HHS Public Access

Author manuscript

Proc SIAM Int Conf Data Min. Author manuscript; available in PMC 2015 April 27.

Published in final edited form as:

Proc SIAM Int Conf Data Min. 2014 ; 2014: 992–1000. doi:10.1137/1.9781611973440.113.

An Optimization-based Framework to Learn Conditional Random Fields for Multi-label Classification

Mahdi Pakdaman Naeini[*], **Iyad Batal**[†], **Zitao Liu**[‡], **CharmGil Hong**[‡], and **Milos Hauskrecht**[‡]

[*]Intelligent Systems Program, University of Pittsburgh

[†]General Electric Global Research Center

[‡]Department of Computer Science, University of Pittsburgh

Abstract

This paper studies multi-label classification problem in which data instances are associated with multiple, possibly high-dimensional, label vectors. This problem is especially challenging when labels are dependent and one cannot decompose the problem into a set of independent classification problems. To address the problem and properly represent label dependencies we propose and study a pairwise conditional random Field (CRF) model. We develop a new approach for learning the structure and parameters of the CRF from data. The approach maximizes the pseudo likelihood of observed labels and relies on the fast proximal gradient descend for learning the structure and limited memory BFGS for learning the parameters of the model. Empirical results on several datasets show that our approach outperforms several multi-label classification baselines, including recently published state-of-the-art methods.

1 Introduction

In standard binary classification problem data instances are assigned to one of the two classes (or labels). In multi-label classification (MLC) problems data instances are associated with subsets of labels selected from a set of possible labels L. Multi-label classification problems may arise in text categorization, where text documents are associated with topics the document covers; or in image analysis and retrieval where individual image can be associated with multiple labels reflecting, for example, objects shown in the image.

The multi-label classification problem can be cast as an $|L|$ dimensional classification problem where each instance $\mathbf{x}$ is mapped to an output $\mathbf{y}$ defined by a $\{0, 1\}$ vector of values of length $|L|$, such that the i-th component of the $\mathbf{y}$ vector reflects whether the i-th label in L is absent or present when labeling $\mathbf{x}$.

In this work, we study the problem of learning multi-label classifiers from data. This problem is challenging for two reasons; the number of possible label assignments is exponential in the number of labels, and the occurrence of different labels and their combinations is not arbitrary and some label combinations may be more or less likely reflecting the dependencies among labels. This prompts us to find ways of representing the

dependencies among labels in a compact and tractable form, and to devise algorithms capable of learning such dependencies from data.

Many different multi-label classification models and methods have been devised in recent years. In this work we propose, develop and test a model based on the Conditional Markov Random field (CRFs) [16]. Briefly, CRFs define a class of conditional probability models representing $P(Y|X)$, where Y is the output label space and X is the input feature space. The CRF model is based on the Markov random field (MRF) representation [14] to model dependencies/interactions among components of Y using potential functions over subsets of variables. However, CRF extends (unconditional) MRF by permitting to condition the potential functions on features.

In this work we limit the dependencies expressed by our model to a pairwise conditional random Field (CRF) model, where each potential function depends on at most two label variables. We develop a new approach for learning the structure and parameters of such a CRF from data. The approach maximizes the pseudo likelihood of observed labels and relies on the fast proximal gradient descend [2] for learning the structure of the model, and limited memory Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS) [18] for learning the parameters of the model. Empirical results on several datasets from different domains show that our approach outperforms several multi-label classification baselines, including recently published state-of-the-art methods.

1.1 Related work

The majority of existing methods for solving the multi-label classification (MLC) problems fall into two main categories: dimensionality-reduction, and probabilistic approaches. Both of these methods try to find the inherent structure in data and represent the output label space more compactly.

The dimensionality reduction methods, such as singular value decomposition (SVD), attempt to reduce the dimensionality of the output space $\mathbf{Y}$ by defining an intermediate lower dimensional space $\mathbf{Y}'$ that is sufficient to model the relations in between input features and output lables. Multiple approaches fall into this category including the work by Tai and Lin [22] who applied principal component analysis to find the intermediate lower dimensional space, or the work by Zhang and Schneider [31] who applied canonical correlation analysis. They also proposed a maximum margin formulation to find a low dimensional subspace by optimizing the subspace from both the feature side and the label side [32]. That is, the subspace should be easy to predict from the features and at the same time should discriminate well between the labels.

The probabilistic multi-label classification methods attempt to model the relation in between the input space $\mathbf{X}$ and output space $\mathbf{Y}$ probabilistically, by modeling the conditional distribution $P(\mathbf{Y}|\mathbf{X})$. The existing methods include classifier chains (CC) methods [20, 27]. Briefly, both of CC methods attempt to model $P(\mathbf{Y}|\mathbf{X})$ by decomposing it to a product of conditional densities based on the chain rule. The difference is that [20] approached the problem by using a random ordering of output variables, while [27] tried to optimize their ordering first. Other probabilistic MLC methods include conditional tree structured Bayesian

networks (CTBN) [1], and multi-dimensional Bayesian networks [25]. CTBN represents the distribution of $P(\mathbf{X}|\mathbf{Y})$ by modeling the relations among the labels using a tree structured Bayesian network. Multi-dimensional Bayesian networks define the joint probability distribution $P(\mathbf{X}, \mathbf{Y})$ using a Bayesian network with a restricted structure that puts all class variables as ancestors of all feature variables.

