Multi-Target Prediction: A Unifying View on Problems and Methods

Willem Waegeman

Research Unit Knowledge-based Systems (KERMIT)
Department of Mathematical Modelling, Statistics and Bioinformatics
Ghent University, Belgium

October 13 2017

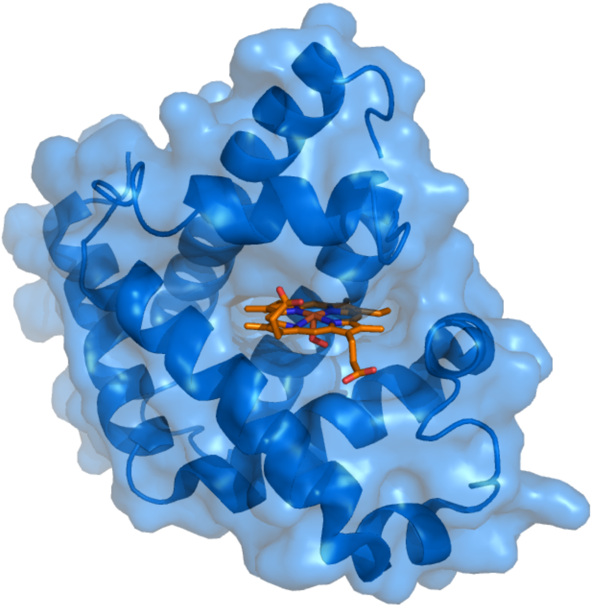
Oranje verslaat Zweden, maar gaat niet naar WK

DINSDAG, 22:38 VOETBAL

Het Nederlands elftal heeft zich zoals verwacht niet geplaatst voor de play-offs om de laatste vier Europese WK-tickets. Oranje won in Amsterdam met 2-0 van Zweden, maar het verschil had zeven of meer doelpunten moeten zijn.

Multi-label classification: the example of document categorization

		Tennis	Football	Biking	Movies	TV	Belgium
01101	**Text1**	0	1	0	0	1	1
00111	**Text2**	1	0	0	0	0	1
01110	**Text3**	0	0	0	1	1	0
10001	**Text4**	0	0	1	0	1	0
01011	**Text5**	1	0	0	1	0	0
11110	**Text6**	?	?	?	?	?	?

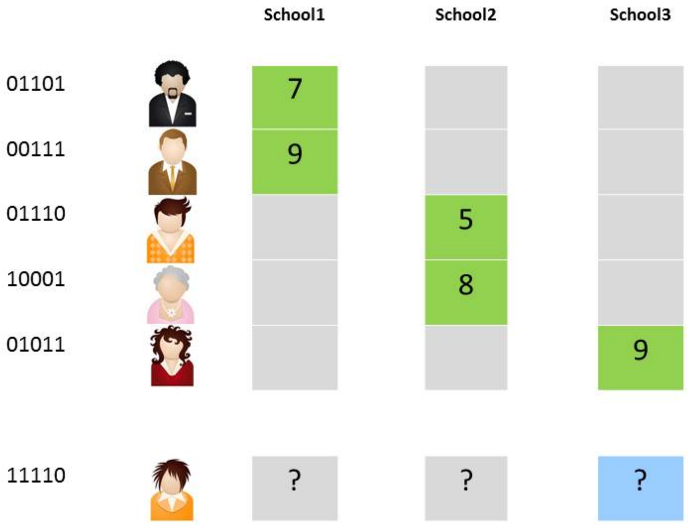


Multivariate regression: the example of protein-ligand interaction prediction

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101	1,3	0,2	1,4	1,7	3,5	1,3
00111	2	1,7	1,5	7,5	8,2	7,6
01110	0,2	0	0,3	0,4	1,2	2,2
10001	3,1	1,1	1,3	1,1	1,7	5,2
01011	4,7	2,1	2,5	1,5	2,3	8,5
11110	?	?	?	?	?	?


SMITHFIELD S.H.S.
STATE HIGH SCHOOL
SUNNYBANK
MANY KINDS OF EXCELLENCE
FERNY GROVE STATE HIGH SCHOOL
ALWAYS AIM HIGH
BRISBANE STATE HIGH SCHOOL
SCIENTIA est POTESTAS
ELANORA STATE HIGH SCHOOL
URANGAN S.H.S.
WORKING TOGETHER
CENTENARY HEIGHTS
CLEVELAND
HIGH
SEMPER DIGNE
ALWAYS AIMING HIGHER
Centena
State High Sch
MIAMI
SEMPER FLOREAT COSTA
KEDRON
HIGH SCHOOL
NITI EST NITERE
TRINITY BAY
ENDEAVOUR
YERONGA
SCIENTIA AC HUMANITATE
SUNSHINE BEACH HIGH
QUALITY IN ALL WE DO
MERRIMAC STATE HIGH SCHOOL
PRIDE IN EXCELLENCE
ENDEAVOUR • DISCOVER • SUCCEED
The Cathedral School
of St Anne & St James
TOWNSVILLE
ROCHEDALE
SHS
INTEGRITY AND ACHIEVEMENT
RECTE CRESCAM
S.H.S
CORINDA
TRUTH
FAITH
COMPASSION
MAROOCHYDORE
HIGH SCHOOL
NITI ET SERVIRE
STATE HIGH SCHOOL
CAIRNS

Multi-task learning: the example of predicting student marks

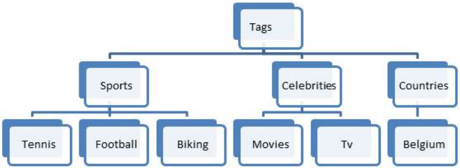


There are a lot of multi-target prediction problems around...

Overview of this talk

1. A unifying view on MTP problems
2. A unifying view on MTP methods
3. Loss functions in multi-target prediction
4. Conclusions

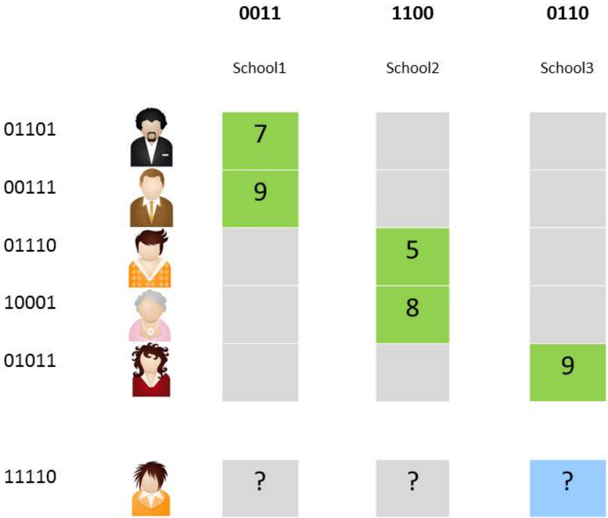
Let's assume a document hierarchy:
How would you call this machine learning problem?



