A Mixtures-of-Trees Framework for Multi-label Classification

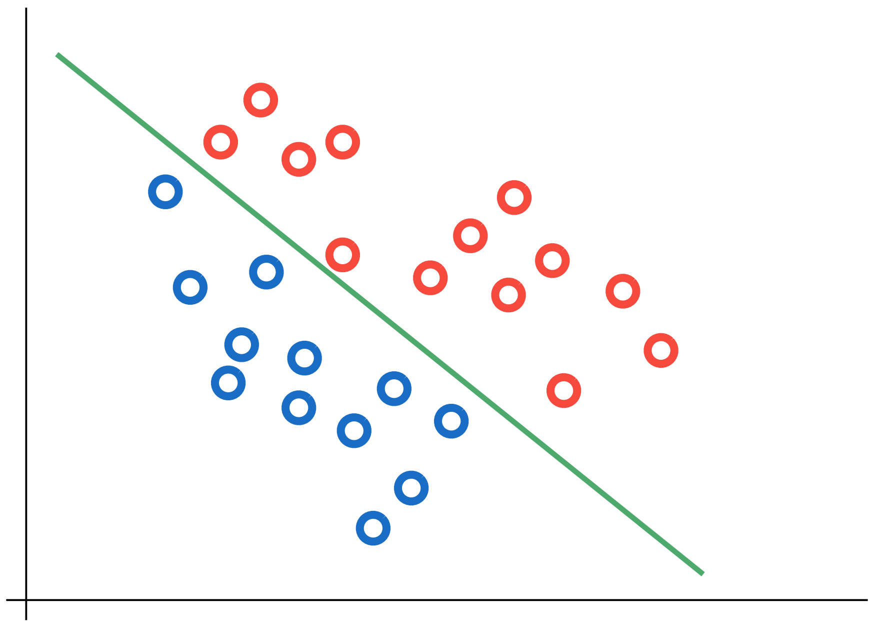
Charmgil Hong[✢] Iyad Batal[♦] Milos Hauskrecht[✢]

[✢] Department of Computer Science
University of Pittsburgh

[♦] GE Global Research

Introduction

- Traditional classification
  - Each data instance is associated with a single class variable

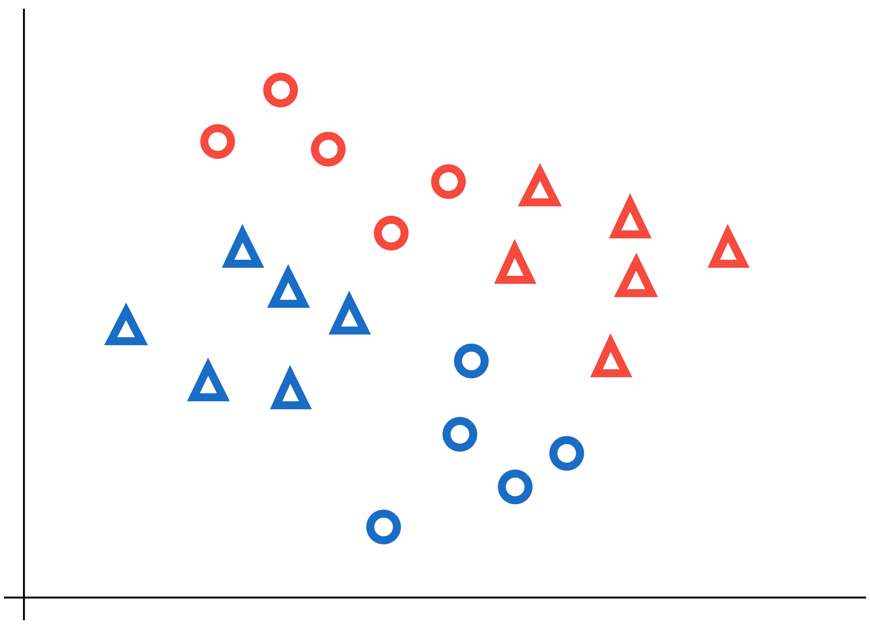


Introduction

- Multi-label classification (MLC)
  - In many real-world applications, each data instance can be associated with multiple class variables
  - Examples
    - A news article may cover multiple topics such as *politics* and *economy*
    - An image may include multiple objects as *building*, *road* and *car*
    - A gene may be associated with several biological functions

