Multi-target Prediction with Classifier Chains

Jesse Read

http://users.ics.aalto.fi/jesse/

Aalto University School of Science,
Department of Information and Computer Science
and Helsinki Institute for Information Technology
Helsinki, Finland

Nancy, France. Sep. 15, 2014

Multi-label and Multi-target Classification

Map D feature (input) variables to L target (output) variables.

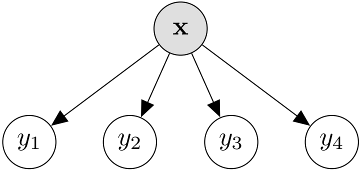
X_1	X_2	X_3	$\ldots$	X_D	Y_1	Y_2	Y_3	Y_4	Y_5
x_1	x_2	x_3	$\ldots$	x_D	0	0	0	4	5
x_1	x_2	x_3	$\ldots$	x_D	1	1	1	2	1
x_1	x_2	x_3	$\ldots$	x_D	0	0	0	2	2
x_1	x_2	x_3	$\ldots$	x_D	1	1	0	3	2
$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	$\ldots$	$\tilde{x}_D$	?	?	?	?	?

multi-label classification: all targets are binary variables, e.g., $\in \{0, 1\}$

Build model $\mathbf{h}$, such that $\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_L] = \mathbf{h}(\tilde{\mathbf{x}})$.

Binary Relevance (BR)

- With any off-the-shelf classifier, train L independent models $\mathbf{h} = (h_1, \ldots, h_L)$, one for each label,



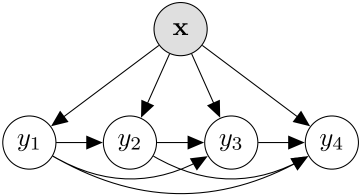
- For input $\tilde{\mathbf{x}}$, predict

$$\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_L] = [h_1(\tilde{\mathbf{x}}), \ldots, h_L(\tilde{\mathbf{x}})] = \mathbf{h}(\tilde{\mathbf{x}})$$

- General consensus in the literature: should model relationship between target variables

Classifier Chains (CC)

- Predictions are cascaded along a chain as additional features[1]:



- For any $\tilde{\mathbf{x}}$, predict

$$\hat{\mathbf{y}} = [\hat{y}_1, \ldots, \hat{y}_L] = [h_1(\tilde{\mathbf{x}}), h_2(\tilde{\mathbf{x}}, \hat{y}_1), \ldots, h_L(\tilde{\mathbf{x}}, \hat{y}_1, \ldots, \hat{y}_{L-1})] = \mathbf{h}(\tilde{\mathbf{x}})$$

[1] [Read et al., 2009], MLJ

Binary Relevance vs Classifier Chains

Table : Binary Relevance: Model h_3

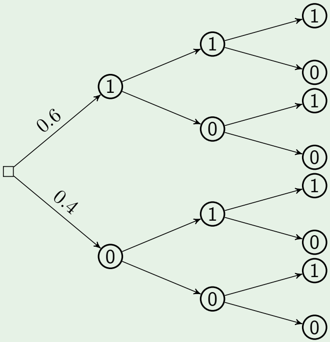
X_1	X_2	X_3	...	X_D	Y_1	Y_2	Y_3	Y_4
x_1	x_2	x_3	...	x_D	0	1	1	0
x_1	x_2	x_3	...	x_D	1	0	0	0
x_1	x_2	x_3	...	x_D	0	1	0	0
x_1	x_2	x_3	...	x_D	0	0	1	1
$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	...	$\tilde{x}_D$			?	

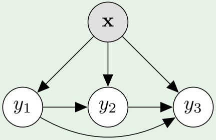
Table : Classifier Chains: Model h_3

X_1	X_2	X_3	...	X_D	Y_1	Y_2	Y_3	Y_4
x_1	x_2	x_3	...	x_D	0	1	1	0
x_1	x_2	x_3	...	x_D	1	0	0	0
x_1	x_2	x_3	...	x_D	0	1	0	0
x_1	x_2	x_3	...	x_D	0	0	1	1
$\tilde{x}_1$	$\tilde{x}_2$	$\tilde{x}_3$	...	$\tilde{x}_D$	$\hat{y}_1$	$\hat{y}_2$	?	