		Tennis	Football	Biking	Movies	Tv	Belgium
01101	**Text1**	0	0	0	0	0	1
00111	**Text2**	0	0	1	0	1	1
01110	**Text3**	0	0	0	1	1	0
10001	**Text4**	0	0	1	0	1	0
01011	**Text5**	1	0	0	1	0	0

11110	**Text6**	?	?	?	?	?	?

Let's assume a target representation:
How would you call this machine learning problem?



Let's assume a target representation:
How would you call this machine learning problem?

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101	1,3	0,2	1,4	1,7	3,5	1,3
00111	2	1,7	1,5	7,5	8,2	7,6
01110	0,2	0	0,3	0,4	1,2	2,2
10001	3,1	1,1	1,3	1,1	1,7	5,2
01011	4,7	2,1	2,5	1,5	2,3	8,5

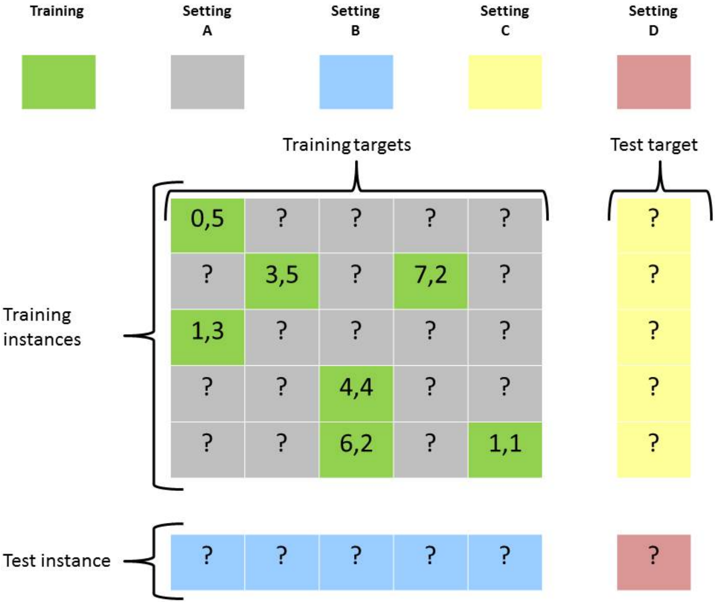
11110	?	?	?	?	?	?

Generalizing to new targets

$g(.,.)$: target similarity

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6	Mol7
01101	1,3	0,2	1,4	1,7	3,5	1,3	?
00111	2	1,7	1,5	7,5	8,2	7,6	?
01110	0,2	0	0,3	0,4	1,2	2,2	?
10001	3,1	1,1	1,3	1,1	1,7	5,2	?
01011	4,7	2,1	2,5	1,5	2,3	8,5	?
11110	?	?	?	?	?	?	?

Important subdivision of different learning settings



General framework

Definition

A multi-target prediction setting is characterized by instances $\boldsymbol{x} \in \mathcal{X}$ and targets $\boldsymbol{t} \in \mathcal{T}$ with the following properties:

1. A training dataset consists of triplets $(\boldsymbol{x}_i, \boldsymbol{t}_j, y_{ij})$, where $y_{ij} \in \mathcal{Y}$.
2. In total n instances and m targets are observed during training, with n and m finite numbers.
3. As such, the scores y_{ij} of the training data can be arranged in an $n \times m$ matrix Y.
4. The score set $\mathcal{Y}$ is one-dimensional. It consists of nominal, ordinal or real values.
5. The goal consists of making predictions for any instance-target couple $(\boldsymbol{x}, \boldsymbol{t}) \in \mathcal{X} \times \mathcal{T}$.

Overview of this talk

1. A unifying view on MTP problems
2. A unifying view on MTP methods
3. Loss functions in multi-target prediction
4. Conclusions

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

A baseline method: learning a model for each target independently

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101	1,3	0,2	1,4	1,7	3,5	1,3
00111	2	1,7	1,5	7,5	8,2	7,6
01110	0,2	0	0,3	0,4	1,2	2,2
10001	3,1	1,1	1,3	1,1	1,7	5,2
01011	4,7	2,1	2,5	1,5	2,3	8,5
11110	?	?	?	?	?	?

A baseline method:
learning a model for each target independently

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101	1,3	0,2	1,4	1,7	3,5	1,3
00111	2	1,7	1,5	7,5	8,2	7,6
01110	0,2	0	0,3	0,4	1,2	2,2
10001	3,1	1,1	1,3	1,1	1,7	5,2
01011	4,7	2,1	2,5	1,5	2,3	8,5

11110	?	?	?	?	?	?

A baseline: Independent Models

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101	1,3	0,2	1,4	1,7	3,5	1,3
00111	2	1,7	1,5	7,5	8,2	7,6
01110	0,2	0	0,3	0,4	1,2	2,2
10001	3,1	1,1	1,3	1,1	1,7	5,2
01011	4,7	2,1	2,5	1,5	2,3	8,5
11110	?	?	?	?	?	?

A baseline: Independent Models

Linear basis function model for i-th target:

$$f_i(\boldsymbol{x}) = \boldsymbol{a}_i^\intercal \phi(\boldsymbol{x}) \,,$$

Solving as a joint optimization problem:

$$\min_A ||Y - XA||_F^2 + \sum_{i=1}^{m} \lambda_i \, ||\boldsymbol{a}_i||^2 \,,$$

With the following notations:

$$X = \begin{bmatrix} \phi(\boldsymbol{x}_1)^T \\ \vdots \\ \phi(\boldsymbol{x}_n)^T \end{bmatrix} \qquad A = [\boldsymbol{a}_1 \quad \cdots \quad \boldsymbol{a}_m] \,.$$

Hamming loss as alternative for binary labels:

$$L_{\mathit{Ham}}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{j=1}^{m} I(y_j = \hat{y}_j)$$

Learning a model for each target independently is still state-of-the-art in extreme multi-label classification[1]:

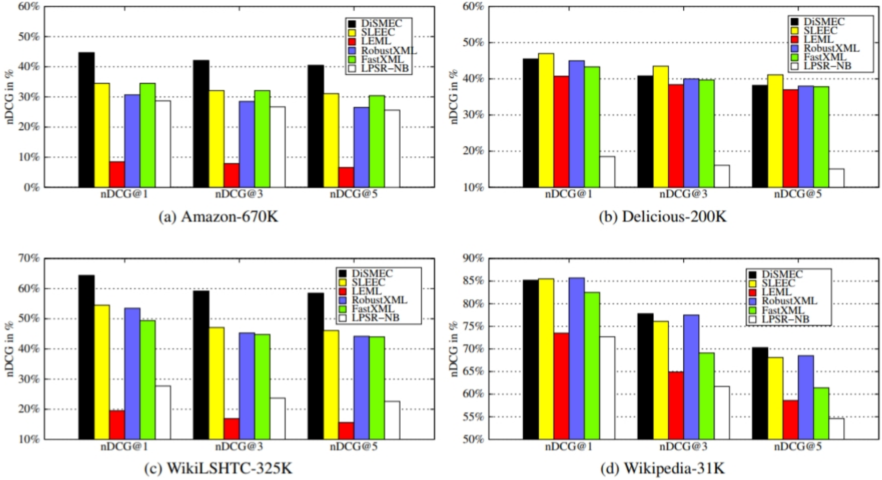


Figure 3: nDCG@k for k=1, 3 and 5

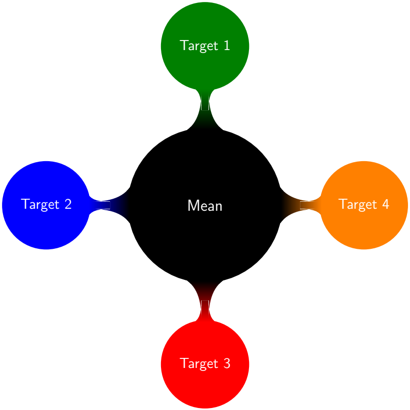
[1] Babbar and Schölkopf, DISMEC: Distributed Sparse Machines for Extreme Multi-label classification, WSDM 2017

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

Mean-regularized multi-task learning[2]

- **Simple assumption**: models for different targets are related to each other.
- **Simple solution**: the parameters of these models should have similar values.
- **Approach**: bias the parameter vectors towards their mean vector.



$$\min_{A} ||Y - XA||_F^2 + \lambda \sum_{i=1}^{m} ||\boldsymbol{a}_i - \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{a}_j||^2 \,,$$

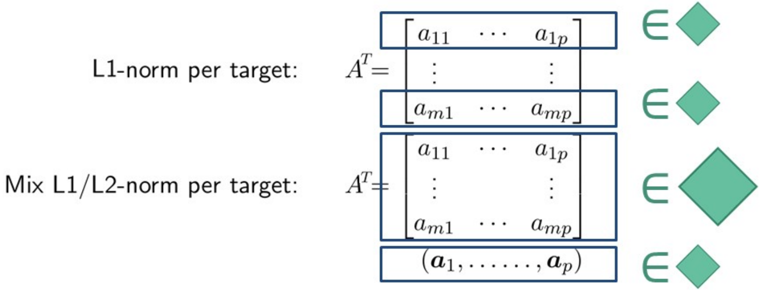
[2] Evgeniou and Pontil, Regularized multi–task learning, KDD 2004.

Joint feature selection

- Enforce that the same features are selected for different targets[3]:

$$\min_A ||Y - XA||_F^2 + \lambda \sum_{j=1}^{p} ||\boldsymbol{a}_j||^2$$

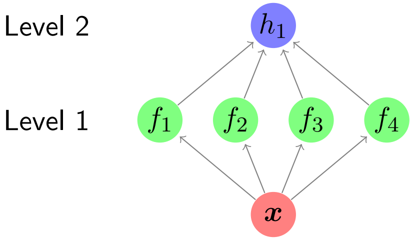
- The vectors $\boldsymbol{a}_j$ now represent the rows of matrix A^T:



[3] Obozinski et al. Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing 2010

Stacking (Stacked generalization)

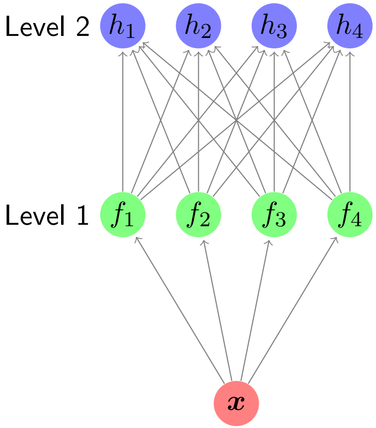
- Originally introduced as a general ensemble learning or blending technique.[4]
- Level 1 classifiers: apply a series of ML methods on the same dataset (or, one ML method on bootstrap samples of the dataset)
- Level 2 classifier: apply an ML method to a new dataset consisting of the predictions obtaining at Level 1



[4] Wolpert, Stacked generalization. Neural Networks 1992

Stacking applied to multi-target prediction[5]

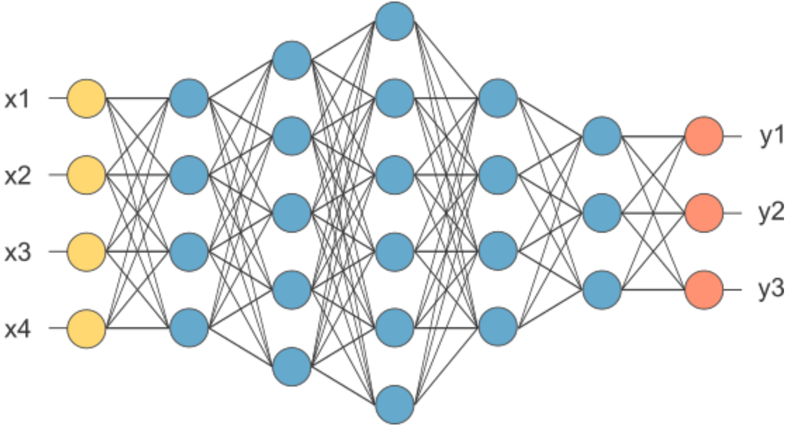
- Level 1 classifiers: learn a model for every target independently
- Level 2 classifier: learn again a model for every target independently, using the predictions of the first step as features



[5] Cheng and Hüllermeier, Combining Instance-based learning and Logistic Regreession for Multi-Label classification, Machine Learning, 2009