Introduction

- Multi-label classification (MLC)
  - Each data instance is associated with multiple binary class variables
  - Objective: assign to each instance the most probable assignment of the class variables



Class 1 ∈ { red, blue }
Class 2 ∈ { ○, △ }

Simplest MLC solution

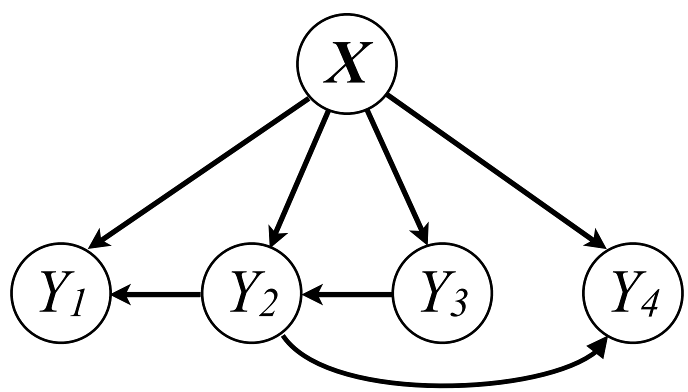
- Binary Relevance [Boutell et al., '04]
  - Learning d independent classifiers for d class labels
    - It does not capture the dependency relations between the classes

Baseline: CTBN [Batal et al., '13]

- Conditional Tree-structured Bayesian Network (CTBN)

Baseline: CTBN [Batal et al., '13]

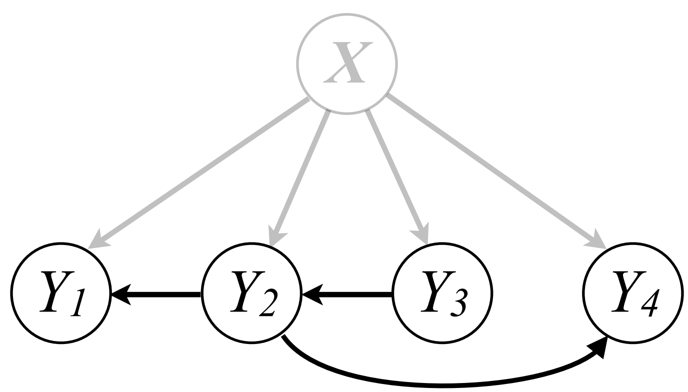
- Conditional Tree-structured Bayesian Network (CTBN) for modeling $\mathrm{P}(Y_1, \ldots, Y_d | \boldsymbol{X})$
  - A class variable can have at most one other class variable as a parent (the dependencies among classes form a directed tree)
  - The feature vector $\boldsymbol{X}$ is the common parent for all class variables



An example CTBN

Baseline: CTBN [Batal et al., '13]

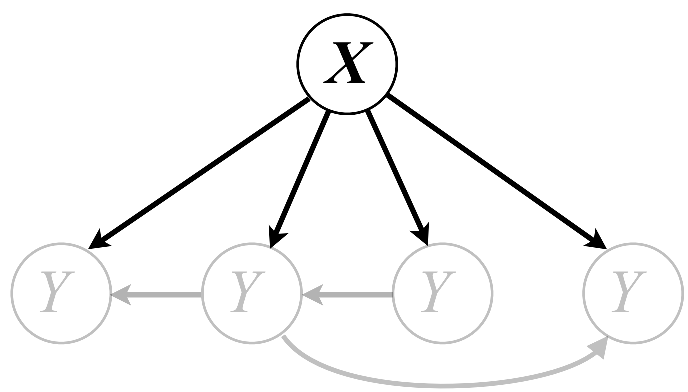
- Conditional Tree-structured Bayesian Network (CTBN) for modeling $\mathrm{P}(Y_1, \ldots, Y_d | \boldsymbol{X})$
  - A class variable can have at most one other class variable as a parent (the dependencies among classes form a directed tree)
  - The feature vector $\boldsymbol{X}$ is the common parent for all class variables



