A Revised Understanding of Multi-label Learning and its Implications for Model-Agnostic Transfer Learning and Adaptation to Concept Drift

Jesse Read

December 21, 2023
@ University of Waikato

Outline

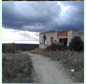
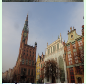
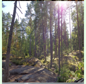
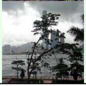
Multi-Label Learning

Multi-label Classification

Multi-label classification: a subset/vector of labels is be assigned to each input instance.

$\boldsymbol{x} =$ 

$\boldsymbol{y} = [1, 0, 1, 0] \Leftrightarrow$ labels $\{\texttt{Beach}, \texttt{Foliage}\}$ are relevant to $\boldsymbol{x}$.

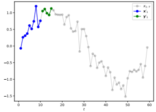
Input	Beach	Sunset	Foliage	Urban
	1	0	1	0
	0	1	0	0
	0	1	0	1
	0	1	1	0
	0	0	1	1
	?	?	?	?

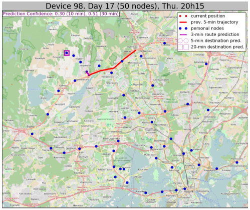
Task:

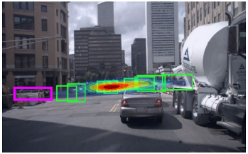
$$\widehat{\boldsymbol{y}} = [?, ?, ?, ?] = h(\boldsymbol{x}) \qquad \widehat{\boldsymbol{y}} \in \{0,1\}^m$$

Also,

- text categorization
- missing-value imputation
- recommender systems
- time-series forecasting
- network inference
- tracking and localization
- image segmentation
- molecule design
- audio labelling
- ...



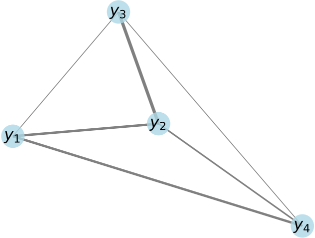




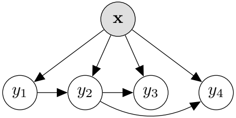
	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
	1,3	0,2	1,4	1,7	3,5	1,3
	2	1,7	1,5	7,5	8,2	7,6
	0,2	0	0,3	0,4	1,2	2,2
	3,1	1,1	1,3	1,1	1,7	5,2
	4,7	2,1	2,5	1,5	2,3	8,5
	?	?	?	?	?	?

Standard ‘Recipe’/Traditional Approach

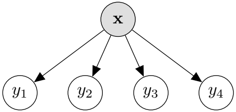
1. ‘We measure label dependence using `<insert method>`’



2. ‘We construct a model called `<insert novel method>`’
3. ‘We show `<insert small number>`%-improvement vs independent models’



vs



vs . . . ?

Implication: Predictive performance $\Leftrightarrow$ **label dependence**.

This talk: A fresh investigation, and an updated view.

A Timeline of Multi-label Learning in Academia

$<$ 2000s Just use independent models.

…2010 Model labels together; label dependence/co-occurrences.

…2015 Using label dependence in a more sophisticated/efficient way.

…2015 Multi-label learning for image, text, forecasting, recommendation, audio, health applications, distilling wine …

2020 […and for covid19].

…2020 Just use independent models

…2020 Must use deep [convolution / recurrent] neural networks.

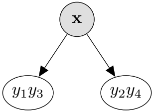
2020… …deep [graph-embedding / residual / generative adversarial / transformer/…] neural networks with [missing / weak / incremental / evolving / imbalanced / millions of/…] labels.

2023 Still persistent in the literature[1]

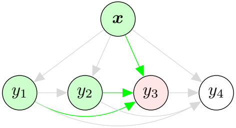
Except: Multi-target regression? $< 1/10$-th volume of literature.