Classifier Chains

Example - Greedy Inference



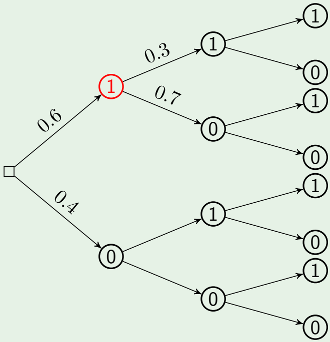


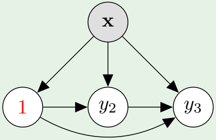
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \mathrm{argmax}_{y_1}\ p(y_1|\tilde{\mathbf{x}}) = ?$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, \hat{y}_1) = \mathrm{argmax}_{y_2}\ p(y_2|\tilde{\mathbf{x}}, 1) =$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, \hat{y}_1, \hat{y}_2) = \mathrm{argmax}_{y_3}\ p(y_3|\tilde{\mathbf{x}}, 1, 0) =$

$$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [?, ?, ?]$$

Classifier Chains

Example - Greedy Inference



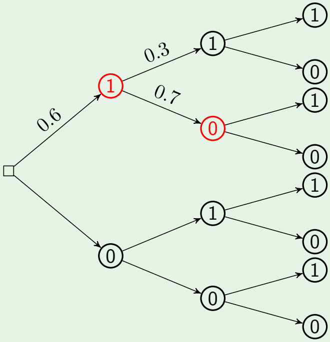


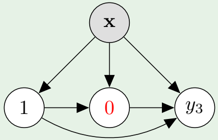
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \mathrm{argmax}_{y_1}\ p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, 1) = \mathrm{argmax}_{y_2}\ p(y_2|\tilde{\mathbf{x}}, 1) =$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, 1, \hat{y}_2) = \mathrm{argmax}_{y_3}\ p(y_3|\tilde{\mathbf{x}}, 1, 0) =$

$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, ?, ?]$

Classifier Chains

Example - Greedy Inference



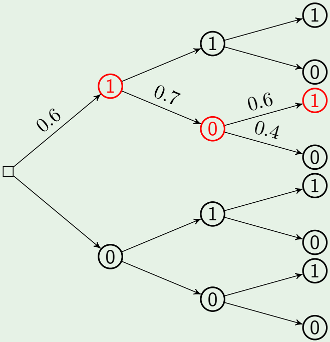


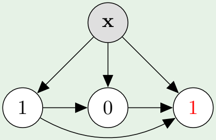
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \text{argmax}_{y_1}\ p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, 1) = \text{argmax}_{y_2}\ p(y_2|\tilde{\mathbf{x}}, 1) = 0$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, 1, 0) = \text{argmax}_{y_3}\ p(y_3|\tilde{\mathbf{x}}, 1, 0) =$

$$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, 0, ?]$$

Classifier Chains

Example - Greedy Inference



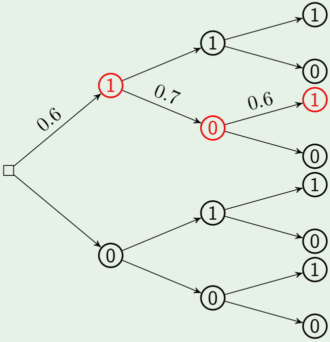


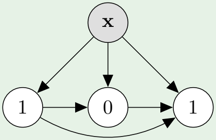
1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \text{argmax}_{y_1}\ p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, 1) = \text{argmax}_{y_2}\ p(y_2|\tilde{\mathbf{x}}, 1) = 0$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, 1, 0) = \text{argmax}_{y_3}\ p(y_3|\tilde{\mathbf{x}}, 1, 0) = 1$

$$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, 0, 1]$$

Classifier Chains

Example - Greedy Inference





1. $\hat{y}_1 = h_1(\tilde{\mathbf{x}}) = \text{argmax}_{y_1}\ p(y_1|\tilde{\mathbf{x}}) = 1$
2. $\hat{y}_2 = h_2(\tilde{\mathbf{x}}, 1) = \text{argmax}_{y_2}\ p(y_2|\tilde{\mathbf{x}}, 1) = 0$
3. $\hat{y}_3 = h_3(\tilde{\mathbf{x}}, 1, 0) = \text{argmax}_{y_3}\ p(y_3|\tilde{\mathbf{x}}, 1, 0) = 1$

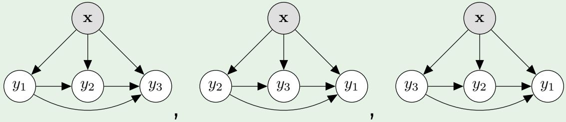
$\hat{\mathbf{y}} = \mathbf{h}(\tilde{\mathbf{x}}) = [1, 0, 1]$

- Better predictions than BR; similar build time (if $L < D$)
- but, errors may be propagated down the chain

Ensemble of Classifier Chains

1. Train M classifier chains, $\mathbf{h}^{(1)}, \ldots, \mathbf{h}^{(M)}$ with random label orders.
2. Ensemble voting

Example



	Y_1	Y_2	Y_3
$\mathbf{h}^{(1)}(\tilde{\mathbf{x}})$	1	0	1
$\mathbf{h}^{(2)}(\tilde{\mathbf{x}})$	1	0	0
$\mathbf{h}^{(3)}(\tilde{\mathbf{x}})$	0	0	1
$\hat{\mathbf{y}}$	1	0	1

- Improves predictive performance, but what about a single chain?

