Better Classifier Chains for Multi-label Classification

Jesse Read, Fernando Pérez Cruz

Department of Signal Theory and Communications
Universidad Carlos III
Madrid, Spain

July 21, 2011

Introduction

Multi-label classification is the supervised classification task where each data instance may be associated with *multiple* class labels.

Introduction

Multi-label classification is the supervised classification task where each data instance may be associated with *multiple* class labels. Given a predefined set of class-labels, e.g. $\mathcal{L} = \{\texttt{beach}, \texttt{trees}, \texttt{urban}, \texttt{people}\}$ and a set of instances from an input domain, e.g. :

- Multi-class (single-label) Classification: data instances are associated with a *single* class label; e.g. `beach`.
- Multi-label Classification: data instances are associated with a label *subset*: e.g. $\{\texttt{beach}, \texttt{trees}\}$.

Notation

- Instance $\mathbf{x} = [x_1, \ldots, x_d] \in \mathbb{R}^d$
- Class labels: $\mathcal{L} = \{1, 2, \ldots, L\}$
- Label space: $\mathcal{Y} = \{0, 1\}^L$
- Labelset: $\mathbf{y} = [y_1, \ldots, y_L] \in \mathcal{Y}$; $y_j = 1$ if jth label relevant to $\mathbf{x}$; else 0)
- Training set: $\{(\mathbf{x}_i, \mathbf{y}_i) | i = 1, \ldots, N\} \subset (\mathcal{X} \times \mathcal{Y})$
- Classification: $h : \mathcal{X} \rightarrow \mathcal{Y}$
- Prediction: $\hat{\mathbf{y}} = h(\mathbf{x})$

Notation

- Instance $\mathbf{x} = [x_1, \ldots, x_d] \in \mathbb{R}^d$
- Class labels: $\mathcal{L} = \{1, 2, \ldots, L\}$
- Label space: $\mathcal{Y} = \{0, 1\}^L$
- Labelset: $\mathbf{y} = [y_1, \ldots, y_L] \in \mathcal{Y}$; $y_j = 1$ if jth label relevant to $\mathbf{x}$; else 0)
- Training set: $\{(\mathbf{x}_i, \mathbf{y}_i) | i = 1, \ldots, N\} \subset (\mathcal{X} \times \mathcal{Y})$
- Classification: $h : \mathcal{X} \rightarrow \mathcal{Y}$
- Prediction: $\hat{\mathbf{y}} = h(\mathbf{x})$
- Evaluation:
  - by example $\hat{\mathbf{y}}_i = \mathbf{y}_i$ (*labelset accuracy*); OR
  - by label $\hat{y}_{ij} = y_{ij}$ of each example i (*label accuracy*).

Datasets and Statistics

	N	L	$\frac{\sum_j \mathbf{y}}{L}$	$\frac{uniq.\mathbf{y}}{N}$	Type
Music	593	6	1.87	0.046	media
Scene	2407	6	1.07	0.006	media
Yeast	2417	14	4.24	0.082	biology
Genbase	661	27	1.25	0.048	biology
Medical	978	45	1.25	0.096	medical text
Slashdot	3782	22	1.18	0.041	news
Lang.Log	1460	75	1.18	0.208	forum
Enron	1702	53	3.38	0.442	e-mail
Reuters(avg)	6000	103	1.46	0.147	news
Ohsumed	13929	23	1.66	0.082	medical text
tmc2007	28596	22	2.16	0.047	text
Media Mill	43907	101	4.38	0.149	media
Bibtex	7395	159	2.40	0.386	text
IMDB	120919	28	2.00	0.037	text
del.icio.us*	16105	983	19.02	0.981	text

Challenges

Challenges:

- discovering and modelling label dependencies
- dimensionality (output space of 2^L instead of L)

Prior work:

- we know some labels tend to be related / correlated
- we build a classifier which models these correlations somehow

We now want to know:

- which labels exhibit dependencies?
- how strong are the dependencies?

Label Dependence

- Label *independence* if:

$$p(\mathbf{Y}) = \prod_{j=1}^{L} p(Y_j)$$

  This should never be the case!