[1] Mylonas et al., "On the Persistence of Multilabel Learning, Its Recent Trends, and Its Open Issues", 2023

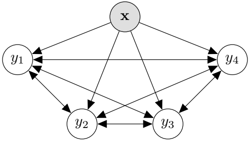
Multi-label Classifiers: Examples



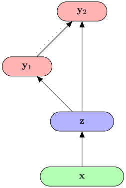
Random *k*-Label Sets and Meta Labels



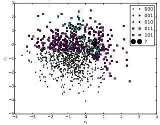
Classifier Chains and Bayesian Networks



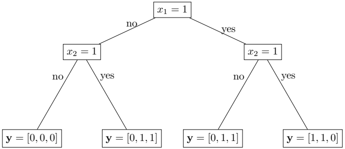
Conditional Dependency Networks



Neural Networks



k-Nearest Neighbours



Decision Trees and Random Forests

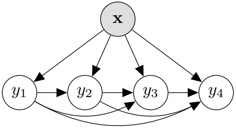
Algorithm Adaptation vs Task Adaptation / Problem Transformation

Refs. in Bogatinovski et al., "Comprehensive comparative study of multi-label classification methods", 2022; plus Cisse, Al-Shedivat, and Bengio, "ADIOS: Architectures Deep In Output Space", 2016

Classifier Chains: An Example of ‘Problem Transformation’

A chain (structure) over the output variables;

- Cascaded prediction across a chain/graph
- Motivation: Model label dependence



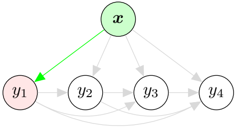
$\boldsymbol{X}$	Y_1	Y_2	Y_3	Y_4
$\boldsymbol{x}^{(1)}$	0	1	1	1
$\boldsymbol{x}^{(2)}$	1	0	0	0
$\boldsymbol{x}^{(3)}$	0	1	0	1
$\boldsymbol{x}^{(4)}$	1	0	0	0
$\boldsymbol{x}^{(5)}$	0	0	0	0
$\tilde{\boldsymbol{x}}$	$\widehat{y_1}$	$\widehat{y_2}$	$\widehat{y_3}$	$\widehat{y_4}$

Read et al., ECML-PKDD 2009 and Read et al., “Classifier Chains: A Review and Perspectives”, 2021

Classifier Chains: An Example of 'Problem Transformation'

A chain (structure) over the output variables;

- Cascaded prediction across a chain/graph
- Motivation: Model label dependence



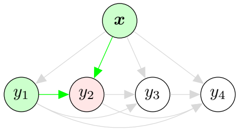
X	Y_1	Y_2	Y_3	Y_4
$x^{(1)}$	0	1	1	1
$x^{(2)}$	1	0	0	0
$x^{(3)}$	0	1	0	1
$x^{(4)}$	1	0	0	0
$x^{(5)}$	0	0	0	0
$\tilde{x}$	$\widehat{y_1}$			

Read et al., ECML-PKDD 2009 and Read et al., "Classifier Chains: A Review and Perspectives", 2021

Classifier Chains: An Example of 'Problem Transformation'

A chain (structure) over the output variables;

- Cascaded prediction across a chain/graph
- Motivation: Model label dependence



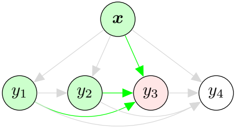
$\boldsymbol{X}$	Y_1	Y_2	Y_3	Y_4
$\boldsymbol{x}^{(1)}$	0	1	1	1
$\boldsymbol{x}^{(2)}$	1	0	0	0
$\boldsymbol{x}^{(3)}$	0	1	0	1
$\boldsymbol{x}^{(4)}$	1	0	0	0
$\boldsymbol{x}^{(5)}$	0	0	0	0
$\tilde{\boldsymbol{x}}$	$\widehat{y_1}$	$\widehat{y_2}$		

Read et al., ECML-PKDD 2009 and Read et al., "Classifier Chains: A Review and Perspectives", 2021

Classifier Chains: An Example of ‘Problem Transformation’

A chain (structure) over the output variables;

- Cascaded prediction across a chain/graph
- Motivation: Model label dependence



$\boldsymbol{X}$	Y_1	Y_2	Y_3	Y_4
$\boldsymbol{x}^{(1)}$	0	1	1	1
$\boldsymbol{x}^{(2)}$	1	0	0	0
$\boldsymbol{x}^{(3)}$	0	1	0	1
$\boldsymbol{x}^{(4)}$	1	0	0	0
$\boldsymbol{x}^{(5)}$	0	0	0	0
$\tilde{\boldsymbol{x}}$	$\widehat{y}_1$	$\widehat{y}_2$	$\widehat{y}_3$	

For example, $\widehat{y}_3 = h_3(\boldsymbol{x}, \widehat{y}_1, \widehat{y}_2)$ with base classifier (or regressor) h_3 (e.g., decision tree, logistic regression, . . .).