Probabilistic Classifier Chains

- Bayes-optimal inference[2] instead of greedy inference.

$$\hat{\mathbf{y}} = \cancel{[\operatorname*{argmax}_{y_1 \in \{0,1\}} p(y_1|\mathbf{x}), \cdots, \operatorname*{argmax}_{y_L \in \{0,1\}} p(y_L|\mathbf{x}, y_1, \ldots, y_{L-1})]} \quad \bullet \text{ greedy}$$

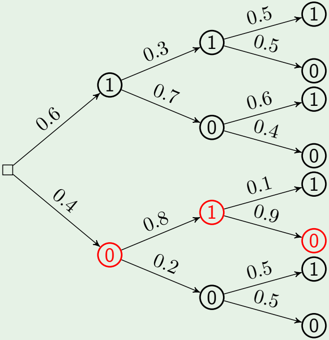
$$= \operatorname*{argmax}_{\mathbf{y} \in \{0,1\}^L} \left\{ p(y_1|\mathbf{x}) \prod_{j=2}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1}) \right\} \quad \bullet \text{ Bayes optimal}$$

[2] [Dembczyński et al., 2010], ICML'10

Probabilistic Classifier Chains

- Explore *all* (2^L) possible paths

Example - Bayes-optimal Inference



1. $p(\mathbf{y} = [0,0,0]) = 0.040$
2. $p(\mathbf{y} = [0,0,1]) = 0.040$
3. $p(\mathbf{y} = [0,1,0]) = 0.288$
4. $p(\mathbf{y} = [0,1,1]) = 0.032$
5. $p(\mathbf{y} = [1,0,0]) = 0.168$
6. $p(\mathbf{y} = [1,0,1]) = 0.252$
7. $p(\mathbf{y} = [1,1,0]) = 0.090$
8. $p(\mathbf{y} = [1,1,1]) = 0.090$

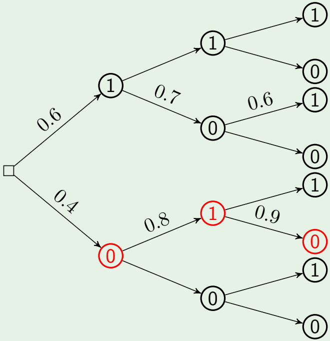
$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \{0,1\}^L}{\operatorname{argmax}}\, p(\mathbf{y}|\tilde{\mathbf{x}}) = [0,1,0]$$

- Better accuracy than CC, but only appropriate for $L \lesssim 15$

Monte-Carlo search for Classifier Chains

- Sampling for inference in CC[2] (instead of greedy / exhaustive)

Example - CC with Monte-Carlo Search



Sample $T = 2$ times ...

1. $p(\mathbf{y}_1 = [1, 0, 1]) = 0.252$
2. $p(\mathbf{y}_2 = [0, 1, 0]) = 0.288$

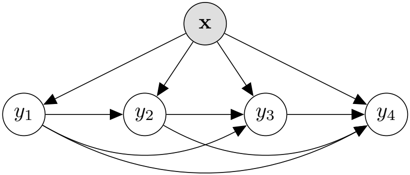
$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y} \in \{\mathbf{y}_t\}_{t=1}^{T}} p(\mathbf{y}_t | \tilde{\mathbf{x}}) = [0, 1, 0]$$

- Becomes tractable (for $T \ll 2^L$), but still $\succ$ CC.

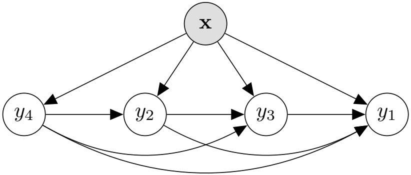
[2] e.g., [Dembczynski et al., 2012] ECAI, [Read et al., 2013], Pat. Reg. and related techinques, e.g., "Beam search" [Kumar et al., 2013]

Chain Order

Are these models equivalent?