Other approaches to multi-label classification include work by Clare et. al. [6] and Boutell et. al. [4] who studied methods for learning a set of independent classifiers. Godbole et. al. [10] and Cheng et. al. [5] improved these early attempts by introducing a second layer of classifiers that combine input features with the outputs of independent classifiers. Zhang et. al. proposed several multi-label prediction algorithms, such as RBF neural networks [28], multi-label lazy learning approach [30], multi-instance multi-label RBF neural networks [26], multi-label BP neural networks [29]. Ji et. al. [13] used matrix decomposition to generate orthogonal linear prediction functions. Hsu et. al. [12] proposed to use compressed sensing in the label space for multi-label prediction. Liu et. al. [19] formulated the problem as a constrained Non-negative Matrix Factorization (NMF) problem. Although these methods work well for a small number of labels, they do not scale-up well to higher dimensional input and output space.

In this paper, we propose a new MLC algorithm that learns a prediction model by modeling the structure of the output space using Conditional Random Fields (CRF). Our method learns jointly the structure and parameters of the CRF from data. CRF has been used for multi-label classification of text and image data [9, 21, 11]. Compared to our approach, these applications make additional simplifying assumptions; for example, they rely on an a priori fixed structure of the output space, and/or assume a discrete set of features.

2 Proposed Model

We first describe the parametrization of the proposed multi-label classification model in Section 2.1. Section 2.2 defines the log-pseudo likelihood as the objective function and formulates the optimization problem. Finally, Section 2.3 describes the optimization methods we use to learn the model.

2.1 Model Parametrization

Our multi-label classification model uses pairwise potential *conditional random fields* (CRF). CRF directly models the conditional distribution $p(\mathbf{y}|\mathbf{x})$ with an associated graphical structure [16]. A CRF under graph $\mathcal{G} = (V, E)$ with node potentials and pairwise potentials can be characterized as follows:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{i \in V} \phi_i(y_i, \mathbf{x}) \prod_{(i,j) \in E} \phi_{ij}(y_i, y_j, \mathbf{x}) \quad (2.1)$$

where $\mathbf{x} \in \mathbb{R}^m$ is the feature vector for the input instance of multi-label classification model and $\mathbf{y} = (y_1, \ldots, yd)$ is its corresponding output labels. Also, φ_i is the node potential for every node (label) in V and φ_{ij} is the edge potential for each edge (pair of labels) in E. Let d

denotes the number of **y**s and Z denotes the partition (normalization) function $Z(\mathbf{x}) = \Sigma_{y_1,\ldots,y_d} \Pi_{i\in V} \varphi_i(y_i, \mathbf{x}) \Pi_{ij\in E} \varphi_{ij}(y_i, y_j, \mathbf{x})$.

In this paper we assume that node potentials and the edge potentials have the following form:

$$\phi_i(.,\mathbf{x}) = (e^{f_i(\mathbf{x})\mathbf{v}_i^0}, e^{f_i(\mathbf{x})\mathbf{v}_i^1}) \quad (2.2)$$

$$\phi_{ij}(.,.,\mathbf{x}) = \begin{pmatrix} e^{f_{ij}(\mathbf{x})\mathbf{w}_{ij}^{0,0}} & e^{f_{ij}(\mathbf{x})\mathbf{w}_{ij}^{0,1}} \\ e^{f_{ij}(\mathbf{x})\mathbf{w}_{ij}^{1,0}} & e^{f_{ij}(\mathbf{x})\mathbf{w}_{ij}^{1,1}} \end{pmatrix}, \quad (2.3)$$

where $(\mathbf{v}_i^0, \mathbf{v}_i^1)$ are the node parameters for node i and $(\mathbf{w}_{ij}^{0,0}, \mathbf{w}_{ij}^{0,1}, \mathbf{w}_{ij}^{1,0}, \mathbf{w}_{ij}^{1,1})$ are the edge parameters for edge (i, j). Also, $f_i(\mathbf{x})$ and $f_{ij}(\mathbf{x})$ are row vectors of local features corresponding to node i and edge (i, j). These features can be extracted using any feature extraction method such as (kernel) linear discriminant analysis, L_1 regularization, etc. However, for the simplicity we used all input features **x** as both local node and edge features. In addition, for the identifiability of the model we set $\mathbf{v}_i^1 = \mathbf{w}_{ij}^{1,1} = 0$.

2.2 Objective Function

The most common objective function used in probabilistic modeling is the log-likelihood function of training data. However, learning CRF parameters using log-likelihood requires inference in computing the gradient of the log-likelihood. If exact inference is used, it would be exponential in the tree width which is intractable for dense graphs [15]. To address this problem, one can either use approximate inference [17] or optimize a surrogate function that is easier to deal with.

The other objective function that is commonly used in probabilistic modeling is the *log-pseudo likelihood* of training data, which is known to give a consistent estimator of model parameters when the number of training data is sufficiently large [3]. Assume we have a set of N multi-label training instances $D = \{(\mathbf{x}^n; y_1^n, \ldots, y_d^n)\}_{n=1}^N$. Defining $\boldsymbol{\theta} = \{\mathbf{v}_i, \mathbf{w}_{i,j}\}$ as the set of all node and edge parameters of the model, the log-pseudo likelihood of training data is defined as follow:

$$lpl(D;\boldsymbol{\theta}) = \sum_{n=1}^{N}\sum_{i=1}^{d} \log(P(y_i^n | \mathbf{y}_{N(i)}^n, \mathbf{x}^n)) \quad (2.4)$$