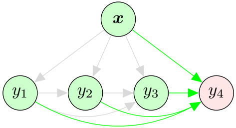
Typical example of a “problem transformation" (or model agnostic) meta method that works well vs independent models

Read et al., ECML-PKDD 2009 and Read et al., “Classifier Chains: A Review and Perspectives”, 2021

Classifier Chains: An Example of 'Problem Transformation'

A chain (structure) over the output variables;

- Cascaded prediction across a chain/graph
- Motivation: Model label dependence



$\boldsymbol{X}$	Y_1	Y_2	Y_3	Y_4
$\boldsymbol{x}^{(1)}$	0	1	1	1
$\boldsymbol{x}^{(2)}$	1	0	0	0
$\boldsymbol{x}^{(3)}$	0	1	0	1
$\boldsymbol{x}^{(4)}$	1	0	0	0
$\boldsymbol{x}^{(5)}$	0	0	0	0
$\tilde{\boldsymbol{x}}$	$\widehat{y}_1$	$\widehat{y}_2$	$\widehat{y}_3$	$\widehat{y}_4$

For example, $\widehat{y}_3 = h_3(\boldsymbol{x}, \widehat{y}_1, \widehat{y}_2)$ with base classifier (or regressor) h_3 (e.g., decision tree, logistic regression, ...).

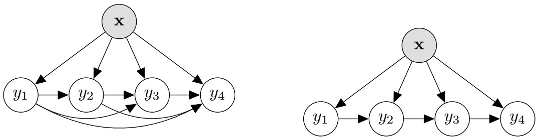
Typical example of a "problem transformation" (or model agnostic) meta method that works well vs independent models – but why?

Read et al., ECML-PKDD 2009 and Read et al., "Classifier Chains: A Review and Perspectives", 2021

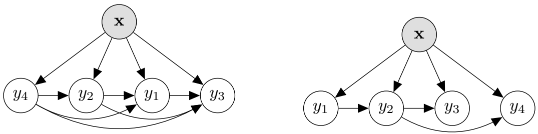
Digging Deeper

War Story 1 (Intuition Fails)

These models perform well:



These ones perform not so well:

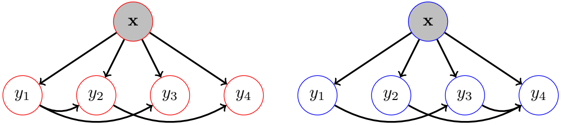


But no obvious pattern/explanation why.

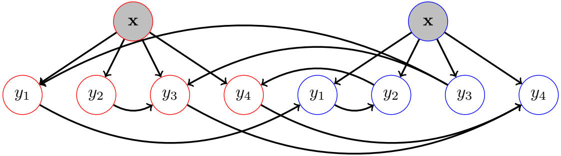
War Story 2 (Sanity Check Fails)

Take two totally unrelated datasets; stick them together; search for inherent structure.

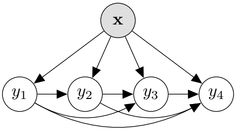
Hypothesis: Find something like this,



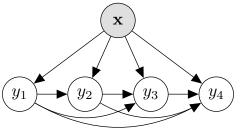
Outcome: Found something like this,



War Story 3 (More Weirdness)



outperforms



Average accuracy over 100 random train/test splits:

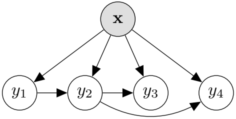
(Left) 0.47 $>$ 0.41 (Right)

and the left wins 100/100 times! Yet, it's the

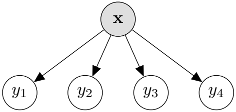
- **same model** (classifier chains)!
- same base classifier (SGD, same initialization)
- same structure
- same data (Scene dataset; same splits)
  except: on the right, we flip the label-indicator bits,
  $\mathbf{Y}_{\text{Right}} = \mathbf{1} - \mathbf{Y}_{\text{Left}}$ (N.B. No information removed/added!)