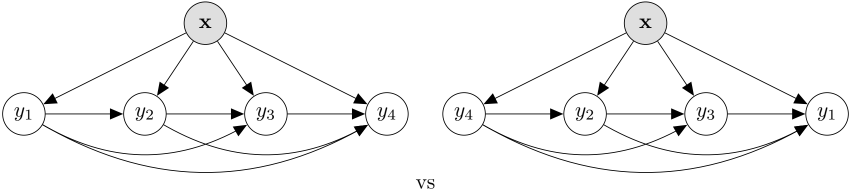


vs



Chain Order

Are these models equivalent?



Not necessarily. Although

$$p(y_2|\mathbf{x})p(y_1|y_2,\mathbf{x}) = p(y_1|\mathbf{x})p(y_2|y_1,\mathbf{x})$$

we *estimate* p from finite and noisy data; so

$$\hat{p}(y_2|\mathbf{x})\hat{p}(y_1|\hat{y}_2,\mathbf{x}) \neq \hat{p}(y_1|\mathbf{x})\hat{p}(y_2|\hat{y}_1,\mathbf{x})$$

How to Order the Chain?

1. Use the 'default' chain
2. Use several random chains in ensemble
3. Search the chain space (try several chains) at training time[3], and
   - use the best one;
   - use several (possibly just one per test instance); or
   - weighted average.

Empirical results: **good accuracy**, but **expensive**.

[3] e.g., [Read et al., 2014] Pat. Reg., [Kumar et al., 2013] Mach. Learn., [Li and Zhou, 2013], MCS '13

How to Order the Chain?

1. Use the 'default' chain
2. Use several random chains in ensemble
3. Search the chain space (try several chains) at training time[3], and
   - use the best one;
   - use several (possibly just one per test instance); or
   - weighted average.

   Empirical results: **good accuracy**, but **expensive**.
4. Order the chain according to some heuristic, e.g.,
   - by difficulty / model accuracy: easiest labels first
   - by label dependence ...

[3] e.g., [Read et al., 2014] Pat. Reg., [Kumar et al., 2013] Mach. Learn., [Li and Zhou, 2013], MCS '13

How to Order the Chain?

1. Use the 'default' chain
2. Use several random chains in ensemble
3. Search the chain space (try several chains) at training time[3], and
   - use the best one;
   - use several (possibly just one per test instance); or
   - weighted average.

   Empirical results: **good accuracy**, but **expensive**.
4. Order the chain according to some heuristic, e.g.,
   - by difficulty / model accuracy: easiest labels first
   - by label dependence ...

...why a *chain* (cascade)?

[3] e.g., [Read et al., 2014] Pat. Reg., [Kumar et al., 2013] Mach. Learn., [Li and Zhou, 2013], MCS '13

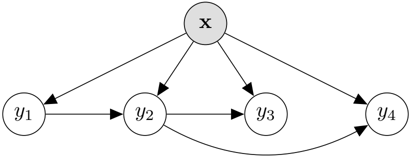
From a Chain to a Tree

Can formulate any structure,

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} \prod_{j=2}^{L} p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1}) \quad \bullet \text{ chain}$$

$$\approx \operatorname*{argmax}_{\mathbf{y}} \prod_{j=1}^{L} p(y_j|\mathbf{x}, \mathbf{pa}j) \quad \bullet \text{ directed graph}$$

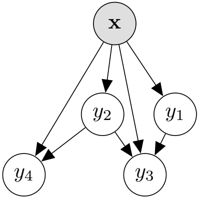
where $\mathbf{pa}j$ = parents of node j.



- 'Plug in' any CC (e.g., greedy inference)
- Benefits wrt train/test time, interpretability, but
- **how to find a good structure?**

Classifier Directed Graphs

1. Measure
   - marginal dependence, i.e., dependence among $Y_1, \ldots, Y_L$[4]; or
   - conditional dependence, e.g., dependence among *errors* $\epsilon_1, \ldots, \epsilon_L$[5]
2. Create a directed graph (there are many existing methods)



3. Plug in CC (or use standard message passing algorithms[6] – *complexity permitting*).

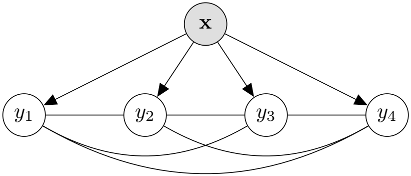
[4] e.g., [Zaragoza et al., 2011], IJCAI

[5] as in 'LEArning with label Dependence' [Zhang and Zhang, 2010], KDD '10

[6] [Alessandro et al., 2013] IJCAI "Ensemble of Bayes Nets for MLC"

Undirected Graph

- Graph can be undirected, becomes like conditional random fields[7]



- chain 'order' no longer an issue; but
- greedy inference no-longer possible. Can use, e.g., Gibbs sampling,
  1. sample many times $y_j \sim p(y_j|\mathbf{x}, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_L)$
  2. collect marginals.
- reduced structure usually necessary

[7] e.g., [Guo and Gu, 2011] IJCAI "Conditional Dependency Networks for MLC"; [Dembszyński et al., 2011] CoLISD

So far

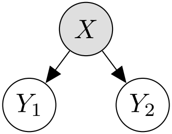
Classifier Chains (and Trees, Graphs, etc.):

1. Measure label dependence
2. Choose a structure for labels
   - more interpretable
   - more efficient
3. Choose an inference procedure

But, empirical results: *accuracy no better than random structures*. Why?