Using our model parameterizations in Equations 2.2,2.3 we have:

$$P(y_i^n | \mathbf{y}_{N(i)}^n, \mathbf{x}^n) = \frac{\exp\{f_i(\mathbf{x}^n)\mathbf{v}_i^{y_i} + \sum_{j\in N_i} f_{ij}(\mathbf{x}^n)\mathbf{w}_{ij}^{y_i,y_j}\}}{Z_i},$$

where Z_i is the local partition function which normalizes with respect to y_i. Consequently, *lpl* is computed as following:

$$lpl(D;\boldsymbol{\theta})=\sum_{i=1}^{d}\sum_{n=1}^{N}\left(f_i(\mathbf{x}^n)\mathbf{v}_i^{y_i^n}+\sum_{j\in N(i)}f_{ij}(\mathbf{x}^n)\mathbf{w}_{ij}^{y_i^n,y_j^n}\right.$$
$$-\log\left\{\exp(f_i(\mathbf{x}^n)\mathbf{v}_i^0+\sum_{j\in N(i)}f_{ij}(\mathbf{x}^n)\mathbf{w}_{ij}^{0,y_j^n}) \quad (2.5)\right.$$
$$\left.\left.+\exp(\sum_{j\in N(i)}f_{ij}(\mathbf{x}^n)\mathbf{w}_{ij}^{1,y_j^n})\right\}\right)$$

We can see in the above equation that *lpl*(*D*; $\boldsymbol{\theta}$) is a concave function in both node and edge parameters.

To prevent model overfitting we use a regularization term $R(\boldsymbol{\theta})$ that penalizes model complexity. In our method $R(\boldsymbol{\theta})$ is defined as following:

$$R(\theta)=\sum_{i=1}^{d}\lambda_{\mathbf{v}_i}\|\mathbf{v}_{i,0}\|_2^2+\lambda_w\left((1-\eta)\sum_{e\in E}\|\mathbf{w}_e\|_2^2+\eta\sum_{e\in E}\|\mathbf{w}_e\|_2\right), \quad (2.6)$$

where $\mathbf{w}_e=(\mathbf{w}_e^{0,0};\mathbf{w}_e^{0,1};\mathbf{w}_e^{1,0})$. The first term in $R(\boldsymbol{\theta})$ is the ridge penalty (ℓ_2 norm regularization) over node parameters. The second term (inside the parentheses) is the *elastic-net* penalty [34], which is a compromise between the ridge regression penalty ($\eta = 0$) and the group lasso penalty ($\eta = 1$). This penalty imposes block sparsity effect over the edge parameters of the CRF model. That is, for a specific edge in the graph, either all its parameters go to zero (the edge is absent from the graph) or not (the edge exists in the graph). For setting the regularization parameters $\lambda_{\mathbf{v}i}$, λ_w, η, we apply the following sequential cross validation process. We first set $\lambda_{\mathbf{v}i}$ separately for each node (*yi*) using cross-validation over independent logistic regression models. After that, we apply the LARS (or homotopy) method [8] to set λw and η, using a k-fold cross validation and a warm start procedure.

By defining *nlpl* $\triangleq -$*lpl* as the corresponding loss function, we can learn the structure and parameters of the model by minimizing the regularized loss function as following:

$$\boldsymbol{\theta}^*=argmin_{\boldsymbol{\theta}}(nlpl(D;\boldsymbol{\theta})+R(\boldsymbol{\theta})) \quad (2.7)$$

Since both *nlpl*(*D*; $\boldsymbol{\theta}$) and $R(\boldsymbol{\theta})$ are convex functions, the above optimization program is a convex optimization program in which all local minima are global minima.

2.3 Optimization

In this section we describe the proposed optimization method for solving the optimization program in Equation 2.7. In order to use any gradient-based optimization, we should be able to compute the gradient of *nlpl*. Taking derivative with respect to $\mathbf{v}_i^0$ (the parameters of node *i*), we would have the following:

$$\nabla_{\mathbf{v}_i^0} nlpl = \sum_{n=1}^{N} -f_i(\mathbf{x}^n)^T \mathbb{I}(y_i^n=0) + \sum_{n=1}^{N} f_i(\mathbf{x}^n)^T P(y_i^n=0|\mathbf{y}_{N(i)}^n, \mathbf{x}^n), \quad (2.8)$$

where superscript T denotes the transpose. Also taking the derivative of the *nlpl* with respect to $\mathbf{w}_{ij}^{b_i,b_j}$ (the parameters of edge (*i*, *j*) and values (b_i, b_j)), we would have the following:

$$\begin{aligned} &\nabla_{\mathbf{w}_{ij}^{b_i,b_j}} nlpl = \\ &\sum_{n=1}^{N} -2 f_{ij}(\mathbf{x}^n)^T \mathbb{I}(y_i^n=b_i, y_j^n=b_j) + \\ &\sum_{n=1}^{N} f_{ij}(\mathbf{x}^n)^T \Big(P(y_i^n=b_i|\mathbf{y}_{N(i)}^n, \mathbf{x}^n) \mathbb{I}(y_j^n=b_j) \\ &+ P(\mathbf{y}_j^n=b_j|\mathbf{y}_{N(j)}^n, \mathbf{x}^n) \mathbb{I}(y_i^n=b_i) \Big) \end{aligned} \quad (2.9)$$

Fast Proximal Gradient Descent—Although the minimization problem in equation 2.7 is a convex optimization problem, the objective function is non-smooth due to the non-smoothness of the $\ell_{1,2}$-norm induced by the elastic-net regularization term in $R(\boldsymbol{\theta})$. To overcome this difficulty, we use the fast proximal gradient decent algorithm [2] to solve this non-smooth optimization problem.