War Story 4 (Theory $\neq$ Practice?)

Under *Hamming loss* we find that



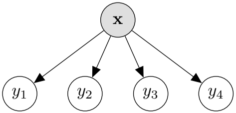
outperforms



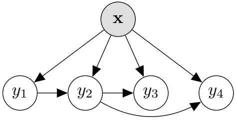
(significantly) even though there is no reason for this to happen (Hamming loss does not require joint modelling to optimize[2]!)

[2]Neither do ranking-based metrics, by the way; Dembczyński et al., "On Label Dependence and Loss Minimization in Multi-label Classification", 2012

War Story 5 (Back to Square One?)



equals performance of



under 0/1-*Loss*/exact-match metric which requires joint modelling to optimize, and even though we know there is label dependence. (Especially common in multi-target regression[3]).

Hence, why the deep learning community usually do not provide structure over outputs (also: the multi-label deep learning papers don't show much interest in exact-match metrics).

[3] Borchani et al., "A Survey on Multi-output Regression", 2015

A Revised Understanding

Suggestion 1: Because Label Dependence

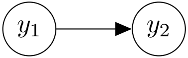
Argument: If label variables are correlated/interdependent, we should model/predict them together; accuracy will better.

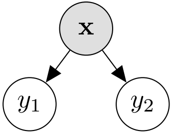
Label dependence:

$$P(Y_1, Y_2) \neq P(Y_1)P(Y_2)$$

Actually, we should be interested in conditional dependence:

$$P(Y_1, Y_2|\boldsymbol{x}) \neq P(Y_1|\boldsymbol{x})P(Y_2|\boldsymbol{x})$$

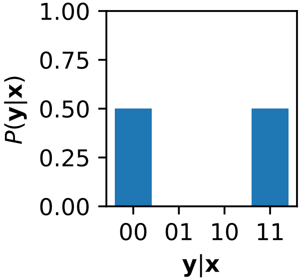
e.g., 

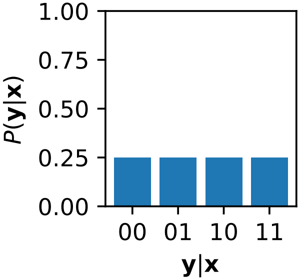
vs 

Dembczyński et al., "On Label Dependence and Loss Minimization in Multi-label Classification", 2012

...Because of Conditional Label Dependence?

Posterior of two multi-label classifiers (2 labels, test instance $\mathbf{x}$):





$\mathbb{E}$[Hamming loss] the same; $\mathbb{E}$[0/1-loss]: twice as large!

Not only a question of dependence, but of loss metrics and uncertainty; modelling together $\neq$ predicting together.

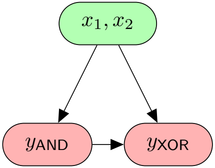
Figures from work with Ekaterina Antonenko and Ander Carreño

The 'Wrong' Dependence

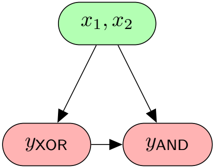
X_1	X_2	Y_{XOR}	Y_{AND}
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	1

vs

X_1	X_2	$\widehat{Y}_{\text{XOR}}$	$\widehat{Y}_{\text{AND}}$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1



outperforms



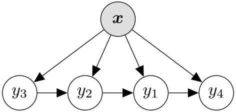
but dependence is symmetrical?

$$Y_2 \sim P_\star(Y_2 \mid \boldsymbol{x}, Y_1) \neq \widehat{Y}_2 \sim \widehat{P}(Y_2 \mid \boldsymbol{x}, \widehat{Y}_1)$$

where $\widehat{P}$ depends on base classifier, inference, etc.
Different distributions! Essentially: distribution shift ('concept drift').

Suggestion 2: Put Easy Labels First

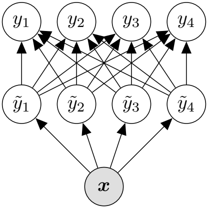
Argument: There may be error propagation across the structure, so we should, e.g., put easy labels first.



