Graph-based Regularization of Ranking

(MADSPAM contribution)

Artem Sokolov[1] Tanguy Urvoy[1] Ludovic Denoyer[2] Olivier Ricard[3]

[1]Orange Labs, Lannion (france)

[2]Univ. Pierre et Marie Curie, Paris (france)

[3]BlogSpirit, Paris (france)

ECML/PKDD Discovery challenge, 2010

About MADSPAM

MADSPAM is and industrial research project supported by french research agency

Consortium:

Purpose: automatic SPAM detection in large networks
Duration: 2008-2010
Leader: Orange labs

Outline of this talk

1. Problem statement (1 slide)
2. Approach (8 slides)
3. Experiments and Results (4 slides)

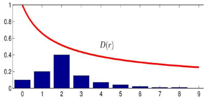
ECML/PKDD 2010 Web Quality Ranking Challenge

Given:

- host instances $\mathcal{I} = \mathcal{I}_{train} \uplus \mathcal{I}_{test}$
- host features x_i $\quad (i \in \mathcal{I})$
- 10 categories $c \in C$
  no independence:
  trust vs. spam or neutral vs. biased
- reference ranking levels y_i^c $\quad (i \in \mathcal{I}_{train}, c \in C)$
- link matrix $n_{i,j} = |\{\text{links between i and j}\}|$

Predict (for each category c):

- a linear ordering of $\mathcal{I}_{test}$ optimizing NDCG



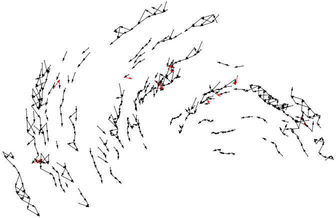
$\sim$ predict ranking scores $\hat{y}_i^c$

LTR $\neq$ Classification

Critical ranking pairs

$$D^c = \{(i,j) \mid y_i^c < y_j^c\}$$

Learning to Rank intuition



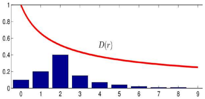
ECML/PKDD 2010 Web Quality Ranking Challenge

Given:

- host instances $\mathcal{I} = \mathcal{I}_{train} \uplus \mathcal{I}_{test}$
- host features x_i $(i \in \mathcal{I})$
- 10 categories $c \in C$
  no independence:
  trust vs. spam or neutral vs. biased
- reference ranking levels y_i^c $(i \in \mathcal{I}_{train}, c \in C)$
- link matrix $n_{i,j} = |\{\text{links between i and j}\}|$

Predict (for each category c):

- a linear ordering of $\mathcal{I}_{test}$ optimizing NDCG



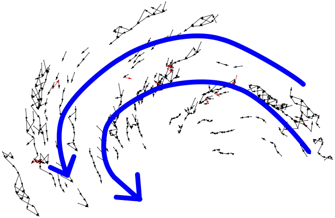
$\sim$ predict ranking scores $\hat{y}_i^c$

LTR $\neq$ Classification

Critical ranking pairs

$$D^c = \{(i,j) \mid y_i^c < y_j^c\}$$

Learning to Rank intuition



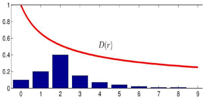
ECML/PKDD 2010 Web Quality Ranking Challenge

Given:

- host instances $\mathcal{I} = \mathcal{I}_{train} \uplus \mathcal{I}_{test}$
- host features x_i $(i \in \mathcal{I})$
- 10 categories $c \in C$
  no independence:
  trust vs. spam or neutral vs. biased
- reference ranking levels y_i^c $(i \in \mathcal{I}_{train}, c \in C)$
- link matrix $n_{i,j} = |\{\text{links between i and j}\}|$

Predict (for each category c):

- a linear ordering of $\mathcal{I}_{test}$ optimizing NDCG



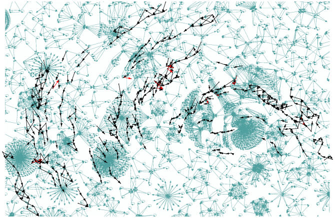
$\sim$ predict ranking scores $\hat{y}_i^c$

LTR $\neq$ Classification

Critical ranking pairs

$$D^c = \{(i,j) \mid y_i^c < y_j^c\}$$

Learning to Rank intuition



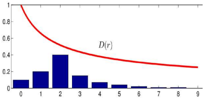
ECML/PKDD 2010 Web Quality Ranking Challenge

Given:

- host instances $\mathcal{I} = \mathcal{I}_{train} \uplus \mathcal{I}_{test}$
- host features x_i $\quad (i \in \mathcal{I})$
- 10 categories $c \in C$
  no independence:
  trust vs. spam or neutral vs. biased
- reference ranking levels y_i^c $\quad (i \in \mathcal{I}_{train}, c \in C)$
- link matrix $n_{i,j} = |\{\text{links between i and j}\}|$

Predict (for each category c):

- a linear ordering of $\mathcal{I}_{test}$ optimizing NDCG



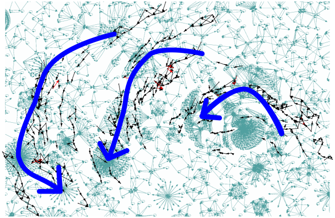
$\sim$ predict ranking scores $\hat{y}_i^c$

LTR $\neq$ Classification

Critical ranking pairs

$$D^c = \{(i,j) \mid y_i^c < y_j^c\}$$

Learning to Rank intuition



Approach

Outline of our approach

Objective:

Combine efficiently instance-level information and relational information

Two-steps semi-transductive approach:

1. **inductive step:** train a ranking model on instance-based features
   - RankBoost model [Freund & Shapire 2002]
2. **transductive step:** use relational data to consolidate rank predictions
   - two methods:
     - smoothing regularization (WITCH [Abernethy et al. 2008] like)
     - multi-category iterative algorithm [Denoyer et al. 2010]

Instance-based Ranking Model: RankBoost

Rankboost loss function [Freund & Shapire 2002]

Target Loss : weighted pairwise disagreement (1 - weighted Kendall τ)

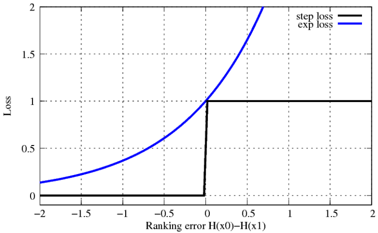
$$\mathcal{W} = \sum_{(x_0,x_1)\in D} D(x_0,x_1)\cdot[H(x_0)\geq H(x_1)]$$

- importance of respecting $x_0 < x_1$ given by matrix $D(x_0,x_1)\in[0,1]$
- if same weight for each pair: same as Kendall τ
- approximation of other IR metrics (NDCG,ERR,MAP…) via D matrix
- hard to optimize

Rankboost convex approximation of $\mathcal{W}$

$$\mathcal{W} \leq \sum_{(x_0,x_1)\in D} D(x_0,x_1)\cdot e^{H(x_0)-H(x_1)}$$

- errors should have small rank difference
- nice properties of the exponential



Rankboost algorithm [Freund & Shapire 2002]

Rankboost set of functions

- We define a family of weak ranking functions :

$$h : \mathcal{X} \rightarrow I\!R$$

- The space to explore is the set of linear combinations of these weak functions:

$$H_T(x) = \sum_{t}^{T} \alpha_t h_t(x)$$

Rankoost principle

- Boosting is iterative, at each step:
  - it keeps trace of wrongly ordered pairs by changing the D matrix
  - it searches the weak model that best reduces the loss for these pairs
- This is a kind of gradient descent
- T set by cross-validation

Remark: no category interdependence

Weak learners

Most simple: feature selection (linear model)

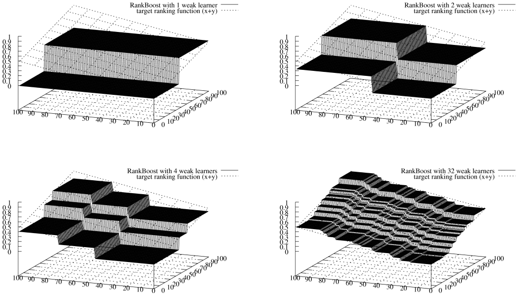
one parameter: i

$$h(x) = f_i(x)$$

Most common : decision stump

two parameters: i and θ

$$h(x) = \begin{cases} 1, & \text{if } x_i > \theta, \\ 0, & \text{if } x_i \leq \theta, \end{cases}$$



Most powerful : grids and trees

unstable: need a regularization

Graph-based Regularization

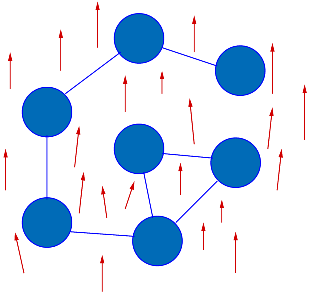
Regularization-based propagation (WITCH [Abernethy et al. 2008] like)

Assumptions

1. **Consistency**: a good ranking score should be close to the instance-based model
2. **Smoothness**: it should associate similar ranks to connected nodes