The unconstrained problem in Equation 2.7 can be optimized with two approaches. The first approach directly defines proximal mapping for the non-smooth part of the elastic-net penalty i.e. the L_2 penalty over the edge variables. The second approach converts the unconstrained program to an equivalent constraint optimization program by defining slack variables α_e as in Equation 2.10. These two approaches are very similar and they both have the same order of computations. However, we use the latter one mainly because in this case we can use the slack variables α_e for monitoring existence of edges in the CRF model and for early termination of the structure learning phase (having $\alpha_e = 0$ means that edge *e* is not present in the graph structure). In the following, for the simplicity of representation, we use $\boldsymbol{\theta}_k$ to denote the parameters vector $\{\boldsymbol{\alpha}, \mathbf{w}, \mathbf{v}\}$ at iteration *k* and $f(\boldsymbol{\theta}_k)$ to denote its corresponding objective function.

$$\begin{aligned} &\underset{\boldsymbol{\alpha},\mathbf{v},\mathbf{w}}{\text{minimize}} \left(nlpl(\boldsymbol{\theta}) + \sum_{i=1}^{d} \lambda_{\mathbf{v}_i} \|\mathbf{v}_i^0\|_2^2 + \lambda_{\mathbf{w}} ((1-\eta) \sum_{e \in E} \|\mathbf{w}_e\|_2^2 + \eta \sum_{e \in E} \alpha_e) \right) \\ &\text{subject to} \|\mathbf{w}_e\|_2 \leq \alpha_e, \; e \in E. \end{aligned} \quad (2.10)$$

For solving the optimization program in Equation 2.10, we use FPGD with backtracking line search. The high level idea of proximal gradient descend methods for minimizing a non-smooth function *f* is to write it as two separate parts $f = g + h$, in which *g* is a smooth differentiable function and *h* is a *simple* non-differentiable convex function. In our case, $g = nlpl(D;\boldsymbol{\theta}) + \lambda_v \sum_{i=1}^{d} \|\mathbf{v}_i^0\|_2^2 + \lambda_w \left((1-\eta) \sum_{e \in E} \|\mathbf{w}_e\|_2^2 + \eta \sum_{e \in E} \alpha_e \right)$ and the non-smooth function $h = \mathbf{I}_\Omega(\boldsymbol{\theta})$, where $\Omega = \{(\boldsymbol{\alpha}, \mathbf{w}, \mathbf{v}) | \forall e \in E : ||\mathbf{w}_e||_2 \leq \alpha e\}$ is the set of feasible solutions (a convex set) and $\mathbf{I}_\Omega$ is the indicator function defined as follows:

$$\mathbf{I}_{\Omega}(\boldsymbol{\theta})=\begin{cases} 0 & \text{if } \boldsymbol{\theta} \in \Omega \\ \infty & \text{otherwise} \end{cases}$$

Proximal gradient based optimization methods use a proximal function defined as $prox_t(\mathbf{x})=argmin_{\mathbf{z}\in\mathbb{R}^n}\frac{1}{2t}\|\mathbf{x}-\mathbf{z}\|+h(\mathbf{z})$ to optimize a second order approximation of the objective function *f*. In our case, the proximal function is the projection operator $\Pi(\mathbf{w}_e, \alpha_e)$ of the parameters onto the convex set of feasible solutions Ω (defined above). Interestingly, the the above projection operator can be decoupled and solved separately over different edges of CRF model which gives following simple program for each edge:

$$\begin{aligned} \underset{\mathbf{w_e}',\alpha_e'}{\text{minimize}} \quad & \|(\mathbf{w_e}', \alpha_e')-(\mathbf{w_e}, \alpha_e)\|_2^2 \\ \text{subject to} \quad & \|\mathbf{w_e}'\|_2 \leq \alpha_e' \end{aligned} \tag{2.11}$$

For solving the above optimization program we used the linear-time projection algorithm proposed in [24]. Algorithm 1 outlines the optimization method. In this algorithm the termination condition is defined based on the maximum number of iterations (50 in our experiments), the norm of the changes in value of $\boldsymbol{\alpha}$, or the change in the function values $|f(\boldsymbol{\theta}_k) f(\boldsymbol{\theta}_{k-1})|$.

Algorithm 1

Fast proximal gradient descend with line search.

$t_0 \in \mathbb{R}^+$

$\boldsymbol{\theta}_0 = \boldsymbol{\theta}_{-1} \in \mathbb{R}^n$

$k \leftarrow 1$

repeat

$\mathbf{y}=\boldsymbol{\theta}_{k-1}+\frac{k-2}{k+1}(\boldsymbol{\theta}_{k-1}-\boldsymbol{\theta}_{k-2})$

$t_k = t_{k-1}$

$\boldsymbol{\theta}_k = prox_{t_k}(t_k \nabla g(\mathbf{y}))$

while $g(\boldsymbol{\theta}_k)>g(\mathbf{y})+\nabla g(\mathbf{y})^T(\boldsymbol{\theta}^{(k)}-\mathbf{y})+\frac{1}{2t_k}\|\boldsymbol{\theta}_k-\mathbf{y}\|_2^2$

do

$t_k \leftarrow \beta t_k$

$\boldsymbol{\theta}_k = prox_{t_k}(\mathbf{y} - t_k \nabla g(\mathbf{y}))$

end while

$k \leftarrow k+1$

until (termination condition)

Parameter Tuning—After fixing the structure of the CRF model by termination of FPGD optimization method, we remove the *L*1 penalty term and use limited memory BFGS [18], a quasi-newton optimization method, to tune the model parameters. We found this step

necessary for having a high performance predictive model and it can affect the final results, up to 10% in terms of exact match between the predicted and the true labels.

3 Experimental Setting

In this section, we will discuss the baseline methods as well as the measures used to evaluate the performance of MLC.

