Semi-supervised Ranking via Preference Regularization

Martin Szummer **_Emine Yilmaz_**

Microsoft Research

The Problem

Input: 1) labeled set L : items with features $\boldsymbol{x}$,

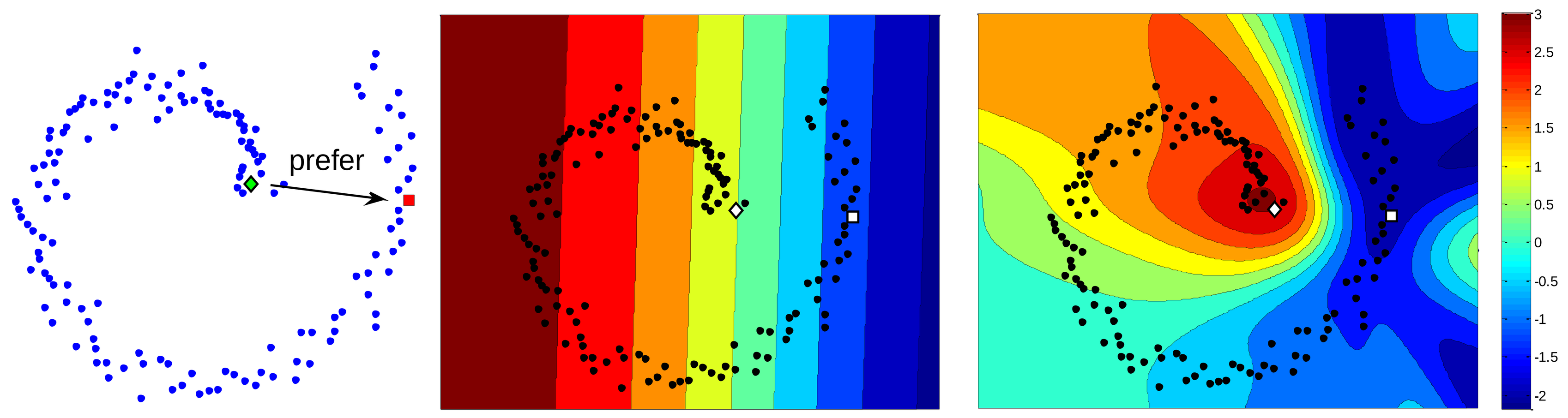
a (partial) preference ordering: item $i \succ$ item $j \succ$ item k

OR ordinal labels l

2) unlabeled set U : items with features $\boldsymbol{x}$

Output: **ranking function** $f(\boldsymbol{x};\ \boldsymbol{w})$.

Assigns **ranks** r by sorting items by **score** $s = f(\boldsymbol{x};\ \boldsymbol{w})$.



Training data — f learned from only labeled data — f learned from both labeled & unlabeled data

Applications

Information retrieval: search engine ranking where little labeled training data is available, or expensive to get:

languages of small markets, special-interest domains,
company-specific search, user relevance feedback

How to benefit from unlabeled data?

Unlabeled data gives information about the data distribution $P(\boldsymbol{x})$.

We must make assumptions about what the structure of the unlabeled data tells us about the ranking function

A common assumption: the **cluster assumption**

Unlabeled data defines the extent of clusters,

Labeled data determines the class/function value of each cluster

Semi-supervised classification:	similar items	⇒	same class
regression:	similar items	⇒	similar value
ranking:	similar items	⇒	similar preference

Preference Regularization

similar items i, j $\Rightarrow$ want $i \not\succ j$ and $j \not\succ i$ (indifference)

is a type of **regularizer** on the function we are learning.

(1) Input similarity: probability of transitioning from i to j under a noise model

$$\hat{q}_{j|i} = \begin{cases} \mathrm{e}^{-d_{ij}^2/\sigma_i^2} / \sum_{k \in N_K(i)} \mathrm{e}^{-d_{ik}^2/\sigma_i^2} & \text{if } j \in N_K(i), \\ 0 & \text{otherwise.} \end{cases}$$

$\hat{q}_{ij} = \hat{q}_{i|j}\hat{q}_{j|i} / Z$ where Z normalizes to sum to 1.

Follow **manifold structure**, by only transitioning to K nearest neighbors.

(2) Probability of indifference : $P(i \not\succ j)P(j \not\succ i)$

Link the probabilistic input similarity to the probability of indifference via KL-divergence.

