notational simplicity, we drop the dependence on $\mathbf{x}_i$. The second term in this summation can be computed efficiently when the conditional entropy and conditional feature expectation are available. Note that efficient computation is made possible in the linear chain CRF with Markov order 1 by using the special structure of $F_k(\mathbf{y}_i) = \sum_{j,j+1} F_k(y_{i,j}, y_{i,j+1})$, and efficient marginal probability computation using the Viterbi algorithm. For the case of semi-supervised learning with fully labeled and fully unlabeled examples, [15] proposes a method to efficiently compute the first term of the gradient. The key idea is to construct an entropy lattice and define a dynamic program with forward and backward computations [15]. We make use of this idea and compute the relevant quantities by clamping the labels of observed nodes. Due to space limitation, we omit the details. The computational complexity is same as in the method of [15] and, is $O(m|\mathcal{Y}|^2)$ (where m is the sequence length). To compute the label regularization term, the class marginal probabilities of the unobserved nodes and their gradients are used.

7 Conclusion

In this paper we presented the ideas of using entropy and label regularizations in the partial label scenario for structured output problems. We demonstrated the effectiveness of the proposed methods on two popular structured output problems, namely, taxonomy (hierarchical) and multi-label classification. Experimental results strongly suggest using label regularization methods to get a significant improvement in both Macro-F1 and accuracy/zero Hamming loss performance, when the degree of labeling is low. In many cases, entropy regularization helps in improving the performance further. Results on other structured output problems such as sequence labeling will be reported in a future work.

References

[1] M. Álvarez, A. Pan, J. Raposo, F. Bellas, and F. Cacheda, *Extracting lists of data records from semi-structured web pages*, Data Knowl. Eng., 2 (2008), pp. 491–509.

[2] K. Bellare and A. McCallum, *Learning extractors from unlabeled text using relevant databases*, IIWeb Workshop at AAAI (2007).

[3] G. Chen and Y. Song and F. Wang and C. Zhang, *Semi-supervised multi-label learning by solving a Sylvester equation*, In SDM, (2008).

[4] W. Bi and J. T. Kwok, *Multi-label classification on tree and DAG structured hierarchies*, In ICML, (2011).

[5] S. Zhu and X. Ji and W. Yu and Y. Gong, *Multi-labelled classification using maximum entropy method*, In SIGIR, (2005).

[6] Y. Liu and R. Jin and L. Yang, *Semi-supervised multi-label learning by constrained non-negative matrix factorization*, In AAAI, (2006).

[7] L. Cai and T. Hofmann, *Hierarchical document categorization with support vector machines*, In CIKM, (2004), pp. 78–87.

[8] A. Carlson, S. Gaffney, and F. Vasile, *Learning a named entity tagger from gazetteers with the partial perceptron*, AAAI Spring Symposium on Learning by Reading and Learning to Read (2009).

[9] Y. Grandvalet and Y. Bengio, *Learning from partial labels with minimum entropy*, CIRANO Working Papers 2004s-28, CIRANO (2004).

[10] B. Hariharan, L. Zelnik-Manor, S. V. N. Vishwanathan, and M. Varma, *Large scale max-margin multi-label classification with priors*, In ICML (2010).

[11] F. Jiao, S. Wang, C.-H. Lee, R. Greiner, and D. Schuurmans, *Semi-supervised conditional random fields for improved sequence segmentation and labeling*, In ACL (2006).

[12] R. Jin and Z. Ghahramani, *Learning with multiple labels*, In NIPS (2002), pp. 897–904.

[13] T. Joachims, *Transductive inference for text classification using support vector machines*, In ICML (1999), pp. 200–209.

[14] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*, In ICML (2001), pp. 282–289.

[15] G. S. Mann, and A. McCallum, *Efficient computation of entropy gradient for semi-supervised conditional random fields*, In HLT-NAACL, pp. 109–112.

[16] G. S. Mann and A. McCallum, *Generalized expectation criteria for semi-supervised learning with weakly labeled data*, JMLR, 11 (2010), pp. 955–984.

[17] N. Nguyen and R. Caruana, *Classification with partial labels*, In KDD (2008), pp. 551–559.

[18] X. Zhu, *Semi-supervised learning literature survey*, Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.