An example CTBN

Baseline: CTBN [Batal et al., '13]

- Conditional Tree-structured Bayesian Network (CTBN) for modeling $\mathrm{P}(Y_1, \ldots, Y_d | \boldsymbol{X})$
  - A class variable can have at most one other class variable as a parent (the dependencies among classes form a directed tree)
  - The feature vector $\boldsymbol{X}$ is the common parent for all class variables



An example CTBN

CTBN Representation

- The conditional class distribution is:

$$P(y_1, ..., y_d|\mathbf{x}) = \prod_{i=1}^{d} P(y_i|\mathbf{x}, y_{\pi(i,T)})$$

  where $y_{\pi(i,T)}$ denotes the parent of y_i in CTBN T

  - It is the product of the dependencies in the network
  - Each $P(y_i \mid \boldsymbol{x}, y_{\pi(i,T)})$ is represented by a classifier function (e.g. logistic regression)

CTBN Representation

- The conditional class distribution is:

$$P(y_1, ..., y_d|\mathbf{x}) = \prod_{i=1}^{d} P(y_i|\mathbf{x}, y_{\pi(i,T)})$$

where $y_{\pi(i,T)}$ denotes the parent of y_i in CTBN T

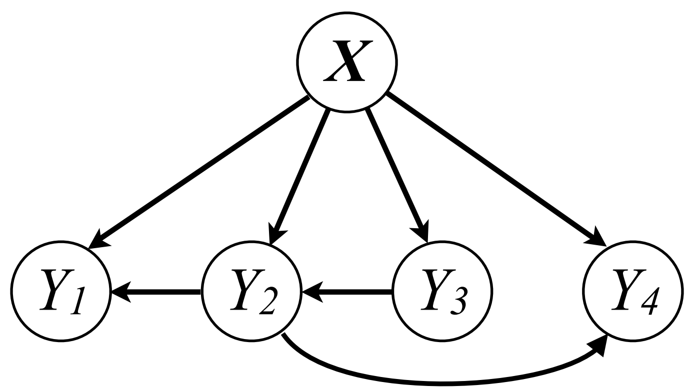
- It is the product of the dependencies in the network
- Each $P(y_i \,|\, \mathbf{x}, y_{\pi(i,T)})$ is represented by a classifier function (e.g. logistic regression)

CTBN Representation

- The conditional class distribution is:

$$P(y_1, \ldots, y_d|\mathbf{x}) = \prod_{i=1}^{d} P(y_i|\mathbf{x}, y_{\pi(i,T)})$$

where $y_{\pi(i,T)}$ denotes the parent of y_i in CTBN T



This network represents

$$P(y_1, y_2, y_3, y_4|\mathbf{x}) = P(y_3|\mathbf{x}) \cdot P(y_2|\mathbf{x}, y_3) \cdot P(y_1|\mathbf{x}, y_2) \cdot P(y_4|\mathbf{x}, y_2)$$

CTBN - Benefits and Limits

- Benefits
  - The optimal structure can be learned efficiently
  - Exact inference can be done in O(d) time

CTBN - Benefits and Limits

- Benefits
  - The optimal structure can be learned efficiently
  - Exact inference can be done in O(d) time
- Limits
  - The underlying dependency structure in data may be more complex than a tree structure
    - In such cases, a single CTBN cannot model the data properly

Goals in this work

- Goals
  1. To develop a more accurate probabilistic model for multi-label classification (MLC)
     - Use ensemble approach to improve the performance
  2. To devise supporting algorithms for efficient learning and prediction

Using Multiple CTBNs

- How to incorporate multiple MLC models?
  - Existing ensemble approaches for MLC [Read et al., '09,]
    - Fit multiple random structures to random subsets of data
    - Make predictions by the majority vote among the models
  - We use the *Mixtures-of-Trees* [Meila and Jordan, '00] approach
    - Learning and prediction become more principled

Mixtures-of-CTBNs (MC)