MTP in (Deep) Neural Networks

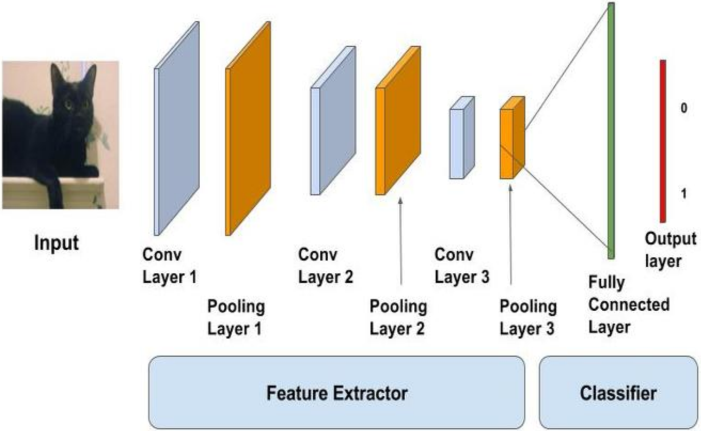
Commonly-used architecture: weight sharing among targets[6]



[6] Caruana, Multitask learning: A knowledge-based source of inductive bias. Machine Learning 1997

Re-using Pretrained Models in (Deep) Neural Networks

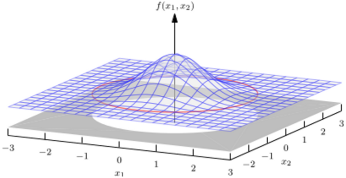
Commonly-used training method: first train on targets that have a lot of observations, only train some parameters for targets that have few observations [7]



[7] Keras Tutorial: Transfer Learning using pre-trained models

An intuitive explanation: James-Stein estimation

- Consider a multivariate normal distribution $\boldsymbol{y} \sim N(\boldsymbol{\theta}, \sigma^2\mathbf{I})$.

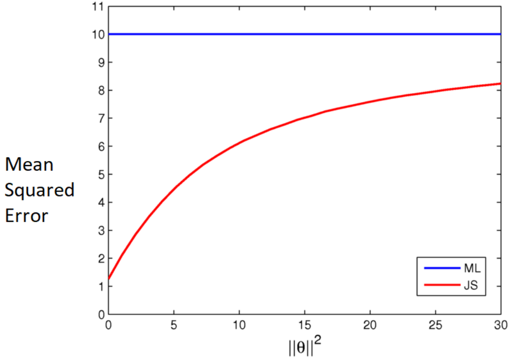


- What is the best estimator of the mean vector $\boldsymbol{\theta}$?
- Evaluation w.r.t. MSE: $\mathbb{E}[(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^2]$
- Single-observation maximum likelihood estimator: $\hat{\boldsymbol{\theta}}^{\mathrm{ML}} = \boldsymbol{y}$
- James-Stein estimator[8]:

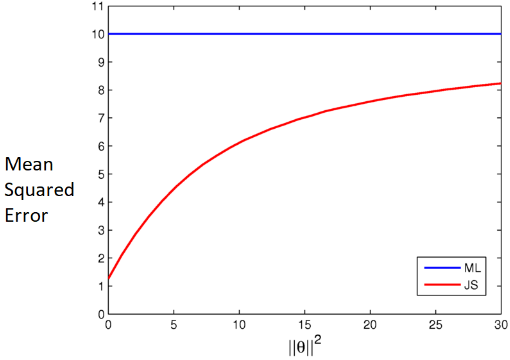
$$\hat{\theta}^{\mathrm{JS}} = \left(1 - \frac{(m-2)\sigma^2}{\|\boldsymbol{y}\|^2}\right)\boldsymbol{y}$$

[8] W. James and C. Stein. Estimation with quadratic loss. In Proc. Fourth Berkeley Symp. Math. Statist. Prob. 1, pages 361-379, 1961

- Works best when the norm of the mean vector is close to zero:



- Works best when the norm of the mean vector is close to zero:



- Regularization towards other directions is also possible:

$$\hat{\theta}^{\text{JS+}} = \left(1 - \frac{(m-2)\sigma^2}{\|\boldsymbol{y} - \boldsymbol{v}\|^2}\right)(\boldsymbol{y} - \boldsymbol{v}) + \boldsymbol{v}$$

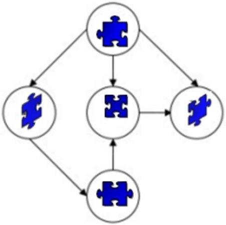
- Only outperforms the maximum likelihood estimator w.r.t. the sum of squared errors over all components.

A unifying view on MTP methods

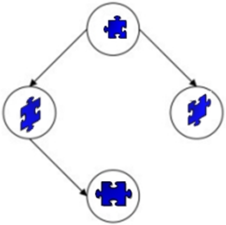
Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

Exploiting relations in regularization terms

Graph



Tree

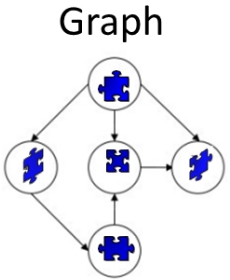


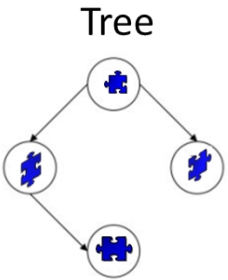
Similarity

	1	0.26	0.26	0.04
	0.26	1	0.7	0.57
	0.26	0.7	1	0.44
	0.04	0.57	0.44	1

[9] Gopal and Yang, Recursive regularization for large-scale classification with hierarchical and graphical dependencies, KDD 2013

Exploiting relations in regularization terms





Similarity

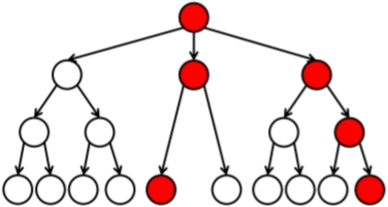
	1	0.26	0.26	0.04
	0.26	1	0.7	0.57
	0.26	0.7	1	0.44
	0.04	0.57	0.44	1

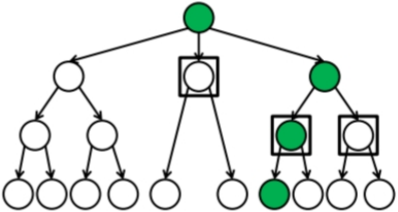
Graph-based regularization is an approach that can be applied to the three types of relations[9]:

$$\min_{A} ||Y - XA||_F^2 + \lambda \sum_{i=1}^{m} \sum_{j \in \mathcal{N}(i)} ||\boldsymbol{a}_i - \boldsymbol{a}_j||^2$$

[9] Gopal and Yang, Recursive regularization for large-scale classification with hierarchical and graphical dependencies, KDD 2013

Hierarchical multi-label classfication


(a) Ground truth.


(b) Prediction A.

