2018 EE448, Big Data Mining, Lecture 9

Learning to Rank

Weinan Zhang

Shanghai Jiao Tong University

http://wnzhang.net

http://wnzhang.net/teaching/ee448/index.html

Content of This Course

- Another ML problem: ranking
- Learning to rank
- Pointwise methods
- Pairwise methods
- Listwise methods

Ranking Problem

Learning to rank

Pointwise methods

Pairwise methods

Listwise methods

Sincerely thank Dr. Tie-Yan Liu

The Probability Ranking Principle

- https://nlp.stanford.edu/IR-book/html/htmledition/the-probability-ranking-principle-1.html

Regression and Classification

- Supervised learning

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i, f_\theta(x_i))$$

- Two major problems for supervised learning
  - Regression

$$\mathcal{L}(y_i, f_\theta(x_i)) = \frac{1}{2}(y_i - f_\theta(x_i))^2$$

  - Classification

$$\mathcal{L}(y_i, f_\theta(x_i)) = -y_i \log f_\theta(x_i) - (1 - y_i) \log(1 - f_\theta(x_i))$$

Learning to Rank Problem

- Input: a set of instances

$$X = \{x_1, x_2, \ldots, x_n\}$$

- Output: a rank list of these instances

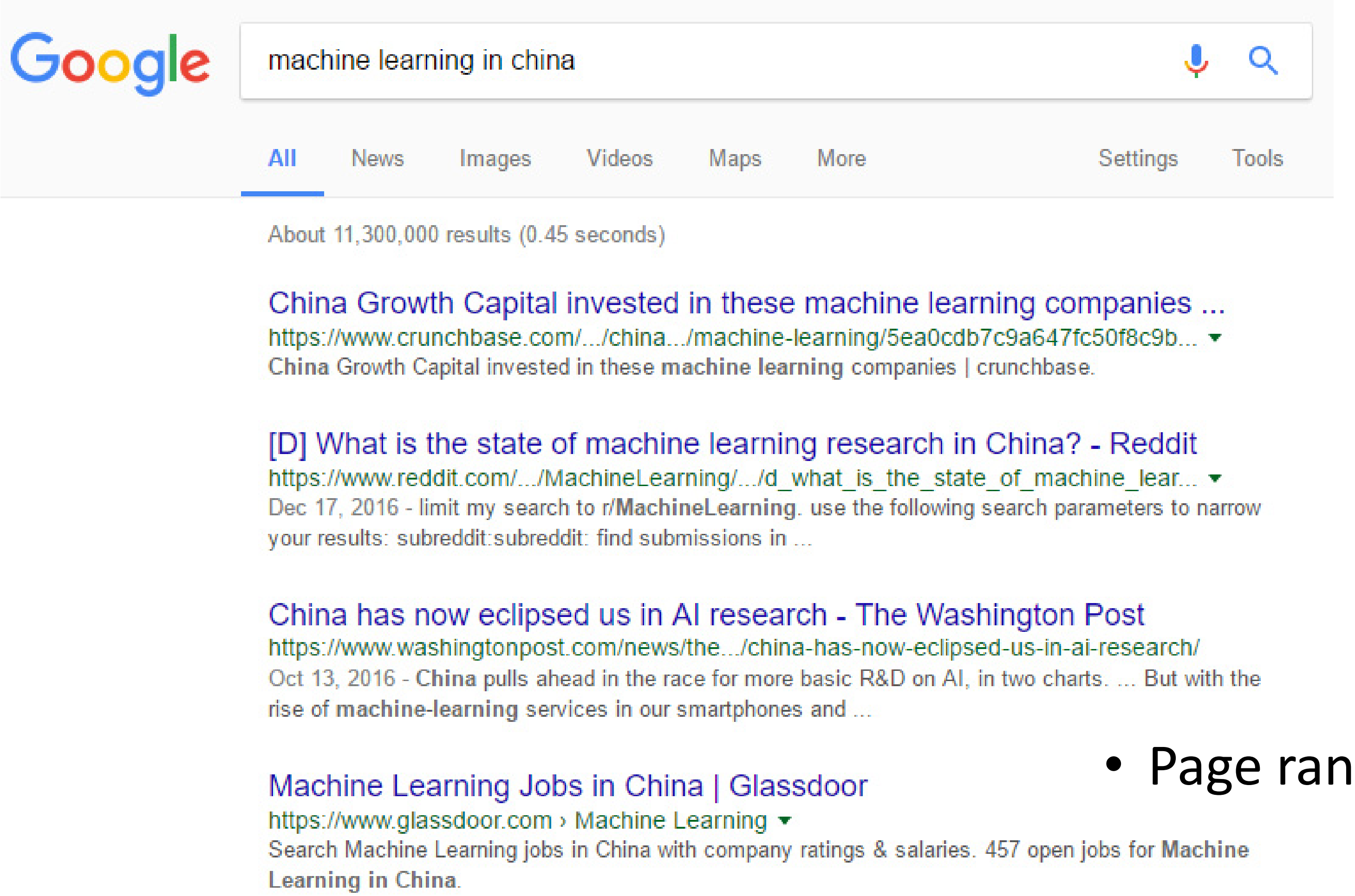
$$\hat{Y} = \{x_{r_1}, x_{r_2}, \ldots, x_{r_n}\}$$