- MC defines the multivariate posterior distribution of class vector $P(\mathbf{y}|\mathbf{x}) = \mathrm{P}(y_1, \ldots, y_d|\mathbf{x})$ as

$$P(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K} \lambda_k P(\mathbf{y}|\mathbf{x}, T_k)$$
$$= \sum_{k=1}^{K} \lambda_k \prod_{i=1}^{d} P(y_i|\mathbf{x}, y_{\pi(i,T)})$$

  - $P(\boldsymbol{y}|\boldsymbol{x}, T_k)$ is the k-th mixture component defined by a CTBN T_k
  - λ_k is the mixture coefficient representing the weight of the k-th component (influence of the k-th CTBN model T_k to the mixture)

Mixtures-of-CTBNs (MC)

- MC defines the multivariate posterior distribution of class vector $P(\mathbf{y}|\mathbf{x}) = \mathrm{P}(y_1, \ldots, y_d|\mathbf{x})$ as

$$P(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K} \lambda_k P(\mathbf{y}|\mathbf{x}, T_k)$$

$$= \sum_{k=1}^{K} \lambda_k \prod_{i=1}^{d} P(y_i|\mathbf{x}, y_{\pi(i,T)})$$

  - $P(\mathbf{y}|\mathbf{x}, T_k)$ is the k-th mixture component defined by a CTBN T_k
  - λ_k is the mixture coefficient representing the weight of the k-th component (influence of the k-th CTBN model T_k to the mixture)

Mixtures-of-CTBNs (MC)

- MC defines the multivariate posterior distribution of class vector $P(\mathbf{y}|\mathbf{x}) = \mathrm{P}(y_1, \ldots, y_d|\mathbf{x})$ as

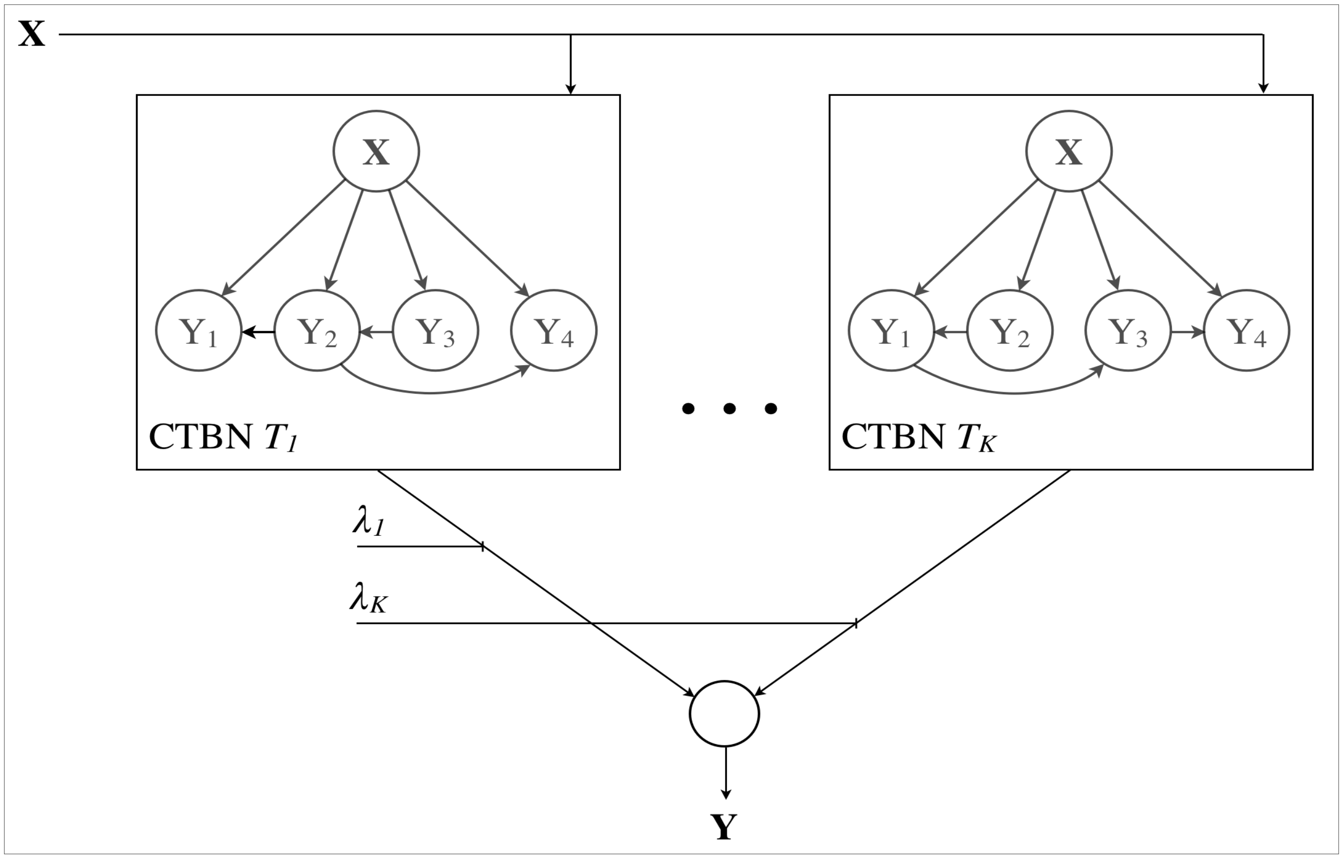
$$P(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K} \lambda_k P(\mathbf{y}|\mathbf{x}, T_k)$$