In addition to performance gains in general, hierarchies can also be used to define specific loss functions, such as the H-loss[10]:

$$L_H(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{i: y_i \neq \hat{y}_i} c_i \, I(\mathit{anc}(y_i) = \mathit{anc}(\hat{y}_i))$$

c_i depends on the depth of node i

[10] Bi and Kwok, Bayes-optimal hierarchical multi-label classification, IEEE Transactions on Knowledge and Data Engineering, 2014

Exploiting similarity measures among targets

	1	0.26	0.26	0.04
	0.26	1	0.7	0.57
	0.26	0.7	1	0.44
	0.04	0.57	0.44	1

Can be done within the framework of vector-valued kernel functions[11]:

$$f(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{w}^T \Psi(\boldsymbol{x}, \boldsymbol{t}) = \sum_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}) \in \mathcal{D}} \alpha_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})} \Gamma((\boldsymbol{x}, \boldsymbol{t}), (\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}))$$

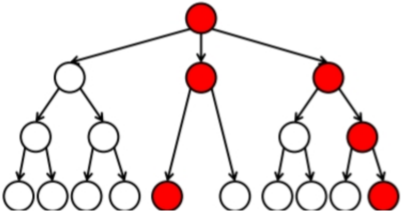
Model the joint kernel as a product of an instance kernel $k(\cdot, \cdot)$ and a target kernel $g(\cdot, \cdot)$:

$$\Gamma((\boldsymbol{x}, \boldsymbol{t}), (\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})) = k(\boldsymbol{x}, \bar{\boldsymbol{x}}) \cdot g(\boldsymbol{t}, \bar{\boldsymbol{t}})$$

[11] Alvarez et al., Kernels for vector-valued functions: a review, Foundation and Trends in Machine Learning

Converting graphs to similarities or target representations

- **Similarities:** use graph structure to express target similarities
  e.g. the shortest-path kernel between two nodes
- **Representations:** often characteristics of a specific vertex or edge
  e.g. the number of positive labels that are siblings of a vertex[12]

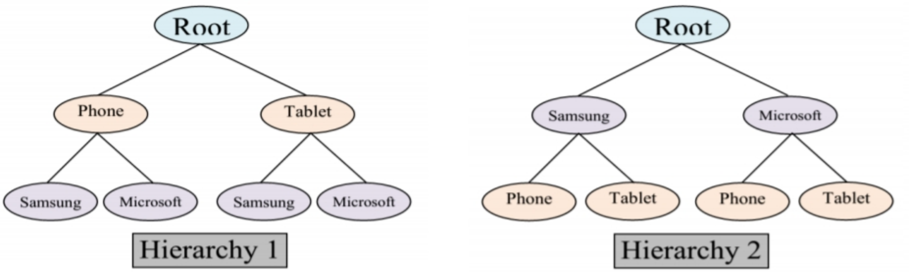


[12] Rousu et al., Kernel-based learning of hierarchical multilabel classification models, JMLR 2006

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

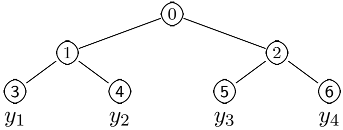
Constructing target hierarchies



- It might be difficult for a human expert to define a hierarchy[13]
- Perhaps one can try to learn the hierarchy from data?
- Algorithms: level flattening, node removal, hierarchy modification, hierarchy generation, etc.

[13] Rangwala and Naik, Tutorial on Large-Scale Hierarchical Classification, KDD 2017.

Label trees ($\neq$ decision trees)



- Organize classifiers in a tree structure (one leaf $\Leftrightarrow$ one label)
- Mainly used in multi-class and multi-label classification
- Goal is fast prediction: almost logarithmic in the number of labels
- Algorithms: Label embedding trees[14], Nested dichotomies[15], Conditional probability trees[16], Hierarchical softmax[17], FastText[18], Probabilistic classifier chains[19]

---

[14] Bengio et al., Label embedding trees for large multi-class tasks, NIPS 2010

[15] Frank and Kramer, Ensembles of nested dichotomies for multi-class problems, ICML 2004

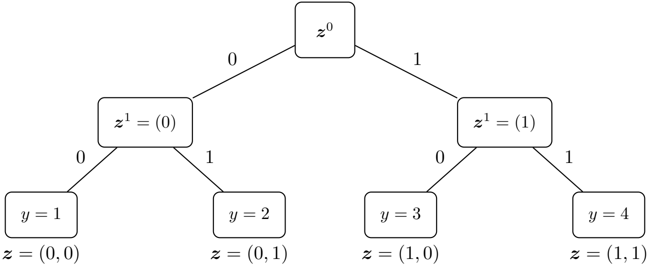
[16] Beygelzimer et al., Conditional probability tree estimation analysis and algorithms. UAI 2009

[17] Morin and Bengio, Hierarchical probabilistic neural network language model, AISTATS 2005

[18] Joulin et al., Bag of tricks for efficient text classifcation. CoRR, abs/1607.01759, 2016

[19] Dembczynski et al., Bayes optimal multilabel classification via probabilistic classifier chains, ICML 2010

Hierarchical softmax / Probabilistic classifier trees



- Encode the targets by a **prefix code** (⇒ tree structure)[20]
- Multi-class classification: each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$
- Multi-label classification: a label vector $\boldsymbol{y} = (y_1, \ldots, y_m)$ is a prefix code.

[20] Dembczynski et al., Consistency of probabilistic classifier trees. ECMLPKDD 2016

Probabilistic classifier chains

- Estimate the joint conditional distribution $P(\boldsymbol{Y} \,|\, \boldsymbol{x})$.
- For optimizing the subset 0/1 loss:

$$\ell_{0/1}(\boldsymbol{y}, \hat{y}) = [\![\boldsymbol{y} \neq \hat{y}]\!]$$

- Repeatedly apply the **product rule of probability**:

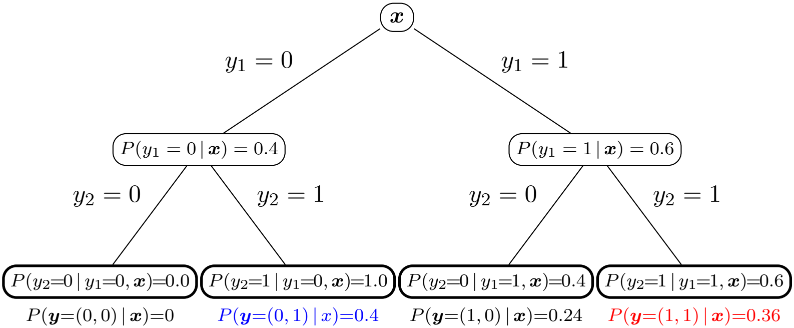
$$P(\boldsymbol{Y} = \boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{i=1}^{m} P(Y_i = y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1})\,.$$

- Learning relies on constructing probabilistic classifiers for estimating

$$P(Y_i = y_i | \boldsymbol{x}, y_1, \ldots, y_{i-1})\,,$$

independently for each $i = 1, \ldots, m$.