- Ground truth: a correct ranking of these instances

$$Y = \{x_{y_1}, x_{y_2}, \ldots, x_{y_n}\}$$

A Typical Application

- Webpage ranking given a query



machine learning in china

About 11,300,000 results (0.45 seconds)

China Growth Capital invested in these machine learning companies ...
https://www.crunchbase.com/.../china.../machine-learning/5ea0cdb7c9a647fc50f8c9b...
China Growth Capital invested in these **machine learning** companies | crunchbase.

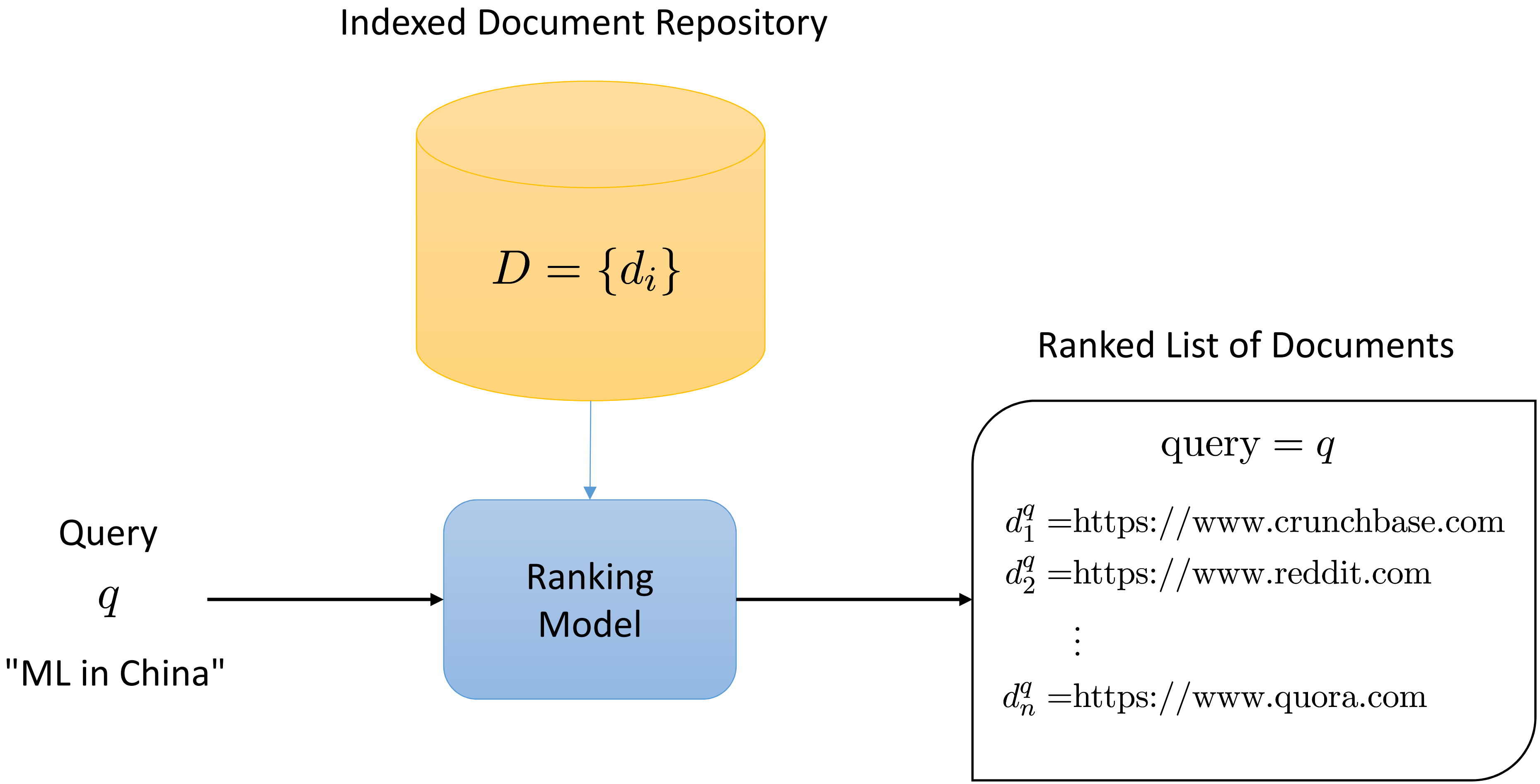
[D] What is the state of machine learning research in China? - Reddit
https://www.reddit.com/.../MachineLearning/.../d_what_is_the_state_of_machine_lear...
Dec 17, 2016 - limit my search to r/**MachineLearning**. use the following search parameters to narrow your results: subreddit:subreddit: find submissions in ...