- Label *dependence*:
  - Unconditional dependence: $P(y_1|y_2)$
  - Conditional dependence: $P(y_1|y_2, \mathbf{x})$

Unconditional dependence

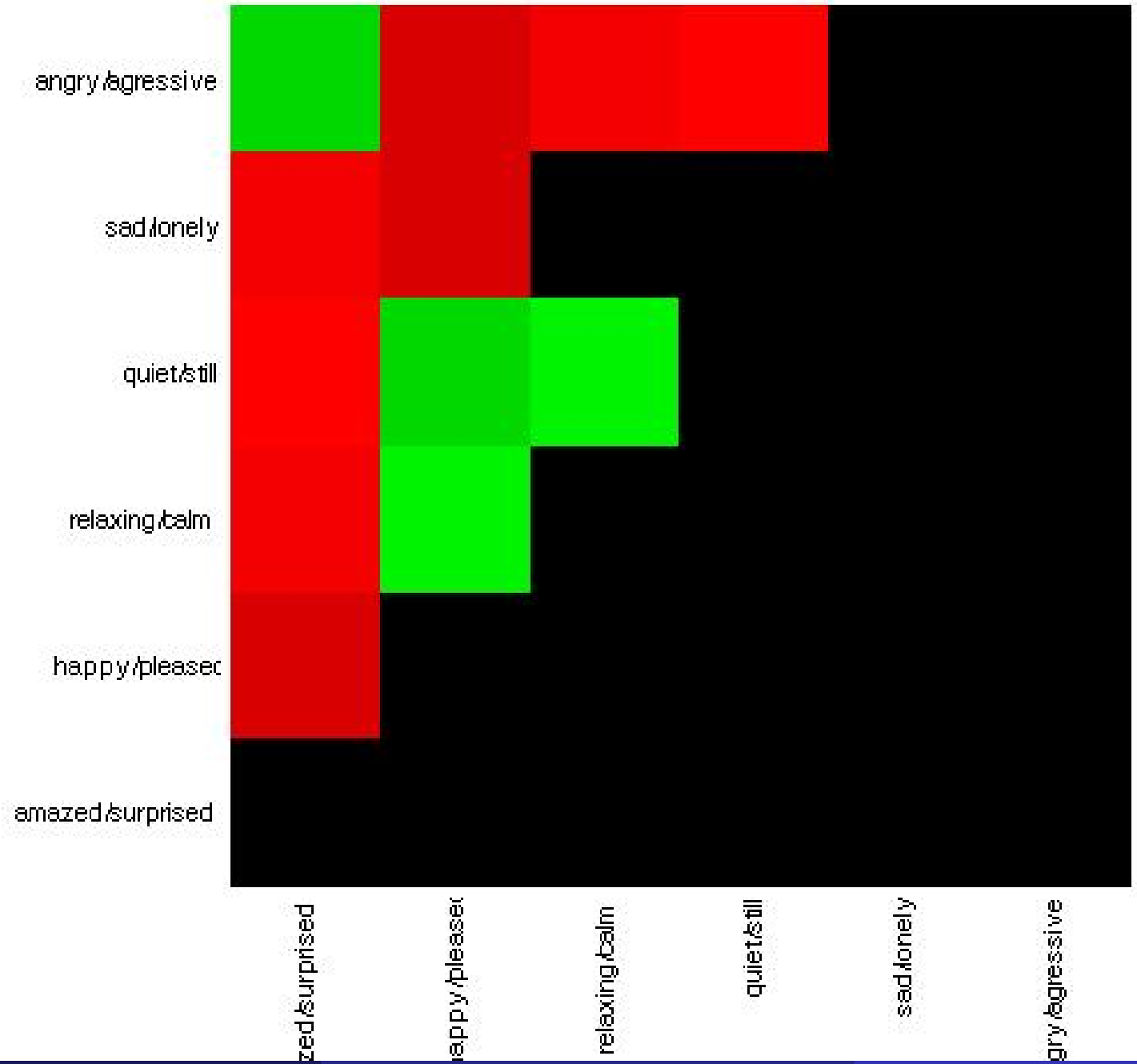
- Can measure with e.g. *Mutual information*,

$$I(Y_j; Y_k) = \sum_{y_j \in \{0,1\}} \sum_{y_k \in \{0,1\}} p(y_j, y_k) \log \left(\frac{p(y_j, y_k)}{p_1(y_j) p_2(y_k)} \right)$$

