A Generalized Framework for Multi-label Classification

Charmgil Hong*, Iyad Batal+, Milos Hauskrecht* / {charmgil,milos}@cs.pitt.edu, iyad.batal@ge.com
* Department of Computer Science, University of Pittsburgh; + GE Global Research

Paper & other resources are available at: cs.pitt.edu/~charmgil

Introduction

- **Multi-label classification (MLC)**
  - In traditional classification settings, a data instance is associated with a single class variable
  - However, in many real-world problems, it is more natural to see that each data instance can be associated with multiple class variables
  - Multi-label Classification (MLC) is the supervised learning problem that formulates such situations
  - Examples of MLC

- **Problem definition**
  - We want to learn a function h from multi-label data that h maps m-dimensional feature (input) $\mathbf{X} = \{X_1, ..., X_m\}$ to the maximum a posteriori assignment of class variables (output) $\mathbf{Y} = \{Y_1, ..., Y_d\}$:

$$h^*(\mathbf{x}) = \arg\max_{y_1,...,y_d} P(Y_1{=}y_1, ..., Y_d{=}y_d|\mathbf{X}{=}\mathbf{x})$$

- **One solution to MLC is to exploit the dependency relation among class variables**
  - By explicitly modeling the dependency among class variable, we can effectively solve MLC
  - By assuming the dependency relation forms a chain or tree structure, we can derive efficient solutions [Read et al. '09; Batal et al. '13]

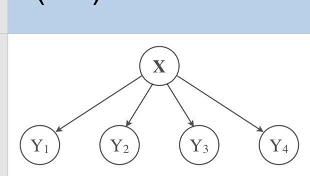
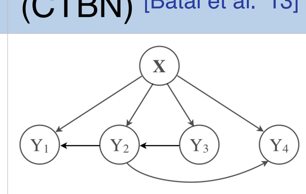
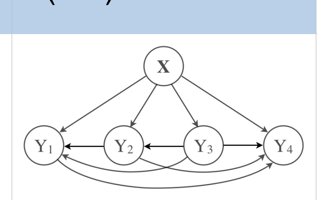
- **What if there exist more complex relations?**
  - The relations among features and class variables may change across a dataset
  - Existing methods may not be sufficient as they are designed to capture a fixed dependency relation
- **Examples of such complex relations can be found in many applications**
  - In *semantic image tagging*, an image of a cat can be tagged as *{cat, pet}* or *{cat, wild animal}* according to its context
  - In *medicine*, patients suffering from the *same disease* may receive *different sets of medications* due to their medical history or allergic reactions

Proposed Solution

- **Our key contributions**
  - We provide a unified perspective, *Classifier Chains Family*, which generalizes existing MLC methods, including [Read et al. '09; Batal et al. '13; Boutell et al. '04]
  - We present a new ensemble approach that incorporates the models in *Classifier Chain Family* using the *mixtures-of-experts* [Jacobs et al. '91] framework
    - It is a generalization of our previous work: *Mixtures-of-Conditional Tree-structured Bayesian Networks* [Hong et al. '14]
- **Classifier Chains Family (CCF)**
  - CCF is a family of structured MLC models, that decompose the multivariate class posterior distribution $P(\mathbf{Y}|\mathbf{X})$ using a product of the posteriors over individual class variables as:

$$P(\mathbf{Y}|\mathbf{X}, M) = \prod_{i=1}^{d} P(Y_i|\mathbf{X}, \mathbf{Y}_{\boldsymbol{\pi}(i,M)})$$

  where $\mathbf{Y}_{\boldsymbol{\pi}(i,M)}$ denotes the parent classes of class variable $\mathbf{Y}_i$ defined by model M
  - By specifying particular structural assumptions, we can instantiate *classifier chains* [Read et al. '09], *conditional tree-structured Bayesian networks* [Batal et al. '13], or *binary relevance* [Boutell et al. '04]

Model	Binary Relevance (BR) [Boutell et al. '04]	Cond. Tree-struct. Bayesian Networks (CTBN) [Batal et al. '13]	Classifier Chains (CC) [Read et al. '09]
Graphical Notation (e.g., $d=4$)			
Structural Assumption	$\boldsymbol{\pi}(i,M) = \{\}$	$\boldsymbol{\pi}(i,M)$ = at most one parent label (tree)	$\boldsymbol{\pi}(i,M)$ = all preceding labels (chain)