Classifier Chains: advantages beyond label dependence

Suppose we know that, *given the input*, the labels are independent,



$$\mathbb{E}(Y_2|Y_1, X) = \mathbb{E}(Y_2|X)$$

Independent classifiers (BR) will work as well here as classifier chains (CC)? . . . Not always!

Example: The XOR Problem

Toy problem,

		OR	AND	XOR
X_1	X_2	Y_1	Y_2	Y_3
0	0	0	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	0

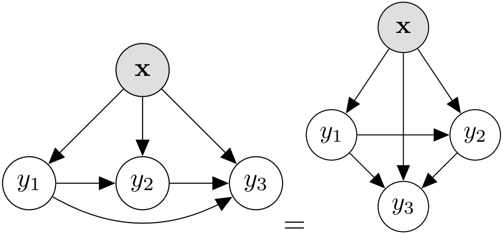
Clearly, $\mathbb{E}(Y_3|Y_1, Y_2, X_1, X_2) = \mathbb{E}(Y_3|X_1, X_2)$, but ...

Table : XOR-problem, 20 examples, base classifier logistic regression.

Measure	BR	CC
Hamming acc.	0.83	1.00
Exact match	0.50	1.00

Example: The XOR Problem

From the point of view of y_3 (the XOR label),

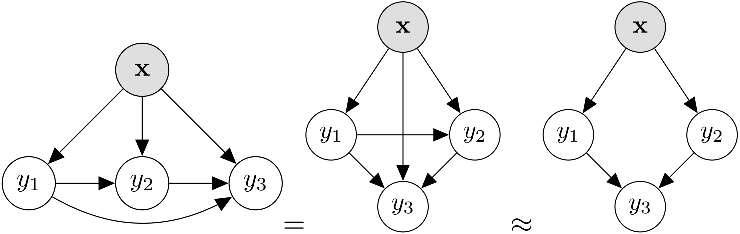


- ≈ A hidden layer![8]

[8] [Read and Hollmén, 2014] to appear in IDA 2014

Example: The XOR Problem

From the point of view of y_3 (the XOR label),



- $\approx$ A hidden layer![8]
- In terms of neural networks, the third graph is enough[9]

8 [Read and Hollmén, 2014] to appear in IDA 2014

9 [Rumelhart et al., 1986]

Classifier Chains: Current Challenges

- Different base classifiers: same chain order?
- Large labelsets
- Deepening connection with related fields, for example
  - neural networks
  - probabilistic graphical models
  - deep learning
  - structured output prediction

End

Thank you!

Questions?

All methods described in this talk implemented in MEKA

http://meka.sourceforge.net

Bibliography

Alessandro, A., Corani, G., Mauá, D., and Gabaglio, S. (2013).

An ensemble of bayesian networks for multilabel classification.

In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI'13, pages 1220–1225. AAAI Press.

Dembczyński, K., Cheng, W., and Hüllermeier, E. (2010).

Bayes optimal multilabel classification via probabilistic classifier chains.

In *ICML '10: 27th International Conference on Machine Learning*, pages 279–286, Haifa, Israel. Omnipress.

Dembczynski, K., Waegeman, W., and Hllermeier, E. (2012).

An analysis of chaining in multi-label classification.

In *ECAI: European Conference of Artificial Intelligence*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 294–299. IOS Press.

Dembszyński, K., Waegeman, W., and Hüllermeier, E. (2011).

Joint mode estimation in multi-label classification by chaining.

In *ECML/PKDD 2011 Workshop on Collective Learning and Inference on Structured Data (CoLISD'11)*.

Guo, Y. and Gu, S. (2011).

Multi-label classification using conditional dependency networks.

In *IJCAI '11: 24th International Conference on Artificial Intelligence*, pages 1300–1305. IJCAI/AAAI.

Kumar, A., Vembu, S., Menon, A., and Elkan, C. (2013).

Beam search algorithms for multilabel learning.

Machine Learning, 92(1):65–89.

Li, N. and Zhou, Z.-H. (2013).