- or e.g. *Pearson's Correlation Coefficient*

$$P_{Y_j, Y_k} = \frac{\mathrm{cov}(Y_j, Y_k)}{\sigma_{Y_j} \sigma_{Y_k}}$$

Music with Emotions Dataset - Pearson' Correlation Coefficient



Conditional dependence

- Differential entropy[1]

$$I(y_j; y_k|\mathbf{x}) = I(y_j; y_k) - h(\mathbf{x}) - h(\mathbf{x}|y_j, y_k) + h(\mathbf{x}|y_j) + h(\mathbf{x}|y_k)$$

- Performance difference

$$Diff(Y_j, Y_k) = Eval(\mathtt{BR}, \mathbf{y}_j, \mathbf{y}_k, \mathbf{x}) - Eval(\mathtt{FW}, \mathbf{y}_j, \mathbf{y}_k, \mathbf{x})$$

where (for labels y_j and y_k), we compare:

- BR: Binary Relevance models: $y_j \in \{0, 1\}$, and $y_k \in \{0, 1\}$; and
- FW: Four-class pairWise models: $y_j, y_k \in \{00, 01, 10, 11\}$

[1] Fernando Pérez-Cruz, *Estimation of Information Theoretic Measures for Continuous Random Variables*. NIPS 2008

Conditional dependence

- Differential entropy[1]

$$I(y_j; y_k|\mathbf{x}) = I(y_j; y_k) - h(\mathbf{x}) - h(\mathbf{x}|y_j, y_k) + h(\mathbf{x}|y_j) + h(\mathbf{x}|y_k)$$

- Performance difference

$$Diff(Y_j, Y_k) = Eval(\texttt{BR}, \mathbf{y}_j, \mathbf{y}_k, \mathbf{x}) - Eval(\texttt{FW}, \mathbf{y}_j, \mathbf{y}_k, \mathbf{x})$$

where (for labels y_j and y_k), we compare:

- BR: Binary Relevance models: $y_j \in \{0, 1\}$, and $y_k \in \{0, 1\}$; and
- FW: Four-class pairWise models: $y_j, y_k \in \{00, 01, 10, 11\}$

(!) In practice, on real data, conditional dependence can be difficult to measure (N too small; $|\mathcal{X}|, L$ too large)

[1] Fernando Pérez-Cruz, *Estimation of Information Theoretic Measures for Continuous Random Variables*. NIPS 2008

Table: Synthetic data with strong conditional dependence and independence.

	Conditional Dependence		Conditional Independence	
	FW	BR	FW	BR
Subset Accuracy	**0.77**	0.70	0.84	**0.89**
Labelset Accuracy	**0.45**	0.38	0.59	**0.61**
Label Accuracy	**0.94**	0.92	0.97	**0.98**

- If label *dependence*, best to model it (e.g. FW)
- If label *independence*, best *not* to model any explicitly (e.g. BR)
- modelling very weak/non-existent label dependencies can be detrimental (and, it's computationally expensive: $L(L-1)/2$-pairwise (FW) compared to L-relevance (BR))

Conditional vs. Unconditional Dependence

Conditional (15×2 CV $Diff(Y_j, Y_k)$) and unconditional ($I(Y_j; Y_k)$) dependence and independence.