- **Multi-Label Mixtures-of-Experts (ML-ME)**

(ML-ME1) Representation

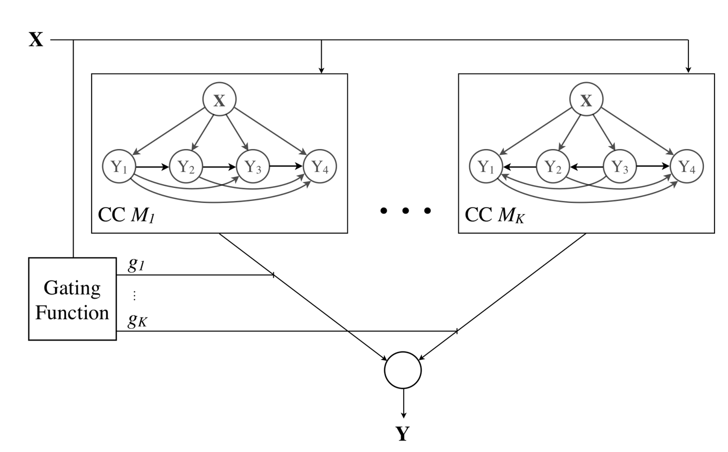
- ML-ME defines the multivariate posterior distribution of class vector $\mathbf{y} = (y_1, ..., y_d)$ by combining multiple MLC models that belong to *classifier chains family* (CCF):

$$P(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K} g_k(\mathbf{x}) P(\mathbf{y}|\mathbf{x}, M_k)$$
$$= \sum_{k=1}^{K} g_k(\mathbf{x}) \prod_{i=1}^{d} P(y_i|\mathbf{x}, \mathbf{y}_{\boldsymbol{\pi}(i,M_k)})$$

where $P(\mathbf{y}|\mathbf{x}, M_k) = \prod_{i=1}^{d} P(y_i|\mathbf{x}, \mathbf{y}_{\boldsymbol{\pi}(i,M_k)})$ is the joint conditional distribution defined by the k-th CCF model; and $g_k(\mathbf{x}) = P(M_k|\mathbf{x})$ is the gate reflecting how much M_k contributes towards prediction
  - We model the gate using the *Softmax* function

$$g_k(\mathbf{x}) = \frac{\exp(\boldsymbol{\theta}_{G_k}\mathbf{x})}{\sum_{k'=1}^{K}\exp(\boldsymbol{\theta}_{G_{k'}}\mathbf{x})}$$

- ML-ME can be graphically represented (e.g., d=4)



(ML-ME2) Parameter Learning

- By assuming the individual CCF structures are known and fixed, we derive an EM algorithm that optimizes the parameters of ML-ME
- Objective: Optimize the log-likelihood of the training data ($\boldsymbol{\Theta}$ denotes the model parameters; n denotes the instance index)

$$l(D;\boldsymbol{\Theta}) = \sum_{n=1}^{N} \log \sum_{k=1}^{K} g_k(\mathbf{x}^{(n)}) P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, M_k)$$

(ML-ME2) Parameter Learning (cont'd)

- By introducing a hidden variable $z^{(n)} \in \{1, ..., K\}$ for each instance $(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})$, rewrite the objective function as the *complete log-likelihood*:

$$l_c(D;\boldsymbol{\Theta}) = \sum_{n=1}^{N}\sum_{k=1}^{K} \mathbb{1}[z^{(n)}=k] \log\left(g_k(\mathbf{x}^{(n)}) P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, M_k)\right)$$

- Optimize the *complete log-likelihood* using *EM*
  - *E-step*: Compute the expectation of $z^{(n)}$

$$E\left[\mathbb{1}[z^{(n)}=k]\right] = \frac{g_k(\mathbf{x}^{(n)}) P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, M_k)}{\sum_{k'=1}^{K} g_{k'}(\mathbf{x}^{(n)}) P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, M_{k'})} = h_k^{(n)}$$

  - *M-step*: Learn the gate and CCF model parameters

$$\sum_{k=1}^{K}\sum_{n=1}^{N} h_k^{(n)} \log g_k(\mathbf{x}^{(n)}) + \sum_{k=1}^{K}\sum_{n=1}^{N} h_k^{(n)} \log P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, M_k)$$

Gate; optimize using the L2-regularized *L-BFGS** | Base models; train K CCF models using $h_k^{(n)}$ as instance weights

* *L-BFGS* [Liu & Nocedal '89]

(ML-ME3) Structure Learning

- To obtain useful structures for learning a mixture from data, we take a boosting-style approach
  - Add new CCF structures one by one to the mixture being trained; On each iteration, learn a structure by focusing on "hard" instances (the current mixture tends to misclassify)
  - Use the normalized prediction error as the instance weights ($\mathcal{M}$ denotes the current mixture):

$$\omega^{(n)} \propto 1 - P(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, \mathcal{M}), \quad s.t. \sum_{n=1}^{N} \omega^{(n)} = 1$$

  - Next CCF model optimizes the weighted conditional log-likelihood (WCLL) of data (refer our paper for details)

(ML-ME4) Prediction

- We search the space of class assignments by defining a Markov chain induced by local changes to individual class assignments
  - Our search is initialized using the MAP prediction from each CCF model in the mixture
  - The *annealed MAP* (*maximum a posteriori*) [Yuan et al. '04] approach is applied to speed up the search