- Inference relies on exploiting a probability tree:



- For subset 0/1 loss one needs to find $\boldsymbol{h}(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} P(\boldsymbol{y} \mid \boldsymbol{x})$.
- Greedy and approximate search techniques with guarantees exist.[21]
- Other losses: compute the prediction on a sample from $P(\boldsymbol{Y} \mid \boldsymbol{x})$.[22]

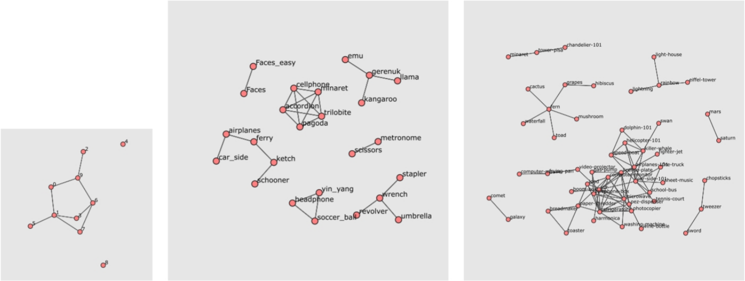
[21] Kumar et al., Beam search algorithms for multilabel learning, Machine Learning 2013

[22] 23 Dembczynski et al., An analysis of chaining in multi-label classification, ECAI 2012

Constructing target similarities by output kernel learning

- Consider models $\mathbf{f} : \mathcal{X} \to \mathbb{R}^{\mathbf{m}}$
- Training dataset $\{\mathbf{x_i}, \mathbf{y_i}\}_{\mathbf{i=1}}^{\mathbf{n}}$
- Learnable correlation matrix $\mathbf{\Gamma}$ between targets
- Learn output kernel and model parameters jointly[23]:

$$\min_{\mathbf{\Gamma} \in \mathbb{R}^{\mathbf{m}\times\mathbf{m}}} \left[\min_{\mathbf{f}\in\mathcal{F}} \sum_{i=1}^{n} \frac{||\mathbf{f}(\mathbf{x_i}) - \mathbf{y_i}||_{\mathbf{2}}^{\mathbf{2}}}{2\lambda} + \frac{||\mathbf{f}||_{\mathcal{F}}^{\mathbf{2}}}{2} + \frac{||\mathbf{\Gamma}||_{\mathbf{F}}^{\mathbf{2}}}{2} \right]$$

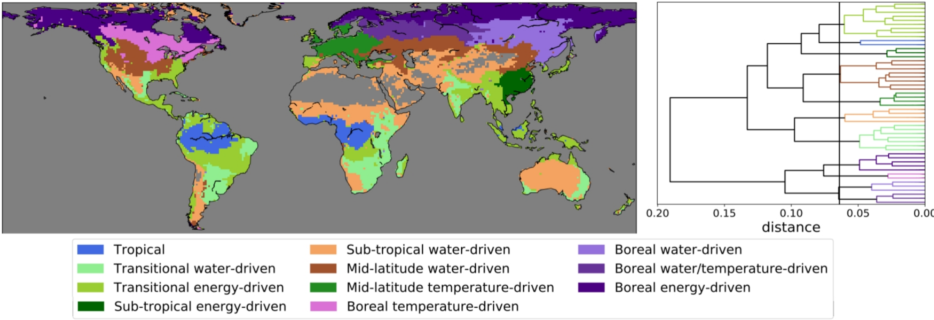


(a) USPS digits (b) Caltech 101 (c) Caltech 256

[23] Dinuzzo et al., Learning Output Kernels with Block Coordinate Descent, ICML 2011

Construcing hierarchies to obtain additional insight

- Application in climate science
- Result of learning 20000 tasks simultaneously with a multi-task learning method
- Followed by hierarchical clustering of the learned weight vectors[24]:

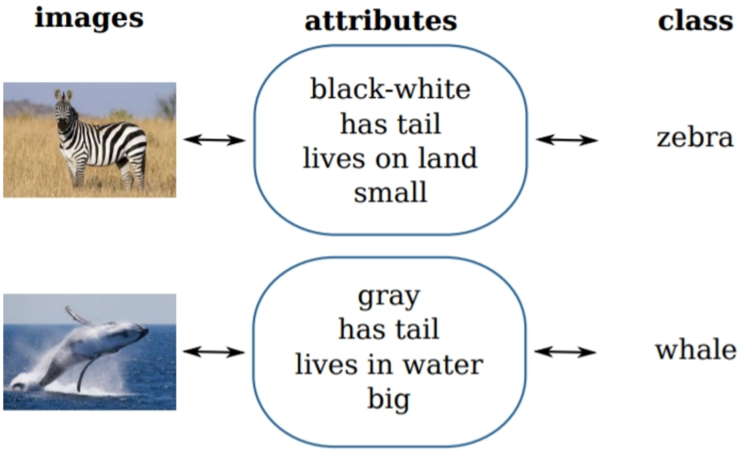


[24] Papagiannopoulou et al. Globral hydro-climatic biomes identified with multi-task learning, Geoscientific Model Development Discussions 2018

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

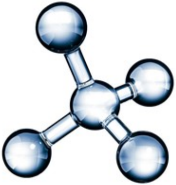
A target representation in computer vision

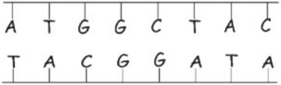


Target representations are the key element of zero-shot learning methods[25]

[25] Examples taken from the CVPR 2016 Tutorial on Zero-shot learning for Computer Vision

Target representations can take many forms





Kronecker kernel ridge regression

Pairwise model representation in the primal:

$$f(\boldsymbol{x}, \boldsymbol{t}) = \boldsymbol{w}^T \left(\phi(\boldsymbol{x}) \otimes \psi(\boldsymbol{t}) \right)$$

Kronecker product pairwise kernel in the dual[26]:

$$f(\boldsymbol{x}, \boldsymbol{t}) = \sum_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}) \in \mathcal{D}} \alpha_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})} k(\boldsymbol{x}, \bar{\boldsymbol{x}}) \cdot g(\boldsymbol{t}, \bar{\boldsymbol{t}}) = \sum_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}) \in \mathcal{D}} \alpha_{(\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}})} \Gamma((\boldsymbol{x}, \boldsymbol{t}), (\bar{\boldsymbol{x}}, \bar{\boldsymbol{t}}))$$