But: *Incorrect* label predictions may also *increase* the accuracy of *other* label predictions!

Suggestion 3: Error Correction

Argument: We can ‘correct’ errors (and distributions) at prediction time, e.g., via stacking.



OK[4], but

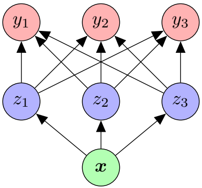
- This is not label dependence modelling, we only correct bias of individual models; and thus
- not much improvement under exact-match metrics
- involves a separate training mechanism for each layer.

[4] e.g., (among many others) Loza Mencía and Janssen, “Learning rules for multi-label classification: a stacking and a separate-and-conquer approach”, 2016

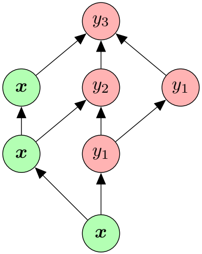
Suggestion 4: Deep Neural Networks

Argument: Just use a deep neural network like everyone else!

It is already! Classifiers as activation functions, labels as non-hidden 'hidden nodes' and delay nodes. A bit like ResNets.



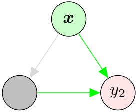
$\approx$



OK, sure – no back propagation (this is deep *prediction*, but not deep learning). So: Yes, deep neural networks can work. In both cases: the structure provides power.

Consider prediction task

$$\tilde{x} \mapsto \widehat{y}_2$$



and the data available at training time (left) vs test time (right):

	X_1		Y_2
Basis expansion	x	ϕ_1	y_2
Stacking	x	$\tilde{y}_2$	y_2
Classifier chain	x	y_1	y_2
Neural network	x		y_2

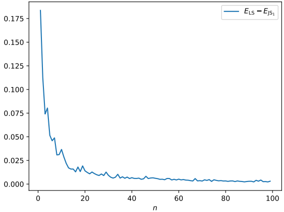
X_1		Y_2
$\tilde{x}$	ϕ_2	$\widehat{y}_2$
$\tilde{x}$	$\tilde{y}_2$	$\widehat{y}_2$
$\tilde{x}$	$\widehat{y}_1$	$\widehat{y}_2$
$\tilde{x}$	z	$\widehat{y}_2$

Suggestion 5: Structure Provides Regularisation

Argument: Modelling together provides regularization.

The James Stein estimator $\widehat{\boldsymbol{y}}_{JS} = \frac{1-(m-2)\hat{\sigma}^2}{\|\widehat{\boldsymbol{y}}\|^2}\widehat{\boldsymbol{y}} = \lambda \cdot \widehat{\boldsymbol{y}}$ where λ *shrinks* (regularises) the max.-likelihood estimate $\widehat{\boldsymbol{y}}$.

Benefit from modelling non-existent label dependence (mainly on the left, low number of examples n):



This helps explain the bit-flip story! Statistical significance, but minimal gains when many examples.

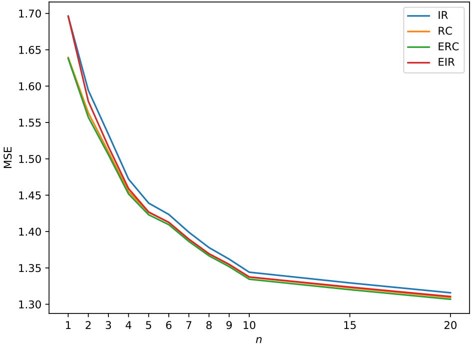
Good discussion by Waegeman, Dembczyński, and Hüllermeier, "Multi-target prediction: a unifying view on problems and methods", 2019

Suggestion 6: The 'Ensemble Effect'

Argument: 'Ensembles of `X`' provides better results but actually the ensemble deserves the credit, not `X`; because ensembles provide

- More predictive power
- More regularisation

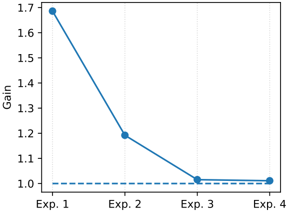
The following methods are all equivalently linear. The ensemble provides a (slight) benefit in terms of regularization only:



E = Ensemble, I = Independent, C = Chain

So Which Is It Then?