Definition: The **preference regularizer** is

$$\sum_{(i,j)\in U} \hat{q}_{ij} \log P(i \not\succ j)P(j \not\succ i).$$

Preference indifference is a weaker constraint than regularizing function values or classes.

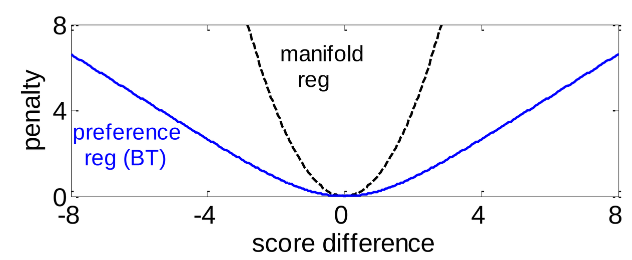
Semi-supervised Bradley-Terry

Bradley-Terry model of preference: a logistic of score difference of i & j

$$P(i \succ j) = 1/(1 + \mathrm{e}^{-(s_i - s_j)}).$$

Likelihood

$$C = \sum_{(i,j)\in L} \mathbb{I}_{i \succ j} \log P(i \succ j) + \beta \underbrace{\sum_{i,j \in U} \hat{q}_{ij} \log P(i \not\succ j)P(j \not\succ i)}_{}.$$



$$\log \frac{0.5}{1 + \cosh(s_i - s_j)}$$

Learning to rank algorithm:

choose a neural net function for the scores $s_j = f(\boldsymbol{x}_j;\ \boldsymbol{w})$ (or any differentiable function). Estimate parameters $\boldsymbol{w}$ by max likelihood & grad descent

Previous work:

a) self-training approaches [Li, Li, Zhou 09]

b) incorporate unlabelled data by learning features, then apply a regular (supervised) ranking algorithms. [Duh, Kirchoff 08]

References:

Burges; .. *Learning to rank with nonsmooth cost functions*. NIPS 2006

Rank-sensitive Bradley-Terry models

Objectives in information retrieval (e.g. NDCG, mean avg precision) depend on:

- sorted order of items
- ranks of items: weight the top of the ranking more

$$\mathrm{NDCG} = \frac{1}{\mathrm{DCG}_{\max}} \sum_i L(l_i) R(r_i)$$

label gain $2^{l_i} - 1$ rank discount $|\ 1/\log(1+r_i)\ |$

Swap cost: cost if item i and j were swapped in the ranking

labeled $|\Delta_{ij}| = |L(l_i)R(r_i) + L(l_j)R(r_j) - L(l_i)R(r_j) - L(l_j)R(r_i)|$

unlabeled $|\Delta_{ij}^U| = |R(r_i) - R(r_j)|.$

LambdaRank objective $C = \sum_{(i,j)\in L} |\Delta_{ij}|\, \mathbb{I}_{i \succ j} \log P(i \succ j)$

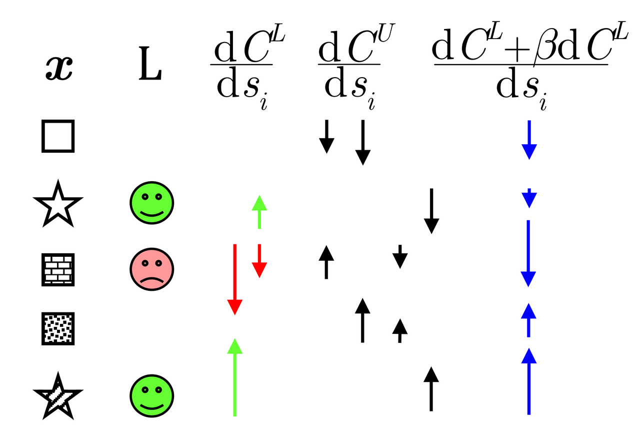
Semi-supervised LambdaRank

$$C = \sum_{(i,j)\in L} |\Delta_{ij}|\, \mathbb{I}_{i \succ j} \log P(i \succ j) + \beta \sum_{i,j \in U} |\Delta_{ij}^U|\, \hat{q}_{ij} \log P(i \not\succ j)P(j \not\succ i),$$

with gradients $\forall i$

$$\mathrm{d}C/\mathrm{d}s_i = \sum_{\{j|(i,j)\in L\}} |\Delta_{ij}|\,(\mathbb{I}_{i \succ j} - P(i \succ j)) + \beta \sum_{j \in U} |\Delta_{ij}^U|\, \hat{q}_{ij}\,(0.5 - P(i \succ j)),$$