3.1 Methods

We compare our proposed method, which we refer to as MLCRF, with simple binary relevance (BR) independent classification (as in [6, 4]) as well as several state-of-the-art multi-label learning methods. These methods include the most recent dimensionality reduction based approach: maximum margin output coding (MMOC) [32], two probabilistic approaches: classifier chains (CC) [20] and probabilistic classifier chains (PCC) [7], and three other methods: classification with heterogenous features (CHF) [10], multi-label k-nearest neighbor (MLKNN) [30] and instance-based with logistic regression (IBLR) [5].

For all methods, we use the same parameter settings as suggested in their papers: For MMOC, λ (decoding parameter) is set to 1 [32]; For CC we we set the order of classes to $Y_1 < Y_2, \ldots < Yd$ [20]; For MLKNN and IBLR, Euclidean distance is used to measure similarity of instances, and the number of nearest neighbours is set to 10 [30, 5]. Also, note that all baseline methods except MMOC, are kind of meta-learners because they can work with several base classifiers. We use L_2-penalized logistic regression for all of these methods and choose their regularization parameters by cross validation.

3.2 Evaluation Measures

Evaluating the performance of MLC methods is more difficult than evaluation of simple classification methods. The most suitable performance measure is the *exact match accuracy* (EMA), which computes the percentage of instances whose predicted label vectors are exactly the same as their true label vectors. However, this measure could be too harsh, specially when the output dimensionality is high. The other evaluation measure is the *log-likelihood loss* (LL) which computes the negative conditional log-likelihood of the test instances:

$$LL=\sum_{k=1}^{n}-\log\left(P(\mathbf{y}^{(k)}|\mathbf{x}^{(k)})\right)$$

LL is very insightful in evaluating the performance of probabilistic MLC methods because it evaluate how much probability mass is given to the true label vectors (the higher the probability, the smaller the loss). Micro F1 and Macro F1 are two other intuitive measures for evaluation MLC methods [23]. Macro F1 computes the F1 score for each class label separately and averages them over all labels. On the other hand, micro F1 aggregates the true positives, false positives, true negatives and false negatives over all labels, and then calculates the overall F1 score. Note that these two measures do not account for the

correlations between labels (see [7, 27]). However, we report them in our results because they have been used in several other papers [32, 33].

4 Experimental Results

We perform two sets of experiments to evaluate the proposed model. First, in order to evaluate the predictive performance of MLCRF, which is the main point of this paper, we perform experiments on several real-world data. In addition, we perform some experiments on synthetic data to verify that the FPGD-based structure learning algorithm of MLCRF also performs reasonably well in finding the true conditional dependencies among output labels.

4.1 Real data

We use five real-world benchmark datasets to evaluate the performance of the MLC models. These datasets are obtained from different domains including image labeling (image and scene), biology (yeast), music recognition (emotions) and text classification (rcv1). Note that many labels in rcv1 data are rare, so we select the 10 most common labels to study. Table 1 shows the characteristics of these datasets. For each multi-label dataset, we use N, m, d to represent the number of instances, number of features (input dimension) and number of labels (output dimension), respectively. In addition, we show two statistics of the data: 1) label cardinality (LC), which measures the average number of labels per instance and 2) distinct label set (DLS), which is the number of all possible label combinations that appear in data.

Tables 2, 3, 4 and 5 show the results of experiments in terms of different evaluation metrics. Standard *tenfold cross validation* is applied for all methods and the mean metric value is reported. Furthermore, to statistically measure the significance of performance difference, paired t-tests at 0.05 significance level are conducted between MLCRF and the other methods. Specifically, whenever MLCRF achieves significantly better/worse performance than the compared method on any dataset, a marker */⊛ is shown in the table. Note that the results of MMOC is not shown on rcv1 data because it did not finish in the time limit of 24 hours for one iteration of learning.

Table 2 shows the exact match accuracy of the methods on the benchmark datasets. We can see that MLCRF outperforms all the other methods for most datasets.

Table 3 shows the performance of each method in terms of log-likelihood loss (LL). Since the output dimensionality of the datasets is relatively small, we use exact inference in the prediction time to obtain accurate probabilistic estimates[1]. We can see that MLCRF is significantly superior to all other methods in terms of LL. This shows that the pseudo likelihood, which MLCRF optimizes, is a good surrogate for the likelihood of the data. Notice that CC performs poorly because of its ad-hoc classification heuristic [7]. Also, note that we cannot compute LL for MMOC because it is not a probabilistic method.

[1]When the output dimensionality is large, we should resort to approximate inference using either variational approximation or Gibbs sampling

Tables 4 and 5 show that MLCRF also produces competitive results in terms of the micro and macro F1 scores.

4.2 Synthetic data

We perform simple experiments on synthetic data to verify that the structure learning part of the proposed method is performing reasonably well. In the following, we describe four different scenarios, each corresponds to a different graph topology with three output nodes, as shown in Figure 1. We generate 500 data instances: $D=\{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}\}_{n=1}^{500}$, where $\mathbf{x}^{(n)} \in \mathbb{R}^6$ and $\mathbf{y}^{(n)} \in \{0, 1\}^2$. The feature vectors are generated uniformly from $[-1, 1]^6$. On the other hand, the class vectors are generated based on the following strategies:

Independent (SD1)—For each instance $\mathbf{x}^{(n)}$, we first generate random variables μ_1, μ_2, μ_3 that depends on the features as follows:

$\mu_1=\sigma(x_1^{(n)}+x_2^{(n)}),\ \mu_2=\sigma(3x_3^{(n)}+3x_4^{(n)}),\ \mu_3=\sigma(5x_5^{(n)}+5x_6^{(n)})$ where σ is the logistic (sigmoid) function. $y_1^{(n)}$, $y_2^{(n)}$, $y_3^{(n)}$ will be bernoulli random variables generated using μ_1, μ_2, μ_3. Note that all three classes are conditionally independent of each other.