Classifier chains vs independent classifiers (Music-Emotions data):



- Exp. 1: ‘Standard’ setup (both with logistic regression as base classifier, under 0/1 loss)
- Exp. 2: Remove benefit from modelling label dependence (use Hamming Loss instead)
- Exp. 3: Remove benefit from predictive power (replace logistic regression with deep NNs)
- Exp. 4: Remove influence of regularisation (heavy regularization)

interesting: 20% higher accuracy by modelling label dependence, even when theoretically pointless!

Read, *From Multi-label Learning to Cross-Domain Transfer*. 2023

Conclusions So Far

We should model and predict labels together mainly because of label dependence (i.e., if our loss metric suggests that we need to learn it), but we can also get benefits from additional capacity and regularisation brought by additional structure inherent to modelling labels together.

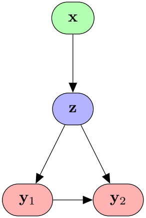
With *enough data/computational power*, regularised deep neural network architectures likely to overpower traditional methods of multi-label learning.

But this is interesting: implies **improvement from modelling totally unrelated tasks together**.

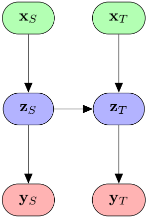
Making use of our Lessons: Model Agnostic Transfer Learning and Adapting to Concept Drift

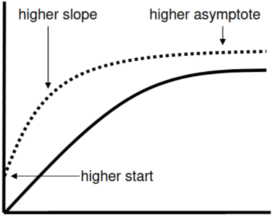
Transfer Learning: A Quick Intro

1. Find related source task (S)
2. Use it to improve the model you deploy on target task (T)



vs



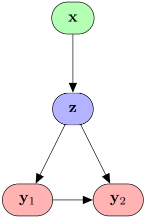


Plot (right) from Torrey and Shavlik, "Transfer learning", 2010.

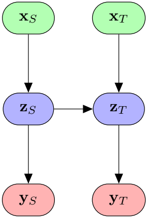
Adapting to concept drift while learning from a data stream = transfer learning.

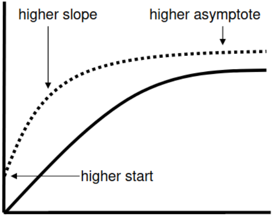
Transfer Learning: A Quick Intro

1. Find related source task (S)
2. Use it to improve the model you deploy on target task (T)



vs





Plot (right) from Torrey and Shavlik, "Transfer learning", 2010.

Adapting to concept drift while learning from a data stream = transfer learning.

A key word was: *related*. But what if relatedness is not a requirement?

Thoughts on That

Transfer learning from unrelated source task; is like connecting the first layer of a neural network randomly (random structure better than no structure)?

> *"Connecting the first layer randomly is just about the stupidest thing you could do" – Yann LeCun*

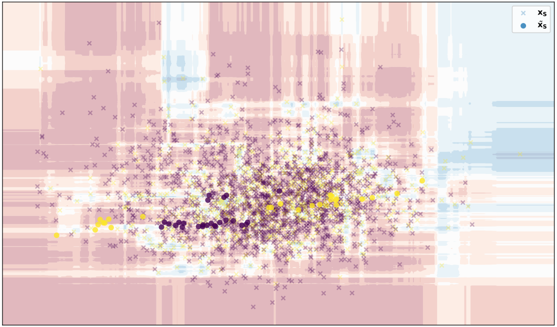
Remarks:

- He said "just about"
- He didn't say it didn't work
- Theres a minor difference: We mean, not randomly drawn from all possible models, rather randomly drawn from all [a collection of] existing *trained* models

So let's try it anyway. . .