China has now eclipsed us in AI research - The Washington Post
https://www.washingtonpost.com/news/the.../china-has-now-eclipsed-us-in-ai-research/
Oct 13, 2016 - **China** pulls ahead in the race for more basic R&D on AI, in two charts. ... But with the rise of **machine-learning** services in our smartphones and ...

Machine Learning Jobs in China | Glassdoor
https://www.glassdoor.com › Machine Learning
Search Machine Learning jobs in China with company ratings & salaries. 457 open jobs for **Machine Learning in China**.

- Page ranking

Webpage Ranking



Indexed Document Repository

$D = \{d_i\}$

Query

q

"ML in China"

Ranking Model

Ranked List of Documents

$\text{query} = q$

d_1^q =https://www.crunchbase.com

d_2^q =https://www.reddit.com

$\vdots$

d_n^q =https://www.quora.com

Model Perspective

- In most existing work, learning to rank is defined as having the following two properties
    - Feature-based
        - Each instance (e.g. query-document pair) is represented with a list of features
    - Discriminative training
        - Estimate the relevance given a query-document pair
        - Rank the documents based on the estimation

$$y_i = f_\theta(x_i)$$

Learning to Rank

- Input: features of query and documents
  - Query, document, and combination features

- Output: the documents ranked by a scoring function

$$y_i = f_\theta(x_i)$$

- Objective: relevance of the ranking list
  - Evaluation metrics: NDCG, MAP, MRR…

- Training data: the query-doc features and relevance ratings

Training Data

- The query-doc features and relevance ratings

Query='ML in China'				Features		
Rating	**Document**	**Query Length**	**Doc PageRank**	**Doc Length**	**Title Rel.**	**Content Rel.**
3	d_1=http://crunchbase.com	0.30	0.61	0.47	0.54	0.76
5	d_2=http://reddit.com	0.30	0.81	0.76	0.91	0.81
4	d_3=http://quora.com	0.30	0.86	0.56	0.96	0.69

		Query features	Document features		Query-doc features	

Learning to Rank Approaches

- Learn (not define) a scoring function to optimally rank the documents given a query

- Pointwise
  - Predict the absolute relevance (e.g. RMSE)
- Pairwise
  - Predict the ranking of a document pair (e.g. AUC)
- Listwise
  - Predict the ranking of a document list (e.g. Cross Entropy)

Tie-Yan Liu. Learning to Rank for Information Retrieval. Springer 2011.