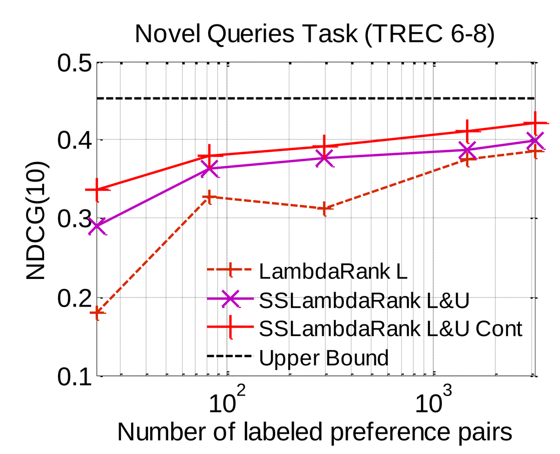
Example gradients:

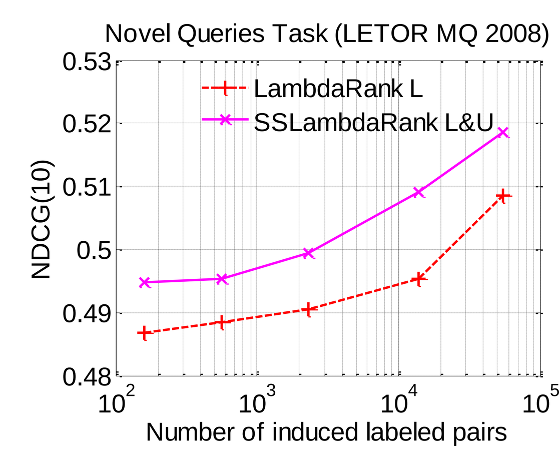


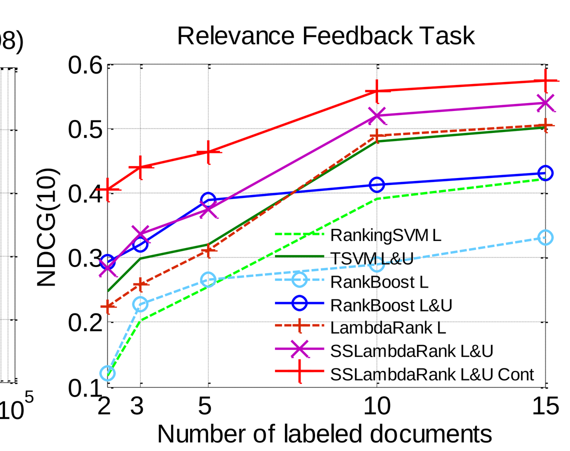
near linear time.
Can use massive unlabeled sets

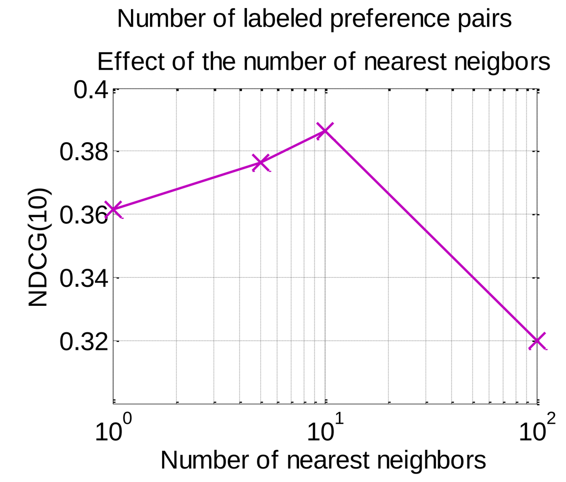
Advantages: 1) End-to-end optimization of ranking metrics 2) Single-stage learning w labeled & unlabeled data 3) Rank instances can be completely unlabeled 4) Linear time 5) Accurate: based on winner in Yahoo 2010 ranking challenge 6) General

Experiments









TREC 6-8 has 90 queries, with 1000 docs/query.

Features for supervised learning: BM25, language modeling, functions of TF and IDF

For unsupervised learning: select query-specific terms. $d(\mathbf{x}_i,\ \mathbf{x}_j)$ is TF-IDF distance.