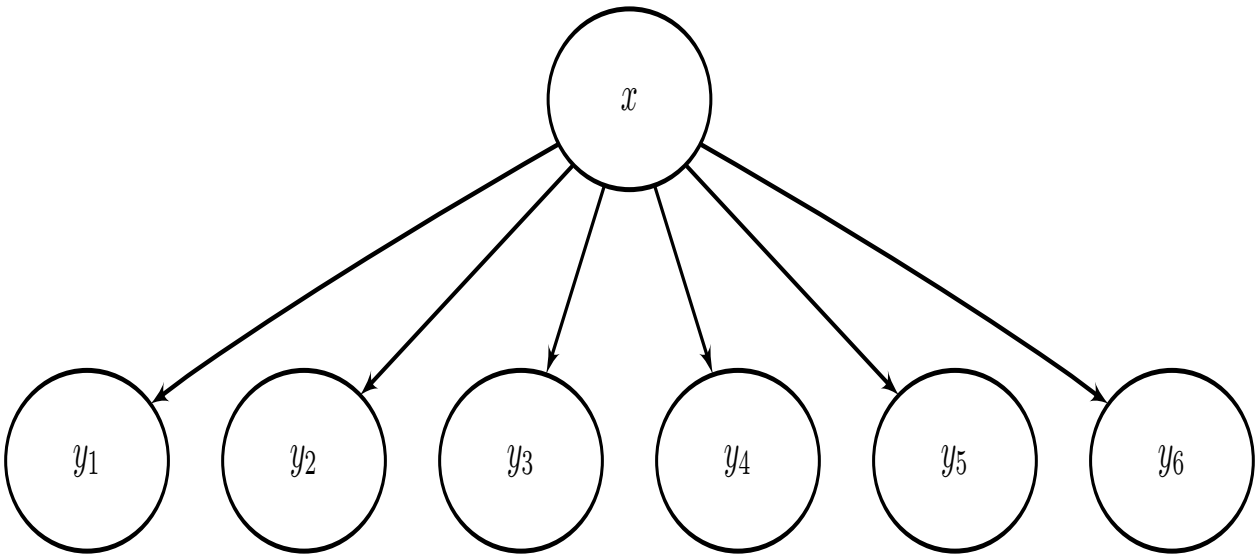
label pair ($\{y_j, y_k\}$)	conditional.	uncond.
{amazed-suprised, happy-pleased}	-0.565 ± 0.33	0.000
{amazed-suprised, relaxing-clam}	3.806 ± 1.028	0.476
{amazed-suprised, quiet-still}	-0.422 ± 0.020	0.590
{amazed-suprised, sad-lonely}	0.566 ± 0.742	0.362
{amazed-suprised, angry-aggresive}	-2.258 ± 2.071	0.126
{happy-pleased, relaxing-clam}	2.26 ± 5.283	0.028
{happy-pleased, quiet-still}	2.534 ± 0.078	0.225
{happy-pleased, sad-lonely}	3.512 ± 10.79	0.427
{happy-pleased, angry-aggresive}	1.685 ± 2.939	0.369
{relaxing-clam, quiet-still}	0.986 ± 0.021	0.455
{relaxing-clam, sad-lonely}	-0.281 ± 0.082	0.190
{relaxing-clam, angry-aggresive}	1.554 ± 3.485	0.800
{quiet-still, sad-lonely}	0.425 ± 0.515	0.547
{quiet-still, angry-aggresive}	1.276 ± 9.072	0.395
{sad-lonely, angry-aggresive}	1.13 ± 1.321	0.215

- We can just rely on unconditional dependence

Binary Relevance (BR)

One binary classifier for each label

- $\hat{y}_j = h_j(\mathbf{x})$ for each label $j = 1, \ldots, L$
- e.g. $P(\text{quiet-still} \mid \mathbf{x})$
- $\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_L]$