http://www.cda.cn/uploadfile/image/20151220/20151220115436_46293.pdf

Pointwise Approaches

- Predict the expert ratings
  - As a regression problem

$$y_i = f_\theta(x_i)$$

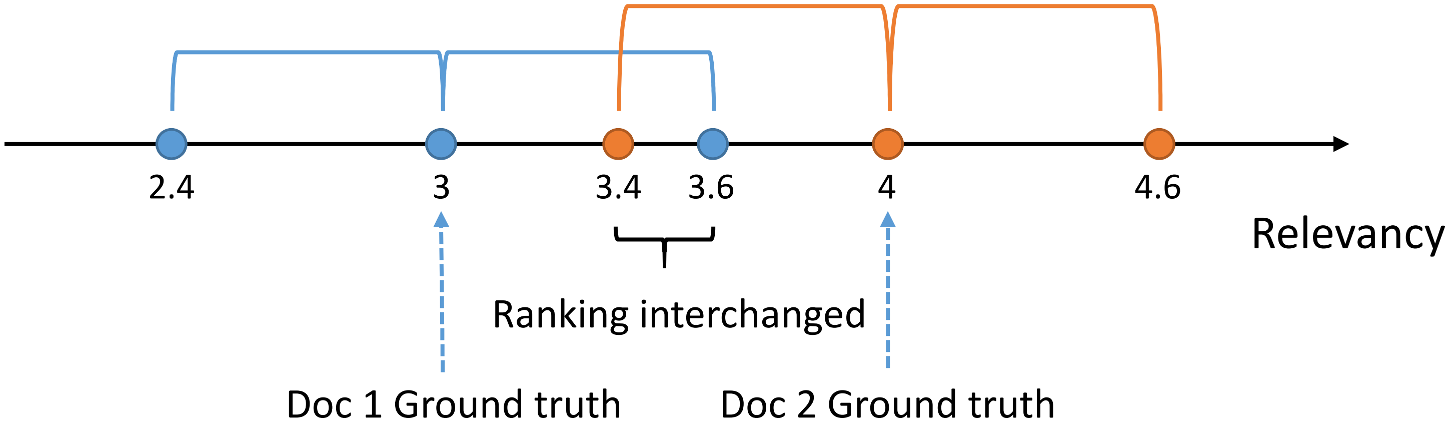
$$\min_\theta \frac{1}{2N} \sum_{i=1}^{N} (y_i - f_\theta(x_i))^2$$

Query='ML in China'

Features

Rating	Document	Query Length	Doc PageRank	Doc Length	Title Rel.	Content Rel.
3	d_1=http://crunchbase.com	0.30	0.61	0.47	0.54	0.76
5	d_2=http://reddit.com	0.30	0.81	0.76	0.91	0.81
4	d_3=http://quora.com	0.30	0.86	0.56	0.96	0.69

Point Accuracy != Ranking Accuracy



- Same square error might lead to different rankings

Pairwise Approaches

- Not care about the absolute relevance but the relative preference on a document pair

- A binary classification

$$\overset{q^{(i)}}{\begin{bmatrix} d_1^{(i)}, 5 \\ d_2^{(i)}, 3 \\ \vdots \\ d_{n^{(i)}}^{(i)}, 2 \end{bmatrix}} \xrightarrow{\text{Transform}} \overset{q^{(i)}}{\left\{ \underset{5 > 3}{(d_1^{(i)}, d_2^{(i)})}, \underset{5 > 2}{(d_1^{(i)}, d_{n^{(i)}}^{(i)})}, \ldots, \underset{3 > 2}{(d_2^{(i)}, d_{n^{(i)}}^{(i)})} \right\}}$$

Binary Classification for Pairwise Ranking

- Given a query q and a pair of documents (d_i, d_j)
- Target probability $y_{i,j} = \begin{cases} 1 & \text{if } i \rhd j \\ 0 & \text{otherwise} \end{cases}$
- Modeled probability

$$P_{i,j} = P(d_i \rhd d_j | q) = \frac{\exp(o_{i,j})}{1 + \exp(o_{i,j})}$$

$$o_{i,j} \equiv f(x_i) - f(x_j)$$

x_i is the feature vector of (*q*, *d_i*)