$$= \sum_{k=1}^{K} \lambda_k \prod_{i=1}^{d} P(y_i|\mathbf{x}, y_{\pi(i,T)})$$

- $P(\boldsymbol{y}|\boldsymbol{x}, T_k)$ is the k-th mixture component defined by a CTBN T_k
- λ_k is the mixture coefficient representing the weight of the k-th component (influence of the k-th CTBN model T_k to the mixture)

Mixtures-of-CTBNs (MC)

- An example MC



$$P(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K} \lambda_k P(\mathbf{y}|\mathbf{x}, T_k)$$

Mixtures-of-CTBNs (MC)

- We present the following algorithms for MC
  - *Parameter learning algorithm*: Learns the parameters of MC using expectation maximization (EM)
  - *Structure learning algorithm*: Learns multiple CTBN structures from data
  - *Prediction algorithm*: Finds the maximum a posteriori (MAP) assignment of class variables

Mixtures-of-CTBNs (MC)

- Parameter learning
  - Objective: Optimize the model parameters (CTBN parameters $\{\theta_1, \ldots, \theta_K\}$ and mixture coefficients $\{\lambda_1, \ldots, \lambda_K\}$)
  - Idea (apply EM)
    1. Associate each instance $(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})$ with a hidden variable $z^{(n)} \in \{1, \ldots, K\}$ indicating which CTBN it belongs to.
    2. Iteratively optimize the *expected complete log-likelihood*:

$$E\left[\sum_{n=1}^{N} \log P(\mathbf{y}^{(n)}, z^{(n)}|\mathbf{x}^{(n)})\right]$$

$$= E\left[\sum_{n=1}^{N}\sum_{k=1}^{K} 1[z^{(n)} = k]\left[\log \lambda_k + \log P\left(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}, T_k\right)\right]\right]$$

Mixtures-of-CTBNs (MC)

- Structure learning
  - Objective: Find multiple CTBN structures from data
  - Idea
    1. On each addition of a new structure to the mixture, recalculate the weight of each data instance (ω) such that it represents the relative "hardness" of the instance
    2. Learn the best tree structure by optimizing the weighted conditional log-likelihood:

$$\sum_{n=1}^{N} \sum_{i=1}^{d} \omega^{(n)} \log P(y_i^{(n)} | \mathbf{x}^{(n)}, y_{\pi(i,T)}^{(n)})$$

Mixtures-of-CTBNs (MC)

- Prediction
  - Objective: Find the maximum a posteriori (MAP) prediction for a new instance $\mathbf{x}$
  - Idea
    1. Search the space of all class assignments by defining a Markov chain
    2. Use an annealed version of exploration procedure to speed up the search