Hence, for a category c, the loss is defined as:

$$L^c = \underbrace{\sum_{i \in \mathcal{I}} (z_i^c - \hat{y}_i^c)^2}_{\text{consistency}} + \lambda \underbrace{\sum_{i,j \in \mathcal{I}} w_{i,j} (z_i^c - z_j^c)^2}_{\text{smoothness}} .$$



- inference: stochastic gradient descent
- λ set by cross-validation

Remarks

- simple and elegant use of relational structure
- strong assumptions about graph locality
- no category interdependence

Regularization-based propagation (WITCH [Abernethy et al. 2008] like)

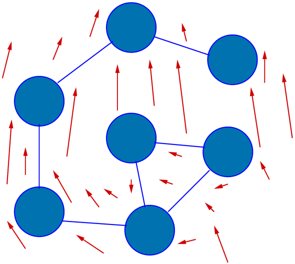
Assumptions

1. **Consistency**: a good ranking score should be close to the instance-based model
2. **Smoothness**: it should associate similar ranks to connected nodes

Hence, for a category c, the loss is defined as:

$$L^c = \underbrace{\sum_{i \in \mathcal{I}} (z_i^c - \hat{y}_i^c)^2}_{\text{consistency}} + \lambda \underbrace{\sum_{i,j \in \mathcal{I}} w_{i,j} (z_i^c - z_j^c)^2}_{\text{smoothness}} .$$

- inference: stochastic gradient descent
- λ set by cross-validation



Remarks

- simple and elegant use of relational structure
- strong assumptions about graph locality
- no category interdependence

Iterative propagation

Objective

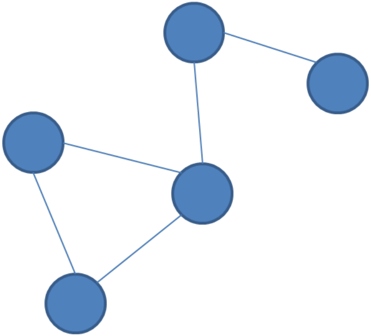
- Try to learn the propagation scheme of the labels
  (instead of making assumptions about how the labels propagate)
- Try lo learn propagation schemes between different labels (like trust $\Rightarrow \neg$spam)

Solution: extension of the Iterative Classification Algorithm

- Propagation is learnt through a classifier
- Iterative inference process

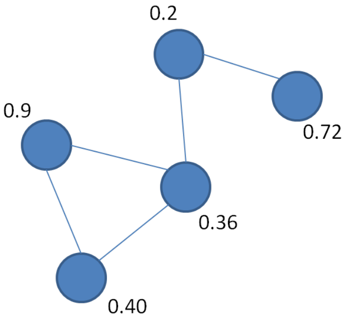
Iterative propagation (ICA [Lu et al. 2003] like)

- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
  - New score is given by a linear classifier: $<\theta, \Phi(n_i, \mathcal{N}_i)>$
  - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
  - Learning of θ made on a labeled graph.
- repeat. . .
- **No assumptions made on the diffusion process**



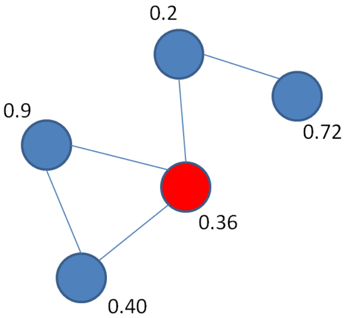
Iterative propagation (ICA [Lu et al. 2003] like)

- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
  - New score is given by a linear classifier: $< \theta, \Phi(n_i, \mathcal{N}_i) >$
  - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
  - Learning of θ made on a labeled graph.
- repeat. . .
- **No assumptions made on the diffusion process**



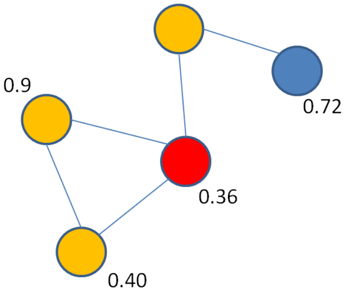
Iterative propagation (ICA [Lu et al. 2003] like)

- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
  - New score is given by a linear classifier: $< \theta, \Phi(n_i, \mathcal{N}_i) >$
  - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
  - Learning of θ made on a labeled graph.
- repeat. . .
- **No assumptions made on the diffusion process**