Single Edge (SD2)—For each instance $\mathbf{x}^{(n)}$, we first generate random variables μ_1, μ_2 that depend on the features as follows: $\mu_1=\sigma(x_1^{(n)}+x_2^{(n)}),\ \mu_2=\sigma(3x_3^{(n)}+3x_4^{(n)})$. y_3, z will be bernoulli random variables generated using μ_1, μ_2. If the value of z is zero, we assign $(y_1, y_2)^{(n)}$ = (0, 0) (both classes are absent). Otherwise (the value of z is one), we assign $(y_1, y_2)^{(n)}$ to either (1, 0) or (0, 1) with equal probability (only one of the two classes is present). Note that we do not have any instance where both classes are present.

Chain Structure (SD3)—For each instance $\mathbf{x}^{(n)}$, we first generate random variables μ_1, μ_2 that depends on the features as follows: $\mu_1=\sigma(x_1^{(n)}+x_2^{(n)}),\ \mu_2=\sigma(3x_3^{(n)}+3x_4^{(n)})$. $y_1^{(n)}$, $y_2^{(n)}$ will be bernoulli random variables generated using μ_1, μ_2. If both $y_1^{(n)}$ and $y_2^{(n)}$ are equal to one we set $y_3^{(n)}=0$; Otherwise, we set $y_3^{(n)}=1$.

Complete Graph (SD4)—For each instance $\mathbf{x}^{(n)}$, we first generate a random variable μ_1 that depends on the features as follows: $\mu_1=\sigma(x_1^{(n)}+x_2^{(n)})$. z will be bernoulli random variables generated using μ_1. If z is equal to one we set $(y_1, y_2, y_3)^{(n)}$ = (0, 0, 0); Otherwise, we set $(y_1, y_2, y_3)^{(n)}$ randomly (with equal probability) to one of the three configurations (1, 0, 0); (0, 1, 0); (0, 0, 1).

The results on synthetic data are shown in Figure 2. For each dataset, we profile the value of the auxiliary edge variables (the a_e variables in Equation 2.10) over the course of the optimization algorithm. Note that these variables imply the graph structure because having a positive $a_{i,j}$ means having an edge between class Y_i and class Y_j in the graph. As we can see, for all four datasets, the algorithm succeeds in finding the correct dependency relations among the outputs.

5 Conclusions and Future Work

In this paper, we proposed a novel probabilistic approach for classifying multi-label data. Our approach uses a special conditional random field to map the input to the multilabel output space. We use elastic net group sparsity penalty and Fast Proximal Gradient Descend (FPGD) optimization method to learn the CRF structure, and the L-BFGS optimization method to tune the parameters of the fixed-structure CRF. Our experimental evaluation on a broad range of real datasets showed that our approach outperforms several state-of-the-art methods and is able to produce reliable probabilistic estimates.

By using pseudo-likelihood for learning the parameters of the model we avoid the need of exact inference in the learning phase which would otherwise be intractable even for small size problems. However, since we use exact inference for finding the MAP output in test time, it would be still intractable when the number of outputs is very large. In the future we plan to develop and test a variational approximation to accomplish this step.

This paper has focused mainly on showing that our method can achieve a good predictive performance on many multi-label classification problems. We have also shown that the proposed structure learning method works reasonably well and recovers the structure of known CRFs from synthetic data. However, we believe a more extensive study will help us to gain additional insights on the performance of the structure learning method on more complex datasets. Another possible direction for future work is to use a max margin loss function (e.g. Hinge loss) in order to define a max-margin CRF for multi-label classification [23, 24, 26]. We conjecture that the max margin approach for learning of a CRF will lead to a more accurate MLC model when considering the exact label match, while our CRF model would perform better in terms of the conditional log-likelihood loss.

Acknowledgments

This work was supported by grants 1*R*01*GM*088224 and 1*R*01*LM*010019 from the NIH. Its content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH..

References

1. Batal, I.; Hong, C.; Hauskrecht, M. An efficient probabilistic framework for multi-dimensional classification. Proceedings of CIKM; ACM; 2013. p. 2417-2422.
2. Beck A, Teboulle M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences. 2009; 2(1):183–202.
3. Besag J. Efficiency of pseudolikelihood estimation for simple gaussian fields. Biometrika. 1977:616–618.
4. Boutell MR, Luo J, Shen X, Brown CM. Learning multi-label scene classification. Pattern Recognition. 2004; 37(9):1757– 1771.
5. Cheng W, Hüllermeier E. Combining instance-based learning and logistic regression for multilabel classification. Machine Learning. 2009; 76(2–3):211–225.
6. Clare, A.; Clare, A.; King, RD. Lecture Notes in Computer Science. Springer; 2001. Knowledge discovery in multi-label phenotype data; p. 42-53.
7. Dembczynski, K.; Cheng, W.; Hüllermeier, E. Bayes optimal multilabel classification via probabilistic classifier chains. Proceedings of the 27th International Conference on Machine Learning (ICML-10); Omnipress; 2010. p. 279-286.