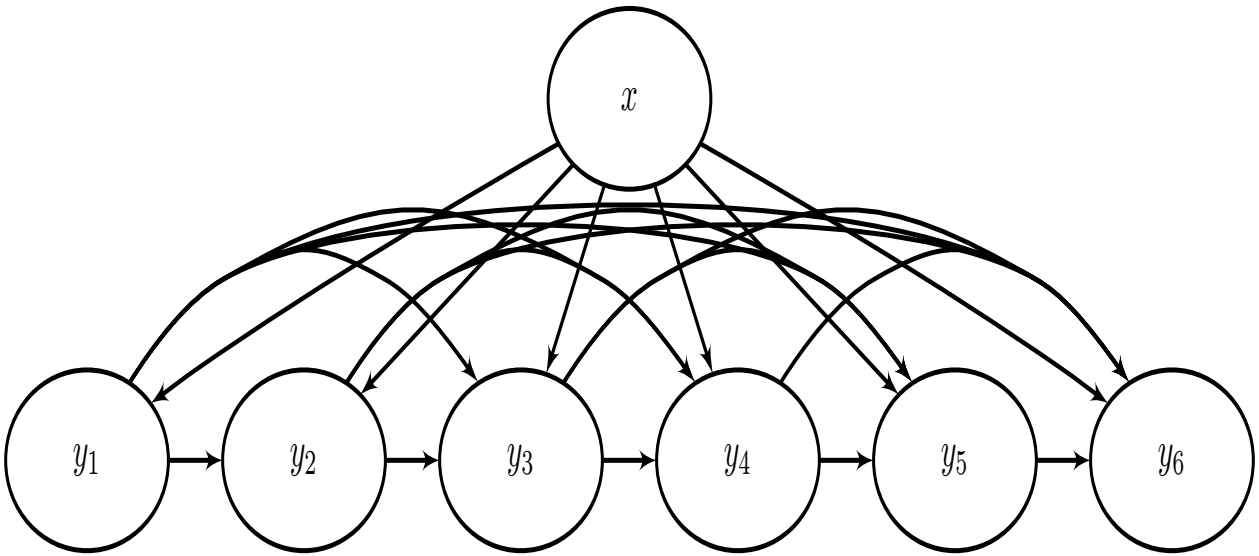


- flexible, good time complexity (L binary models); but
- does not explicitly model label correlations (poor prediction).

Classifier Chains (CC)

Pass information along a 'chain' of binary classifiers

- $\hat{y}_j = h_j(\mathbf{x}, \hat{y}_1, \ldots, \hat{y}_{j-1})$
- e.g. $P(\text{quiet-still} \mid \mathbf{x}, \neg\text{angry-agressive}, \text{sad-lonely})$
- $\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_L]$
- Ensembles of *random* Classifier Chains (ECC)[2]



- improved prediction; almost as fast as BR (when $L \ll |\mathcal{X}|$); but
- memory issues (especially with ECC) when L is large

[2]Read, Pfahringer, Holmes, and Frank. *Classifier Chains for Multi-label Classification*. Machine Learning Journal. Springer, 2011.

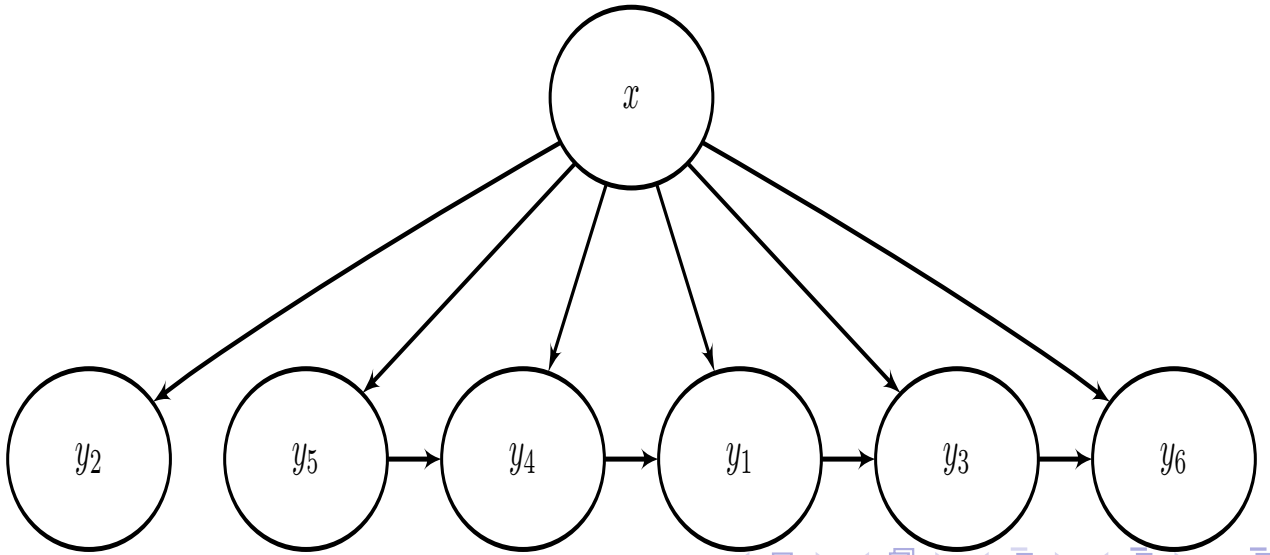
Forming better chains

1. Get $I(Y_j, Y_k)$ for each pair:

$$\begin{bmatrix} 0 & 0.01 & 0.65 & 0.58 & 0.33 & 0.2 \\ 0 & 0 & 0.05 & 0.3 & 0.52 & 0.33 \\ 0 & 0 & 0 & 0.24 & 0.06 & 0.85 \\ 0 & 0 & 0 & 0 & 0.83 & 0.56 \\ 0 & 0 & 0 & 0 & 0 & 0.28 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