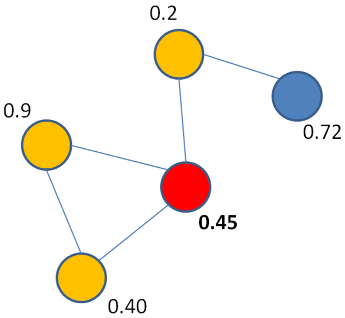
Iterative propagation (ICA [Lu et al. 2003] like)

- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
    - New score is given by a linear classifier: $< \theta, \Phi(n_i, \mathcal{N}_i) >$
    - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
    - Learning of θ made on a labeled graph.
- repeat. . .
- **No assumptions made on the diffusion process**



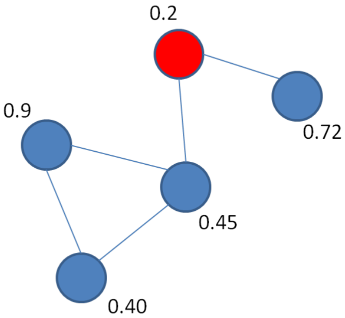
Iterative propagation (ICA [Lu et al. 2003] like)

- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
  - New score is given by a linear classifier: $< \theta, \Phi(n_i, \mathcal{N}_i) >$
  - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
  - Learning of θ made on a labeled graph.
- repeat. . .
- **No assumptions made on the diffusion process**



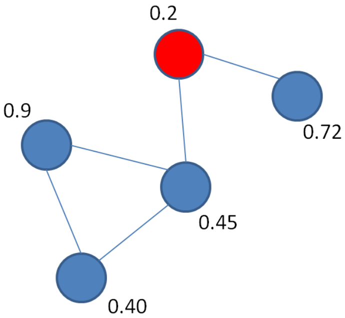
Iterative propagation (ICA [Lu et al. 2003] like)

- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
  - New score is given by a linear classifier: $< \theta, \Phi(n_i, \mathcal{N}_i) >$
  - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
  - Learning of θ made on a labeled graph.
- repeat...
- **No assumptions made on the diffusion process**



Iterative propagation (ICA [Lu et al. 2003] like)

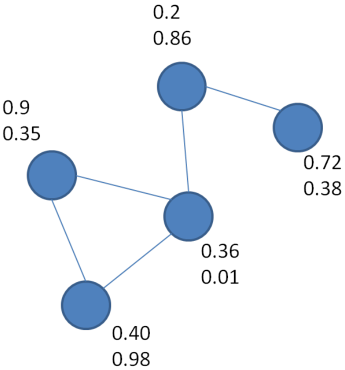
- A graph to label
- Initial labeling made with instance-based model *(RankBoost here)*
- Iterative inference process:
  - Pick randomly a node n_i
  - Consider the neighbourhood $\mathcal{N}_i$ of this node
  - Compute a new score using the neighboring information:
  - New score is given by a linear classifier: $< \theta, \Phi(n_i, \mathcal{N}_i) >$
  - $\Phi(n_i, \mathcal{N}_i)$ is a features vector of n_i and $\mathcal{N}_i$
  - Learning of θ made on a labeled graph.
- repeat. . .
- **No assumptions made on the diffusion process**



Multi-Relational Iterative Classification/Ranking

- Consider a multi-label graph
- Multi-label graph = Set of real valued graphs
- With additional relations

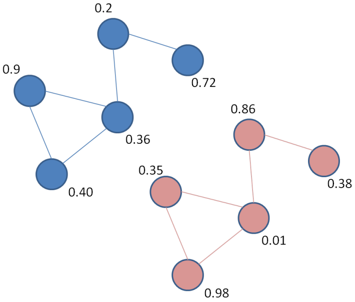
- The model is able to learn complex *inter-categories* and *inter-documents* propagation schemes
- See *Iterative Annotation of Multi-relational Social Networks* S. Peters, L. Denoyer and P. Gallinari. In ASONAM 2010 for details.



Multi-Relational Iterative Classification/Ranking

- Consider a multi-label graph
- Multi-label graph = Set of real valued graphs
- With additional relations

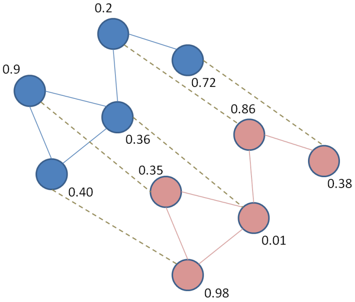
- The model is able to learn complex *inter-categories* and *inter-documents* propagation schemes
- See *Iterative Annotation of Multi-relational Social Networks* S. Peters, L. Denoyer and P. Gallinari. In ASONAM 2010 for details.