Least-squares minimization with $\mathbf{z} = \mathrm{vec}(Y)$:

$$\min_{\boldsymbol{\alpha}} ||\boldsymbol{\Gamma}\boldsymbol{\alpha} - \mathbf{z}||_2^2 + \lambda \boldsymbol{\alpha}^\intercal \boldsymbol{\Gamma} \boldsymbol{\alpha}$$

[26] Waegeman et al., A kernel framework for learning graded relations from data, IEEE Transaction on Fuzzy Systems, 2012

Two-step zero-shot learning[27] [28]

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6	Mol7
01101	1,3	0,2	1,4	1,7	3,5	1,3	?
00111	2	1,7	1,5	7,5	8,2	7,6	?
01110	0,2	0	0,3	0,4	1,2	2,2	?
10001	3,1	1,1	1,3	1,1	1,7	5,2	?
01011	4,7	2,1	2,5	1,5	2,3	8,5	?
11110	?	?	?	?	?	?	?

[27] Pahikkala et al. A two-step approach for solving full and almost full cold-start problems in dyadic prediction, ECML/PKDD 2014.

[28] Romero-Paredes and Torr, An embarrassingly simple approach to zero-shot learning, ICML 2015.

Two-step zero-shot learning[29] [30]

	Mol1	Mol2	Mol3	Mol4	Mol5	Mol6
01101	1,3	0,2	1,4	1,7	3,5	1,3
00111	2	1,7	1,5	7,5	8,2	7,6
01110	0,2	0	0,3	0,4	1,2	2,2
10001	3,1	1,1	1,3	1,1	1,7	5,2
01011	4,7	2,1	2,5	1,5	2,3	8,5
11110	1,2	2,1	1,7	4,3	2,4	2,5

[29] Pahikkala et al. A two-step approach for solving full and almost full cold-start problems in dyadic prediction, ECML/PKDD 2014.

[30] Romero-Paredes and Torr, An embarrassingly simple approach to zero-shot learning, ICML 2015.

Two-step zero-shot learning[31] [32]

Mol1	Mol2	Mol3	Mol4	Mol5	Mol6	Mol7
1,3	0,2	1,4	1,7	3,5	1,3	1,2
2	1,7	1,5	7,5	8,2	7,6	1,4
0,2	0	0,3	0,4	1,2	2,2	3,8
3,1	1,1	1,3	1,1	1,7	5,2	1,1
4,7	2,1	2,5	1,5	2,3	8,5	1,5
1,2	2,1	1,7	4,3	2,4	2,5	4,3

[31] Pahikkala et al. A two-step approach for solving full and almost full cold-start problems in dyadic prediction, ECML/PKDD 2014.

[32] Romero-Paredes and Torr, An embarrassingly simple approach to zero-shot learning, ICML 2015.

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\intercal$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_q))^\intercal$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\intercal$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_q))^\intercal$$

- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\intercal A^{IT} = \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_d \mathbf{I}\right)^{-1} Y$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\intercal$$
$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_q))^\intercal$$

- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\intercal A^{IT} = \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_d \mathbf{I}\right)^{-1} Y$$

- Step 2: generalizing to new targets

$$f^{\mathsf{TS}}(\mathbf{x}, \mathbf{t}) = \mathbf{g}(\mathbf{t})^\intercal \left(\mathbf{G} + \lambda_t \mathbf{I}\right)^{-1} \mathbf{f}_T(\mathbf{x})^\intercal$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n))^\intercal$$

$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t}, \mathbf{t}_1), \ldots, g(\mathbf{t}, \mathbf{t}_q))^\intercal$$

- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\intercal A^{IT} = \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_d \mathbf{I}\right)^{-1} Y$$

- Step 2: generalizing to new targets

$$\begin{aligned} f^{\mathsf{TS}}(\mathbf{x}, \mathbf{t}) &= \mathbf{g}(\mathbf{t})^\intercal \left(\mathbf{G} + \lambda_t \mathbf{I}\right)^{-1} \mathbf{f}_T(\mathbf{x})^\intercal \\ &= \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_d \mathbf{I}\right)^{-1} Y \left(\mathbf{G} + \lambda_t \mathbf{I}\right)^{-1} \mathbf{g}(\mathbf{t}) \end{aligned}$$

Two-step kernel ridge regression

- Kernel evaluations for new test instance:

$$\mathbf{k}(\mathbf{x}) = (k(\mathbf{x},\mathbf{x}_1),\ldots,k(\mathbf{x},\mathbf{x}_n))^\intercal$$
$$\mathbf{g}(\mathbf{t}) = (g(\mathbf{t},\mathbf{t}_1),\ldots,g(\mathbf{t},\mathbf{t}_q))^\intercal$$

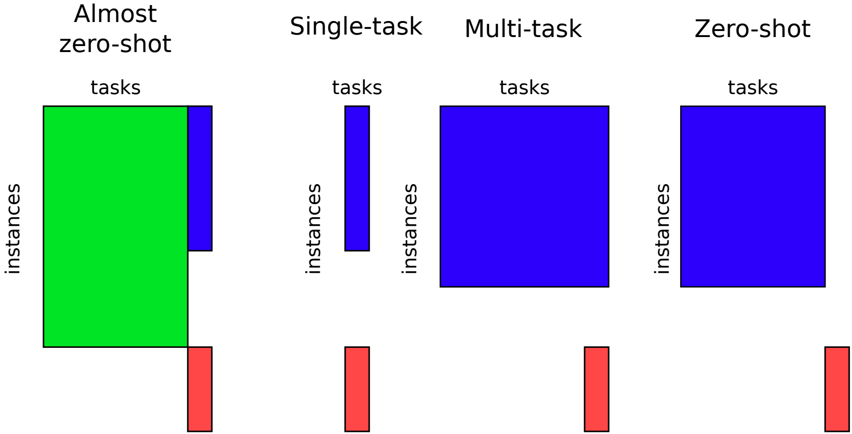
- Step 1: prediction for $\mathbf{x}$ on all the training targets

$$\mathbf{f}_T(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\intercal A^{IT} = \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_d \mathbf{I}\right)^{-1} Y$$