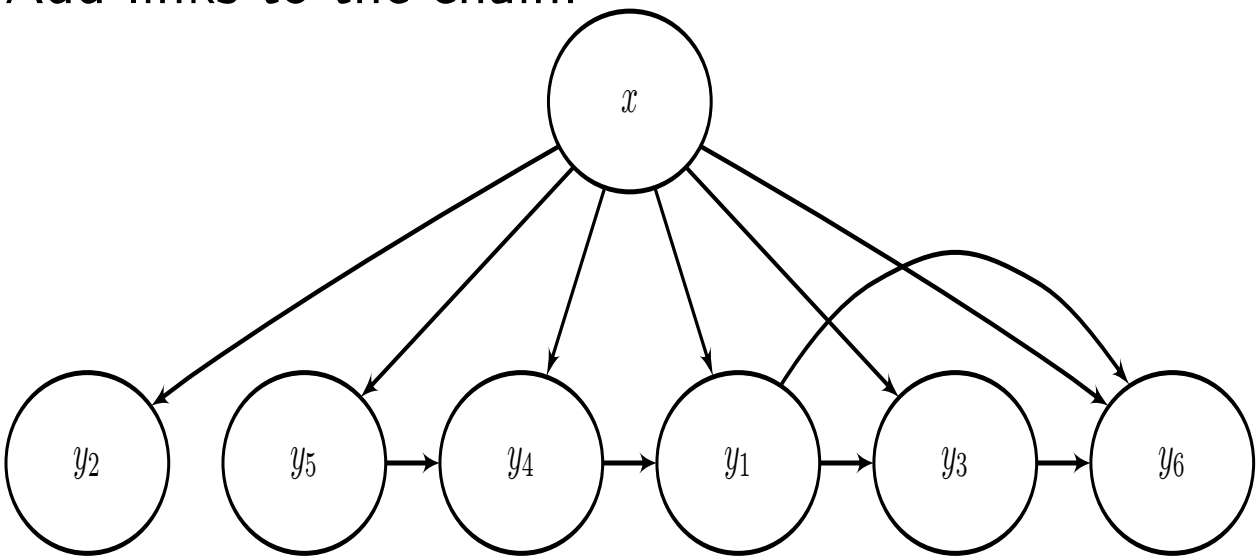
2. Form high-information edges: $\{(y_3, y_6), (y_4, y_5), (y_1, y_3), (y_1, y_4), ...\}$.