Experiments

- Compared methods
  - *Binary Relevance* (BR) [Boutell et al., '04, Clare et al., '01]
  - *Multi-label k-nearest neighbor* (MLKNN) [Zhang and Zhou, '07]
  - *Instance-based logistic regression* (IBLR) [Cheng and Hüllermeier, '09]
  - *Classifier chains* (CC) [Read et al., '09]
  - *Ensemble of Classifier chains* (ECC) [Read et al., '09]
  - *Probabilistic Classifier chains* (PCC) [Dembczynski et al., '10]
  - *Ensemble of Probabilistic Classifier chains* (EPCC) [Dembczynski et al., '10]
  - *Multi-label Conditional Random Fields* (MLCRF) [Pakdaman et al., '14]
  - *Maximum margin output coding* (MMOC) [Zhang and Schneider, '12]
  - *Single CTBN* (SC) [Batal et al., '13]

Experiments

- Data
  - 10 publicly available datasets from different domains

Dataset	# Instances	# Features	# Classes	Domain
Emotions	593	72	6	Music
Yeast	2,417	103	14	Biology
Image	2,000	135	5	Image
Scene	2,407	294	6	Image
Enron	1,702	1,001	53	Text
RCV1_subset1	6,000	8,394	10	Text
RCV1_subset2	6,000	8,304	10	Text
RCV1_subset3	6,000	8,328	10	Text
RCV1_subset4	6,000	8,332	10	Text
RCV1_subset5	6,000	8,367	10	Text

Experiment Results

- *Exact Match Accuracy*

  The probability of all classes being predicted correctly (higher is better)

Experiment Results

- *Exact Match Accuracy*

  The probability of all classes being predicted correctly (higher is better)

Dataset	BR	MLKNN	IBLR	CC	ECC	PCC	EPCC	MLCRF	MMOC	SC	MC
Emotions	0.27	0.28	0.34	0.27	0.29	0.32	0.34	0.30	0.33	0.32	**0.35**
Yeast	0.15	0.18	0.20	0.19	0.20	0.23	0.22	0.18	0.22	0.19	**0.24**
Image	0.28	0.35	0.39	0.43	0.41	0.45	0.44	0.38	0.45	0.41	**0.46**
Scene	0.54	0.63	0.64	0.63	0.66	0.67	0.67	0.58	0.66	0.63	**0.68**
Enron	0.16	0.08	0.16	0.17	0.18	-	-	-	-	0.17	**0.19**
RCV1_subset1	0.33	0.21	0.28	0.43	0.41	0.43	0.42	0.34	Could	0.44	**0.46**
RCV1_subset2	0.44	0.29	0.42	0.52	0.51	0.52	0.52	0.48	not	0.53	**0.54**
RCV1_subset3	0.47	0.33	0.45	0.54	0.54	0.55	0.54	0.49	finish	0.56	**0.56**
RCV1_subset4	0.51	0.35	0.49	0.58	0.57	0.56	0.58	0.55	-	0.59	**0.59**
RCV1_subset5	0.44	0.28	0.41	0.50	0.49	0.52	0.51	0.46	-	0.54	**0.54**
#win-tie-loss	**10-0-0**	**10-0-0**	**9-1-0**	**10-0-0**	**9-1-0**	**4-5-0**	**5-4-0**	**9-0-0**	**0-4-0**	**5-5-0**	