- Step 2: generalizing to new targets

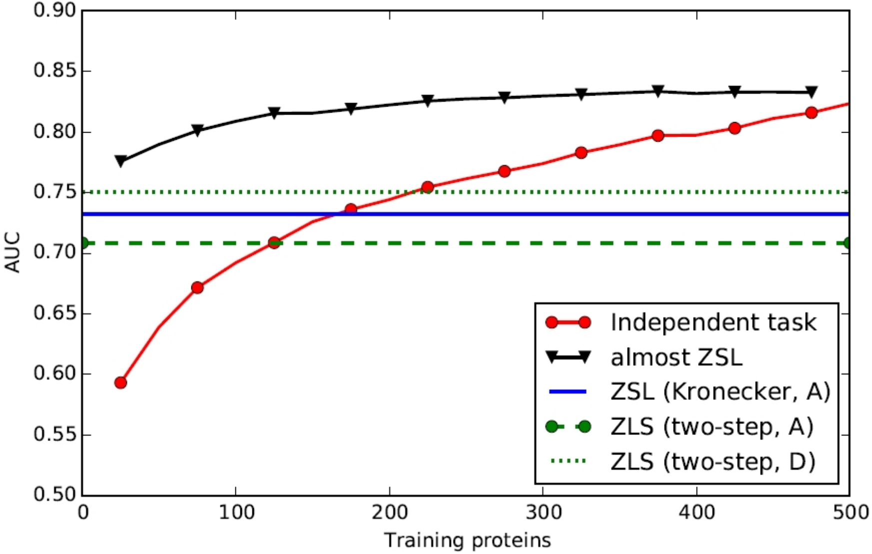
$$\begin{aligned} f^{\mathsf{TS}}(\mathbf{x},\mathbf{t}) &= \mathbf{g}(\mathbf{t})^\intercal \left(\mathbf{G} + \lambda_t \mathbf{I}\right)^{-1} \mathbf{f}_T(\mathbf{x})^\intercal \\ &= \mathbf{k}(\mathbf{x})^\intercal \left(\mathbf{K} + \lambda_d \mathbf{I}\right)^{-1} Y \left(\mathbf{G} + \lambda_t \mathbf{I}\right)^{-1} \mathbf{g}(\mathbf{t}) \\ &= \mathbf{k}(\mathbf{x})^\intercal A^{\mathsf{TS}} \mathbf{g}(\mathbf{t}) \\ &= \boldsymbol{w}^T \left(\phi(\boldsymbol{x}) \otimes \psi(\boldsymbol{t})\right) \end{aligned}$$

Almost zero-shot learning: definition and experimental setup

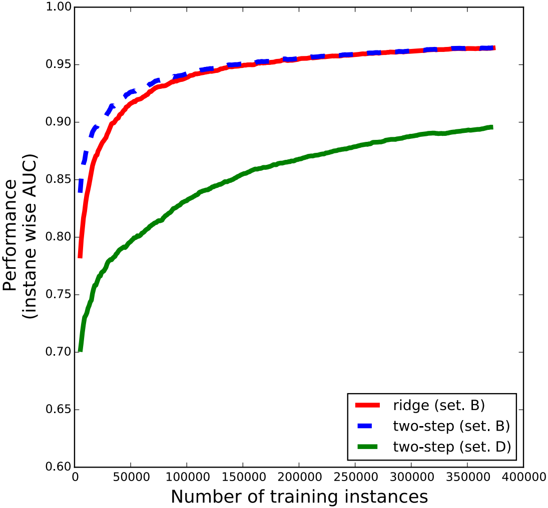


Gradually increase the number of training instances for the "new" task

Almost zero-shot learning: results for protein-ligand interaction prediction



Zero-shot learning of document categorization

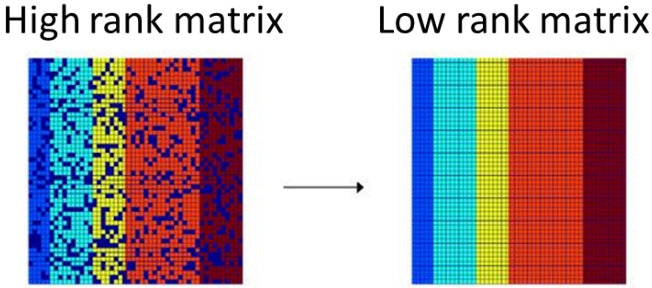


$12,000$ labels: from $5,000$ to $350,000$ instances

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

Methods that learn target representations (B and C)
Example: low-rank parameter matrix approximation[33]



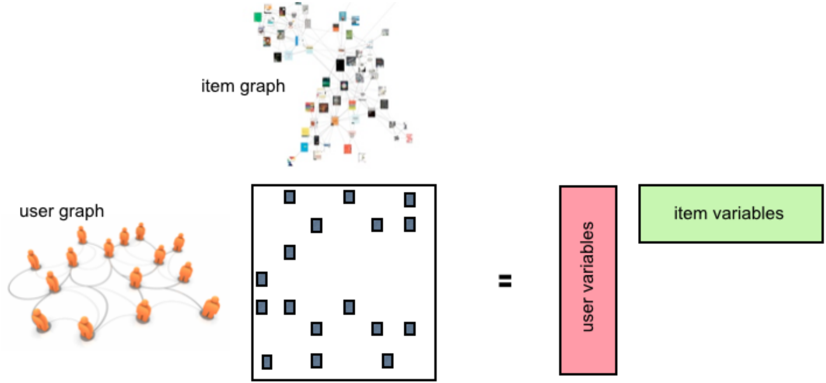
$$\min_{A} ||Y - XA||_F^2 + \lambda \, \mathrm{rank}(A)$$

[33] Chen et al., A convex formulation for learning shared structures from multiple tasks, ICML 2009.

A unifying view on MTP methods

Group of methods	Applicable setting
Independent models	B and C
Similarity-enforcing methods	B and C
Relation-exploiting methods	B, C and D
Relation-constructing methods	B and C
Representation-exploiting methods	B, C and D
Representation-constructing methods	B and C
Matrix completion and hybrid methods	A

Matrix completion and hybrid methods (A)
Example: matrix factorization + bilinear models[34]



$$f(\boldsymbol{x},\boldsymbol{t}) = \boldsymbol{w}^T \big(\phi(\boldsymbol{x}) \otimes \psi(\boldsymbol{t})\big)$$

[34] Menon and Elkan, A log-linear model with latent features for dyadic prediction, ICDM 2010.

Overview of this tutorial

Conclusions

- Multi-target prediction is an active field of research that connects different types of machine learning problems
- In the corresponding subfields of machine learning, problems have typically been solved in isolation, without establishing connections between methods
- Two-step zero-shot learning is a simple MTP method with a lot of interesting properties

Upcoming paper:
Waegeman et al.
Multi-Target Prediction:
A Unifying View on Problems and Methods