3. Build a chain:

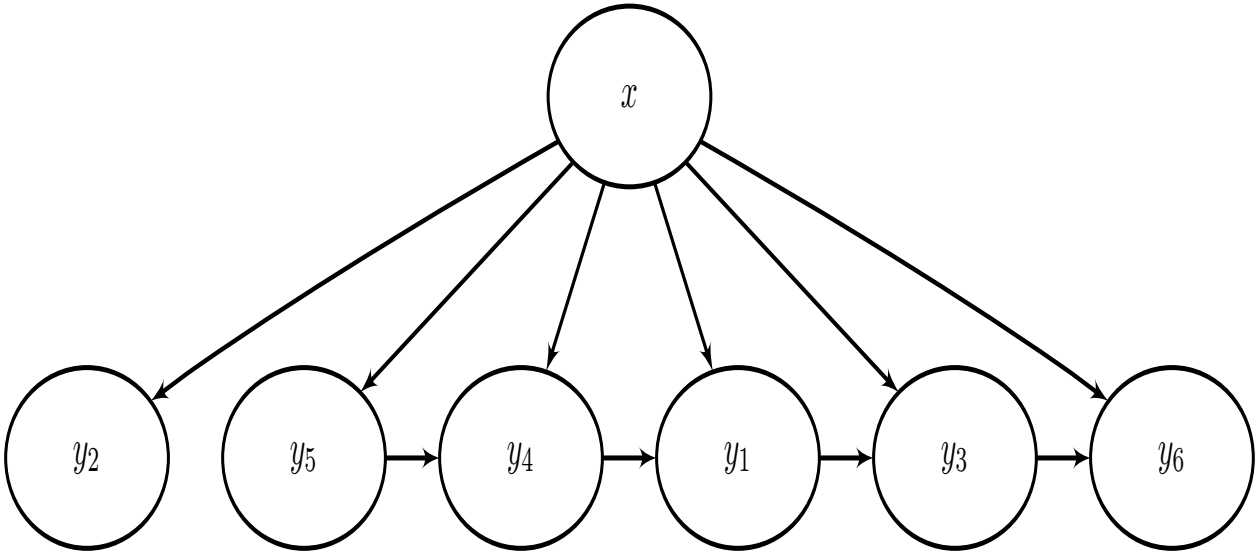


(future work)

4. Find conditional mutual information:
   $I(y_1; y_5|y_6) = 0.06$
   $I(y_2; y_4|y_3) = 0.10$
   $I(y_2; y_6|y_1) = 0.11$
   $I(y_3; y_6|y_1) = 0.23$
   $I(y_3; y_6|y_5) = 0.26$
   $I(y_4; y_5|y_6) = 0.14$
   $I(y_4; y_6|y_3) = 0.09$
   $I(y_4; y_6|y_5) = 0.07$
5. Add links to the chain:

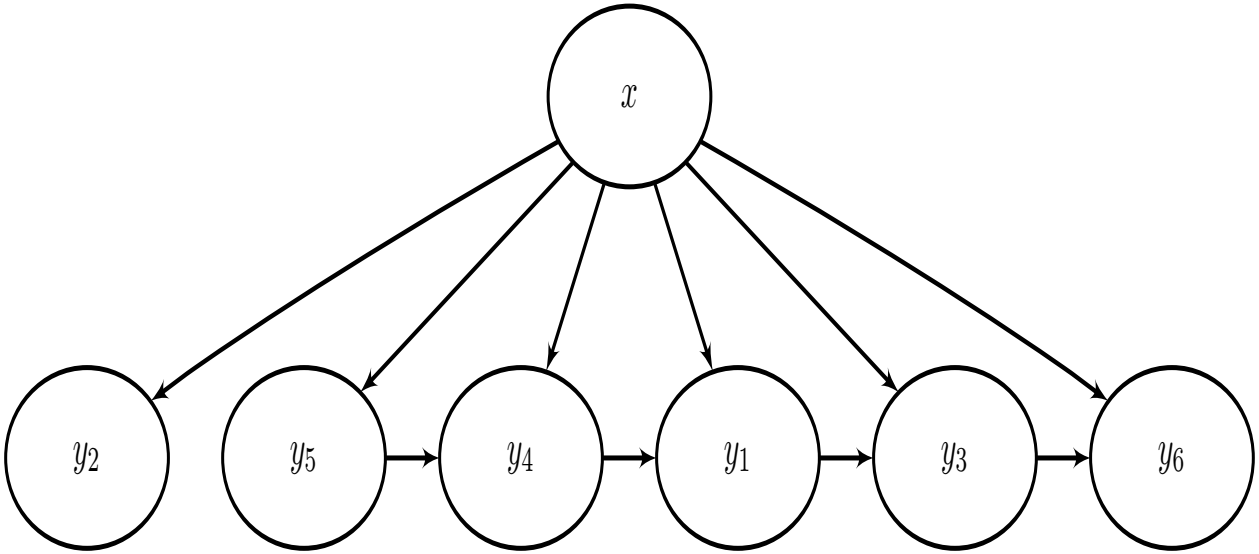


Methods



- model BR ($p(y_j|\mathbf{x})$) on unconnected nodes
- CC_2: models $p(y_j|y_{j-1}, \mathbf{x})$ at each jth connected *node*
- CC_v: models $p(y_{j-1}, y_j|\mathbf{x})$ at each *edge*, with the viterbi algorithm