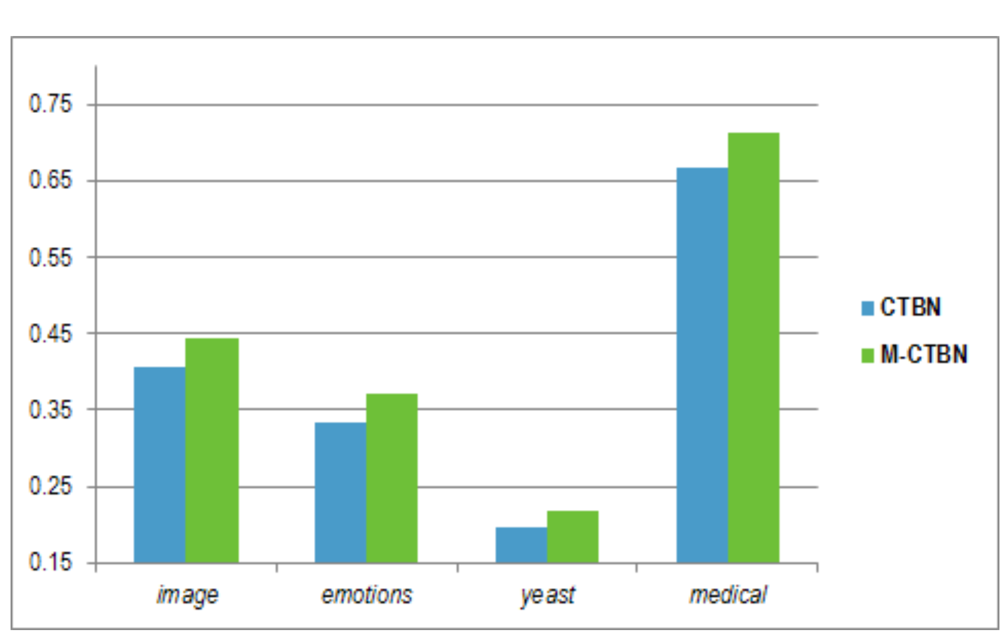
Experimental Results

- **Data**

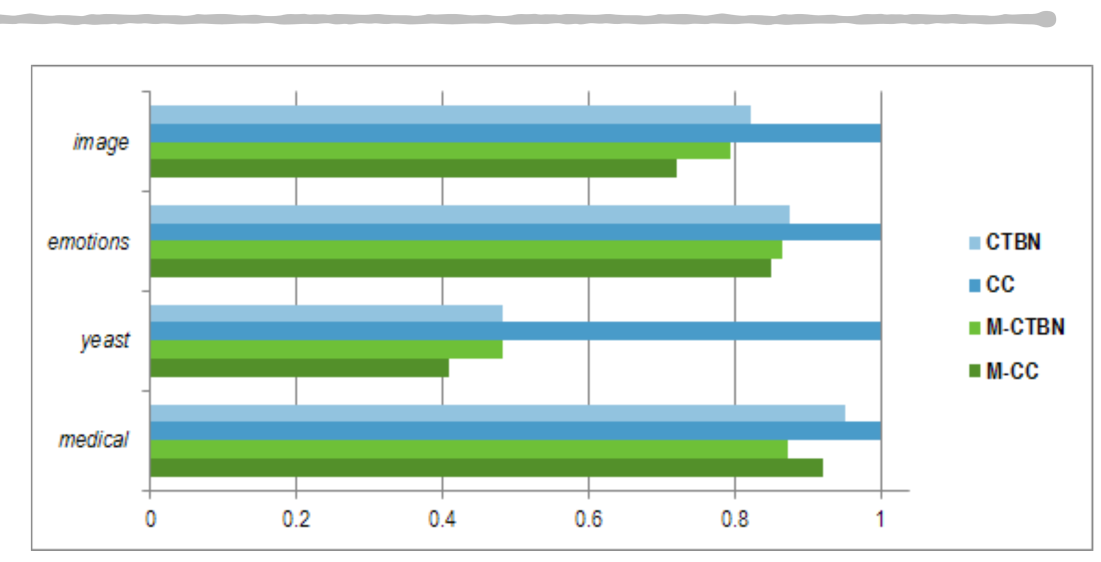
Dataset	N	m	d	LC	DLS	Domain
Image	2,000	135	5	1.24	20	image
Emotions	593	72	6	1.87	27	music
Yeast	2,417	103	14	4.24	198	biology
Medical	978	1,449	45	1.25	94	text

  - N: number of instances, m: number of features, d: number of classes, LC: label cardinality, DLS: distinct label set
  - Source: `http://mulan.sourceforge.net` and `http://cse.seu.edu.cn/people/zhangml/Resources.htm`
- **Methods**
  - CTBN [Batal et al. '13] vs. M-CTBN (our mixture)
  - CC/ECC [Read et al. '09], PCC/EPCC [Dembczynski et al. '10] vs. M-CC (our mixture)
- We perform *ten*-fold cross validation

- **Exact Match Accuracy (EMA)**
  - EMA measures the ratio of test instances whose prediction is exactly the same as their true class vector (higher is better)
  - CTBN vs. M-CTBN



- **Conditional Log-likelihood Loss (CLL-loss)**
  - CLL-loss measures the model fitness by evaluating how much probability mass is given to the true class vector (lower is better)



- **Exact Match Accuracy (EMA) (cont'd)**
  - CC, PCC, ECC, EPCC vs. M-CTBN