Proof of Concept: ‘Insomniac Fungi’

A model (random forest) for classifying patients into insomniac (red) or not (blue), based on clinical sleep data:



We give the same random forest a yeast genome vector, provide an insomnia diagnosis (shown as big dots), use it as new descriptive feature, **boosts** $+2\%$ **accuracy when predicting yeast phenotypes.**

Lessons for dealing with Concept Drift

Some (mostly unsubstantiated) claims, and a few open questions:

- When you react to concept drift, this is why you keep some models! Your models are now somewhat 'irrelevant' but still provide predictive capacity/regularization/...
- Even if drift was complete, you still should keep some models (slow phase-out)!
- What does complete drift mean, anyway? (is it possible for a concept to be 'completely' unrelated to another)?
- If we never deleted any models, would we eventually get a 'universal computation engine' (learn all possible concepts)?
- Limitation of Neural Networks in streams: 'catastrophic forgetting'
- Limitation of Ensembles of Incremental Decision Trees in streams: '**catastrophic remembering**' (relatively poor properties of adaptation and scalability)

A Revised Understanding of Multi-label Learning and its Implications for Model-Agnostic Transfer Learning and Adaptation to Concept Drift

Jesse Read

Thank you!

jesse.read@polytechnique.edu
http://www.lix.polytechnique.fr/~jread/

References I

This talk is based on (many more references within): *From Multi-label Learning to Cross-Domain Transfer: A Model-Agnostic Approach*, J. Read, 2022.
`https://arxiv.org/pdf/2011.11197.pdf`
Accepted/In Press; *Applied Intelligence*.

- Bogatinovski, Jasmin et al. "Comprehensive comparative study of multi-label classification methods". In: *Expert Systems with Applications* 203 (2022), p. 117215.
- Borchani, Hanen et al. "A Survey on Multi-output Regression". In: *Wiley Int. Rev. Data Min. and Knowl. Disc.* 5.5 (Sept. 2015), pp. 216–233. ISSN: 1942-4787. DOI: 10.1002/widm.1157. URL: http://dx.doi.org/10.1002/widm.1157.

References II

- Cisse, Moustapha, Maruan Al-Shedivat, and Samy Bengio. "ADIOS: Architectures Deep In Output Space". In: *Proceedings of The 33rd International Conference on Machine Learning*. Vol. 48. New York, New York, USA: PMLR, 2016, pp. 2770–2779.
- Dembczyński, Krzysztof et al. "On Label Dependence and Loss Minimization in Multi-label Classification". In: *Mach. Learn.* 88.1-2 (July 2012), pp. 5–45. ISSN: 0885-6125. DOI: 10.1007/s10994-012-5285-8.
- Loza Mencía, Eneldo and Frederik Janssen. "Learning rules for multi-label classification: a stacking and a separate-and-conquer approach". In: *Machine Learning* 105.1 (2016), pp. 77–126. ISSN: 1573-0565. DOI: 10.1007/s10994-016-5552-1. URL: https://doi.org/10.1007/s10994-016-5552-1.
- Mylonas, Nikolaos et al. "On the Persistence of Multilabel Learning, Its Recent Trends, and Its Open Issues". In: *IEEE Intelligent Systems* 38.2 (2023), pp. 28–31.

References III

Read, Jesse. "From Multi-label Learning to Cross-Domain Transfer: A Model-Agnostic Approach". In: *Applied Intelligence* 08.2023 (2023), pp. 1537–7497. URL: http://arxiv.org/abs/2207.11742.

Read, Jesse et al. "Classifier Chains: A Review and Perspectives". In: *Journal of Artificial Intelligence Research (JAIR)* 70 (2021). https://jair.org/index.php/jair/article/view/12376/26658, pp. 683–718. URL: https://jair.org/index.php/jair/article/view/12376.

— ."Classifier Chains for Multi-label Classification". In: *ECML-PKDD 2009: 20th European Conference on Machine Learning*. Bled, Slovenia: Springer, 2009, pp. 254–269. URL: http://link.springer.com/chapter/10.1007%2F978-3-642-04174-7_17.

References IV

Torrey, Lisa and Jude Shavlik. "Transfer learning". In: *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

Waegeman, Willem, Krzysztof Dembczyński, and Eyke Hüllermeier. "Multi-target prediction: a unifying view on problems and methods". In: *Data Mining and Knowledge Discovery* 33.2 (2019), pp. 293–324. ISSN: 1573-756X. DOI: 10.1007/s10618-018-0595-5. URL: https://doi.org/10.1007/s10618-018-0595-5.