8. Friedman J, Hastie T, Höfling H, Tibshirani R. Pathwise coordinate optimization. The Annals of Applied Statistics. 2007; 1(2):302–332.
9. Ghamrawi, N.; McCallum, A. Collective multi-label classification. Proceedings of CIKM; ACM; 2005. p. 195-200.
10. Godbole S, Sarawagi S. Discriminative methods for multi-labeled classification. PAKDD'04. 2004:22–30.
11. He, X.; Zemel, RS.; Carreira-Perpinán, MA. Multiscale conditional random fields for image labeling. Proceedings of CVPR; 2004; IEEE; 2004. p. II-695.
12. Hsu D, Kakade S, Langford J, Zhang T. Multi-label prediction via compressed sensing. NIPS. 2009:772–780.
13. Ji S, Tang L, Yu S, Ye J. A shared-subspace learning framework for multi-label classification. ACM Transactions on Knowledge Discovery from Data (TKDD). 2010; 4(2):8.
14. Kindermann, R.; Snell, JL., et al. Markov random fields and their applications. Vol. 1. American Mathematical Society; Providence, RI: 1980.
15. Kollar, D.; Friedman, N. Probabilistic graphical models: principles and techniques. The MIT Press; 2009.
16. Lafferty, JD.; McCallum, A.; Pereira, FCN. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proceedings of, ICML; San Francisco, CA, USA. Morgan Kaufmann Publishers Inc; 2001. p. 282-289.
17. Lee S-I, Ganapathi V, Koller D. Efficient structure learning of markov networks using *l*_1-regularization. Advances in neural Information processing systems. 2006:817–824.
18. Liu DC, Nocedal J. On the limited memory bfgs method for large scale optimization. Mathematical programming. 1989; 45(1–3):503–528.
19. Liu, Y.; Jin, R.; Yang, L. Semi-supervised multi-label learning by constrained non-negative matrix factorization. Proceedings of the National Conference on Artificial Intelligence; Menlo Park, CA; Cambridge, MA; London. 1999; AAAI Press; MIT Press; 2006. p. 421
20. Read, J.; Pfahringer, B.; Holmes, G.; Frank, E. Classifier chains for multi-label classification. Proceedings of ECML PKDD;
21. Rousu J, Saunders C, Szedmak S, Shawe-Taylor J. Kernel-based learning of hierarchical multi-label classification models. The Journal of Machine Learning Research. 2006; 7:1601–1626.
22. Tai, F.; Lin, H-T. Multi-label classification with principle label space transformation. Proceedings of the 2nd International Workshop on Multi-Label Learning; 2010.
23. Tsoumakas, G.; Zhang, M.; Zhou, Z. ECML PKDD Tutorial. 2009. Learning from multi-label data.
24. van den Berg, E.; Schmidt, M.; Friedlander, MP.; Murphy, K. Group sparsity via linear-time projection. Dept. Comput. Sci., Univ. British Columbia; Vancouver, BC, Canada: 2008.
25. van der Gaag LC, de Waal PR. Multidimensional bayesian network classifiers. Probabilistic Graphical Models. 2006:107–114.
26. Zhang ML, Wang ZJ. Mimlrbf: Rbf neural networks for multi-instance multi-label learning. Neurocomputing. 2009; 72(16):3951–3956.
27. Zhang, M-L.; Zhang, K. Multi-label learning by exploiting label dependency. Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '10; ACM; 2010. p. 999-1008.
28. Zhang ML, Zhou ZH. Adapting rbf neural networks to multi-instance learning. Neural Processing Letters. 2006; 23(1):1–26.
29. Zhang ML, Zhou ZH. Multilabel neural networks with applications to functional genomics and text categorization. IEEE Transactions on Knowledge and Data Engineering. 2006; 18(10):1338–1351.
30. Zhang ML, Zhou ZH. Ml-knn: A lazy learning approach to multi-label learning. Pattern Recogn. Jul; 2007 40(7):2038–2048.
31. Zhang, Y.; Schneider, J. AISTATS 2011. 2011. Multi-label output codes using canonical correlation analysis.
32. Zhang, Y.; Schneider, J. Maximum margin output coding. Proceedings of the 29th International Conference on Machine Learning (ICML-12), ICML '12; Omnipress; 2012. p. 1575-1582.

33. Zhu, S.; Ji, X.; Xu, W.; Gong, Y. SIGIR. ACM; 2005. Multi-labelled classification using maximum entropy method; p. 274-281.

34. Zou H, Hastie T. Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 2005; 67(2):301–320.

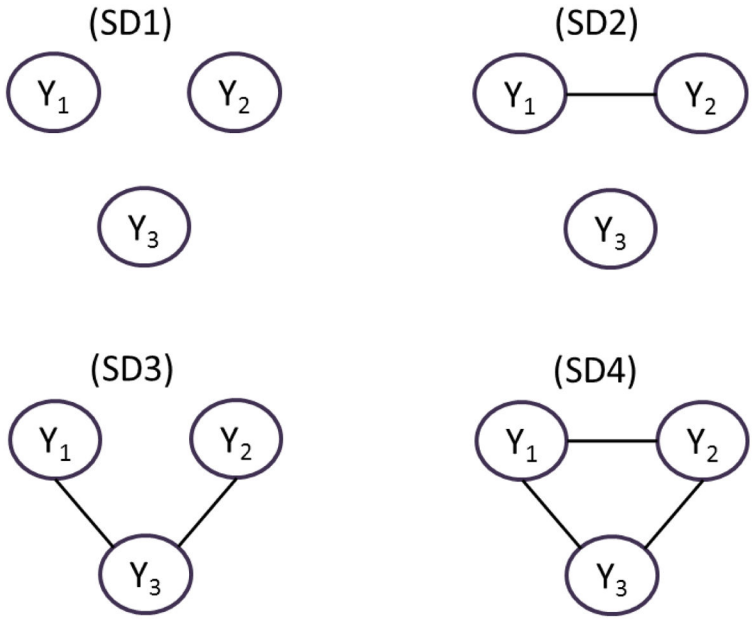

Figure 1.
Graphical model representation of four synthetic data sets.