Methods



- model BR ($p(y_j|\mathbf{x})$) on unconnected nodes
- CC_2: models $p(y_j|y_{j-1}, \mathbf{x})$ at each jth connected *node*
- CC_v: models $p(y_{j-1}, y_j|\mathbf{x})$ at each *edge*, with the viterbi algorithm

Method	Classifiers	Class space	Instances/Classifier
BR	L	$\{0, 1\}$	$\|\mathcal{X}\|$
CC	L	$\{0, 1\}$	$\|\mathcal{X}\| + \frac{L-1}{2}$
CC_2	L	$\{0, 1\}$	$\|\mathcal{X}\| + 1$
CC_v	$L - 1$	$\{00, 01, 10, 11\}$	$\|\mathcal{X}\|$
FW	$\frac{L(L-1)}{2}$	$\{00, 01, 10, 11\}$	$\|\mathcal{X}\|$

Results

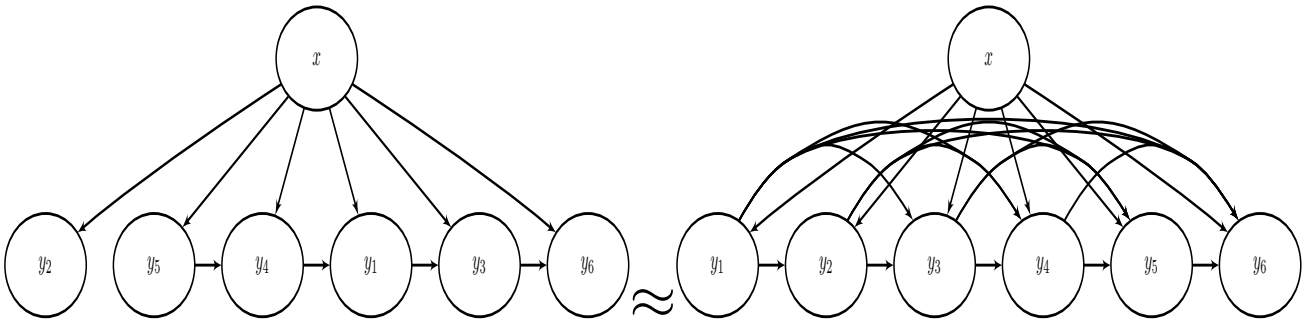
Dataset	BR	CC	CC_2	CC_v	FW
Music	0.479 (5)	0.508 (4)	0.577 (3)	0.590 (2)	0.598 (1)
Scene	0.571 (5)	0.674 (1)	0.639 (3)	0.643 (2)	0.634 (4)
Yeast	0.502 (5)	0.534 (2)	0.524 (3)	0.518 (4)	0.543 (1)
Genbase	0.991 (4)	0.991 (4)	0.993 (1)	0.993 (1)	0.993 (1)
Medical	0.753 (5)	0.777 (1)	0.773 (3)	0.764 (4)	0.774 (2)
Slashdot	0.402 (5)	0.432 (2)	0.434 (1)	0.409 (4)	0.422 (3)
Enron	0.299 (5)	0.300 (4)	0.307 (3)	0.315 (1)	0.307 (2)
Lang.Log	0.083 (5)	0.087 (2)	0.086 (4)	0.086 (3)	0.105 (1)
Reuters	0.325 (5)	0.379 (2)	0.391 (1)	0.331 (4)	0.378 (3)
avg. rank	4.89	2.44	2.44	2.78	2.00
avg. value	0.49	0.52	0.53	0.52	0.53

Nemenyi (rank-based) significance: CC$\succ$BR; $CC_2\succ$BR; $CC_v\succ$BR; FW$\succ$BR;

- gains on *Music*, losses on *Scene*
- overall no significant difference (except to baseline BR)
- similar results under other measures of evaluation
- FW is approx. 30 times slower than other methods
- Ensembling methods increases the accuracy of all evenly

Summary / Conclusions

- modelling label dependence improves prediction
- best to model dependencies only where they exist / are strongest
- conditional label dependence can be difficult to model in the multilabel context (so we just model unconditional dependence)
- results so far not amazing; but promising



The End

Questions?