Multi-Relational Iterative Classification/Ranking

- Consider a multi-label graph
- Multi-label graph = Set of real valued graphs
- With additional relations

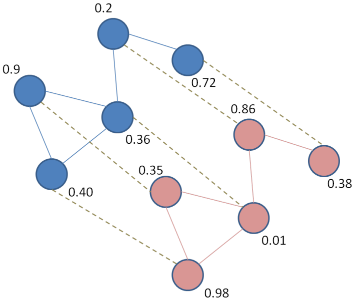
- The model is able to learn complex *inter-categories* and *inter-documents* propagation schemes
- See *Iterative Annotation of Multi-relational Social Networks* S. Peters, L. Denoyer and P. Gallinari. In ASONAM 2010 for details.



Multi-Relational Iterative Classification/Ranking

- Consider a multi-label graph
- Multi-label graph = Set of real valued graphs
- With additional relations

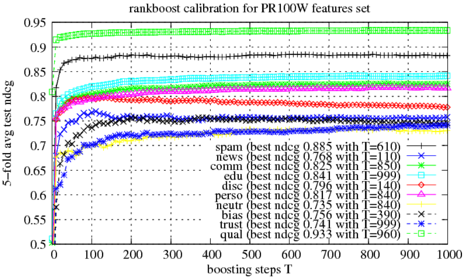
- The model is able to learn complex *inter-categories* and *inter-documents* propagation schemes
- See *Iterative Annotation of Multi-relational Social Networks* S. Peters, L. Denoyer and P. Gallinari. In ASONAM 2010 for details.



Experiments and Results

Experiments

- 5-fold cross validation



Data preprocessing (with intensive use of sort, sed , join, awk, perl, c++. . .)

- graph reweighting according to cosine similarity between hosts
- added feature: weight sensitive Pagerank `PR set`
  - PR_90: almost traditional PR
  - PR_10: mostly cosine-based PR
- Used 100, 1000, and then 70000 words as sparse features to train RB `PR100W,PR1K,PRAll`

Top spam features according to RankBoost

feature	α^f
top_1000_query_prec_hp	1.99
PR_D0.80	1.29
top_100_query_prec_hp	1.16
"http"	1.05
frac_visible_hp	1.03
top_100_query_prec_avg	0.99
num_title_words_std	0.89
frac_anchor_hp	0.82
...	
PR_D0.20	-0.66
num_title_words_mp	-0.75
top_1000_corpus_rec_hp	-0.78
frac_anchor_avg	-0.89
"money"	-0.92
compress_rate_hp	-0.96
log(siteneighbors3/pagerank)_hp	-1.01
avg_length_avg	-1.25
top_500_corpus_prec_avg	-1.57

Table: Most informative features for spam according to RankBoost cumulative weighting.

Web Quality ranking results

	method	task 1	task 2 (en)	task 3 (de)	task 3 (fr)
Rank-Boost	basic	0.657	0.905	0.797	0.821
	PR	0.619	0.916	0.803	0.818
	nPR	0.632	0.911	0.816	0.820
	PR1K	0.649	0.923	**0.821**	**0.845**
	PR100W	0.632	0.918	0.805	0.824
Propa-gation	Reg/PRall	0.696	0.835	0.803	0.794
	Iter/PR1K	**0.701**	**0.923**	0.816	0.836

Table: Results on the full test set.

	method	task 1	task 2 (en)	task 3 (de)	task 3 (fr)
Propa-gation	Reg/PRall	**0.702**	0.817	**0.852**	0.797
	Iter/PR1K	0.659	0.930	0.835	0.823

Table: Results on the validation subset.

Per category results

method	spam	news	com	edu	disc	pers	neut	bias	trus
basic	**0.810**	0.632	0.750	0.782	0.763	0.731	0.511	0.462	0.471
PR	0.682	0.592	0.758	0.803	0.683	0.656	0.406	0.552	0.440
nPR	0.764	0.634	0.784	0.809	0.704	0.741	0.433	0.363	0.453
PR1K	0.677	0.685	0.816	**0.880**	0.759	0.656	0.462	0.496	0.409
PR100W	0.769	0.633	0.809	0.823	0.698	0.737	0.425	0.350	0.443
Reg/PRall	0.786	0.647	0.825	0.833	**0.777**	0.765	**0.563**	**0.568**	0.495
Iter/PR1K	0.7761	**0.737**	**0.830**	0.859	0.739	**0.768**	0.547	0.551	**0.502**

Table: Per category results for task 1 on the full test set.

Conclusion & Questions

Thanks for this challenge!

Any question ?