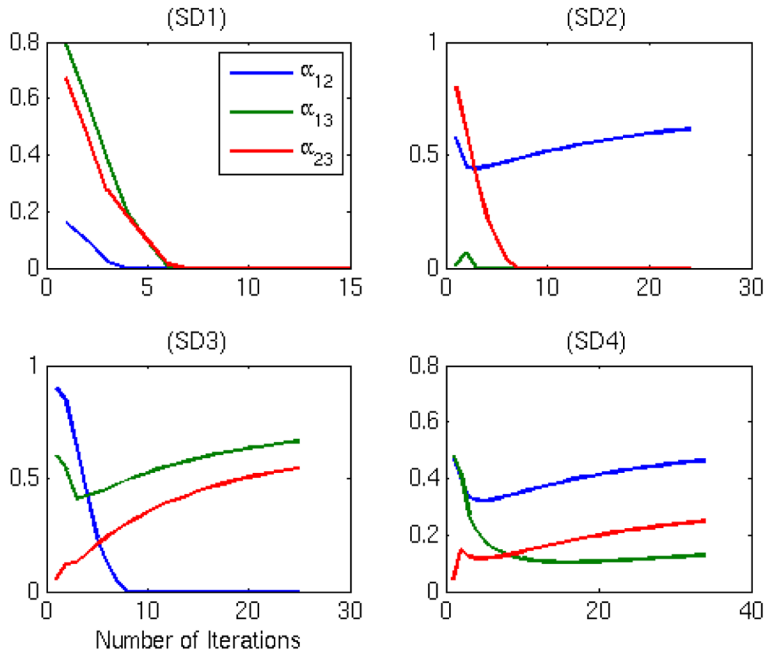

Figure 2.

Profile of auxiliary edge variables over running iterations of FPGD optimization method for four simulation dataset SD1,…, SD4, where all of these variables are initialized to a random number between 0 and 1. A positive value for α_{ij} indicates the presence of an edge between Y_i and Y_j in the CRF model.

Table 1

Datasets characteristics.

Dataset	*N*	*m*	*d*	LC	DLS
Emotions	593	72	6	1.868	27
Yeast	2417	103	14	4.237	198
Scene	2407	294	6	1.074	15
Image	2000	135	5	1.24	20
rcv1_top10	6000	8367	10	1.310	76

Table 2

Performance of each method on the benchmark datasets in terms of exact match accuracy.

Dataset	BR	CHF	CC	PCC	MLKNN	IBLR	MMOC	MLCRF
image	0.281*	0.360*	0.426*	0.449	0.346*	0.387*	0.448	0.452
scene	0.541*	0.605*	0.632*	0.666*	0.629*	0.644*	0.664*	0.687
yeast	0.151*	0.163*	0.193*	0.230	0.179*	0.204*	0.219	0.225
emotions	0.265*	0.300*	0.268*	0.317*	0.283*	0.335*	0.332*	0.371
rcv1_top10	0.439*	0.456*	0.497*	0.519*	0.276*	0.411*	-	0.541

Marker */⊛ indicates whether MLCRF is statistically superior/inferior to the compared method (using paired t-test at 0.05 significance level).

Proc SIAM Int Conf Data Min. Author manuscript; available in PMC 2015 April 27.

Table 3

Performance of each method on the benchmark datasets in terms of log-likelihood loss.

Dataset	BR	CHF	CC	PCC	MLKNN	IBLR	MLCRF
image	432.5*	415.9*	480.3*	354.7*	425.3*	395.6*	347.0
scene	344.7*	318.4*	395.0*	258.9*	310.9*	283.9*	252.4
yeast	1500.3*	1491.7*	2303.8*	932.1*	1464.9*	1434.2*	923.9
emotions	153.5*	147.5*	169.6*	134.9*	151.7*	143.0*	130.4
rcv1_top10	1267.0*	2283.6*	1421.9*	1009.0*	1795.5*	1234.7*	922.6

Marker */⊛ indicates whether MLCRF is statistically superior/inferior to the compared method (using paired t-test at 0.05 significance level).

Proc SIAM Int Conf Data Min. Author manuscript; available in PMC 2015 April 27.

Table 4

Performance of each method on the benchmark datasets in terms of micro F1.

Dataset	BR	CHF	CC	PCC	MLKNN	IBLR	MMOC	MLCRF
image	0.479*	0.541*	0.550*	0.565*	0.504*	0.573	0.572	0.576
scene	0.696*	0.722	0.697*	0.722*	0.736	0.758⊛	0.711*	0.733
yeast	0.635	0.637	0.628	0.645⊛	0.646⊛	0.661⊛	0.651⊛	0.629
emotions	0.645*	0.672*	0.621*	0.664*	0.656*	0.692	0.687	0.697
rcv1_top10	0.582*	0.597	0.595	0.600	0.314*	0.566*	-	0.598

Marker */⊛ indicates whether MLCRF is statistically superior/inferior to the compared method (using paired t-test at 0.05 significance level).

Proc SIAM Int Conf Data Min. Author manuscript; available in PMC 2015 April 27.

Table 5

Performance of each method on the benchmark datasets in terms of macro F1.

Dataset	BR	CHF	CC	PCC	MLKNN	IBLR	MMOC	MLCRF
image	0.486*	0.546*	0.562*	0.575*	0.516*	0.581	0.578	0.585
scene	0.703*	0.730*	0.709*	0.729*	0.743	0.765⊛	0.721*	0.746
yeast	0.457	0.461	0.467	0.486	0.478	0.498	0.473	0.476
emotions	0.632*	0.667*	0.620*	0.659*	0.656*	0.690	0.679*	0.695
rcv1_top10	0.500*	0.526	0.537	0.534	0.257*	0.487*	-	0.525

Marker */⊛ indicates whether MLCRF is statistically superior/inferior to the compared method (using paired t-test at 0.05 significance level).