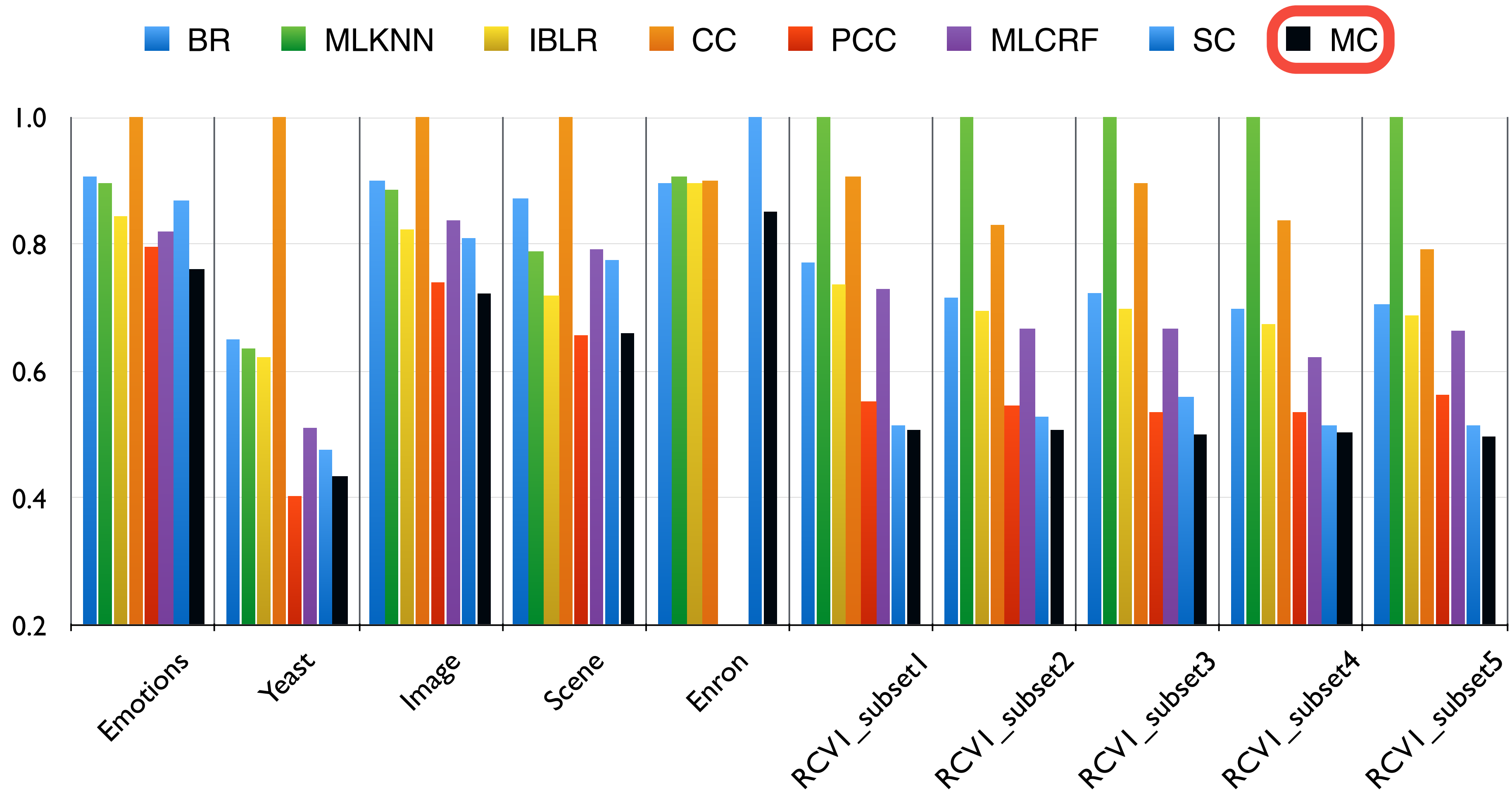
Experiment Results

- *Normalized conditional log-likelihood loss*
  Negative log-likelihood normalized on each dataset (lower is better)

Experiment Results

- *Normalized conditional log-likelihood loss*

  Negative log-likelihood normalized on each dataset (lower is better)



Conclusion

- We proposed the mixture of Conditional Tree-structured Bayesian Networks (MC) framework
  - Developed a probabilistic ensemble framework for multi-label classification
  - Presented efficient algorithms for parameter and structure learning
  - Presented a prediction algorithm that finds the MAP assignment of class variables for new instances
  - Demonstrated through experiments that our mixture framework outperforms several state-of-the-art multi-label classification methods

Epilogue

- Thank you very much for listening
- Our apologies to all for not being able to present in person
- For any questions or comments, please email me at: charmgil@cs.pitt.edu

Thank you!