- Cross entropy loss

$$\mathcal{L}(q, d_i, d_j) = -y_{i,j} \log P_{i,j} - (1 - y_{i,j}) \log(1 - P_{i,j})$$

RankNet

- The scoring function $f_\theta(x_i)$ is implemented by a neural network
- Modeled probability $P_{i,j} = P(d_i \rhd d_j | q) = \dfrac{\exp(o_{i,j})}{1 + \exp(o_{i,j})}$

  $o_{i,j} \equiv f(x_i) - f(x_j)$
- Cross entropy loss

$$\mathcal{L}(q, d_i, d_j) = -y_{i,j} \log P_{i,j} - (1 - y_{i,j}) \log(1 - P_{i,j})$$

- Gradient by chain rule

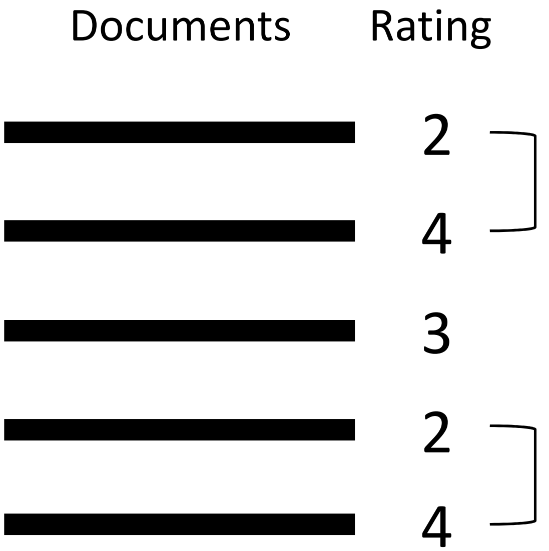
$$\frac{\partial \mathcal{L}(q, d_i, d_j)}{\partial \theta} = \frac{\partial \mathcal{L}(q, d_i, d_j)}{\partial P_{i,j}} \frac{\partial P_{i,j}}{\partial o_{i,j}} \frac{\partial o_{i,j}}{\partial \theta}$$

$$= \frac{\partial \mathcal{L}(q, d_i, d_j)}{\partial P_{i,j}} \frac{\partial P_{i,j}}{\partial o_{i,j}} \Big(\frac{\partial f_\theta(x_i)}{\partial \theta} - \overset{\substack{\text{BP in NN} \\ \downarrow}}{\frac{\partial f_\theta(x_j)}{\partial \theta}} \Big)$$

Burges, Christopher JC, Robert Ragno, and Quoc Viet Le. "Learning to rank with nonsmooth cost functions." *NIPS*. Vol. 6. 2006.

Shortcomings of Pairwise Approaches

- Each document pair is regarded with the same importance



Same pair-level error but different list-level error

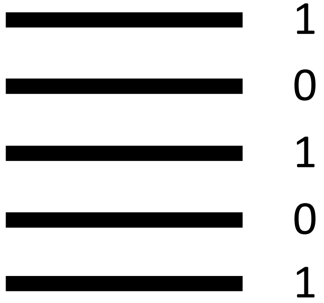
Ranking Evaluation Metrics

- For binary labels $y_i = \begin{cases} 1 & \text{if } d_i \text{ is relevant with } q \\ 0 & \text{otherwise} \end{cases}$

- Precision@*k* for query *q*

$$P@k = \frac{\#\{\text{relevant documents in top } k \text{ results}\}}{k}$$

- Average precision for query *q*



$$AP = \frac{\sum_k P@k \cdot y_{i(k)}}{\#\{\text{relevant documents}\}}$$

  - *i*(*k*) is the document id at *k*-th position

$$AP = \frac{1}{3} \cdot \left(\frac{1}{1} + \frac{2}{3} + \frac{3}{5}\right)$$

- Mean average precision (MAP): average over all queries

Ranking Evaluation Metrics

- For score labels, e.g.,

$$y_i \in \{0, 1, 2, 3, 4\}$$

- Normalized discounted cumulative gain (NDCG@*k*) for query *q*

$$NDCG@k = Z_k \sum_{j=1}^{k} \frac{2^{y_{i(j)}} - 1}{\log(j+1)}$$

Gain

Discount

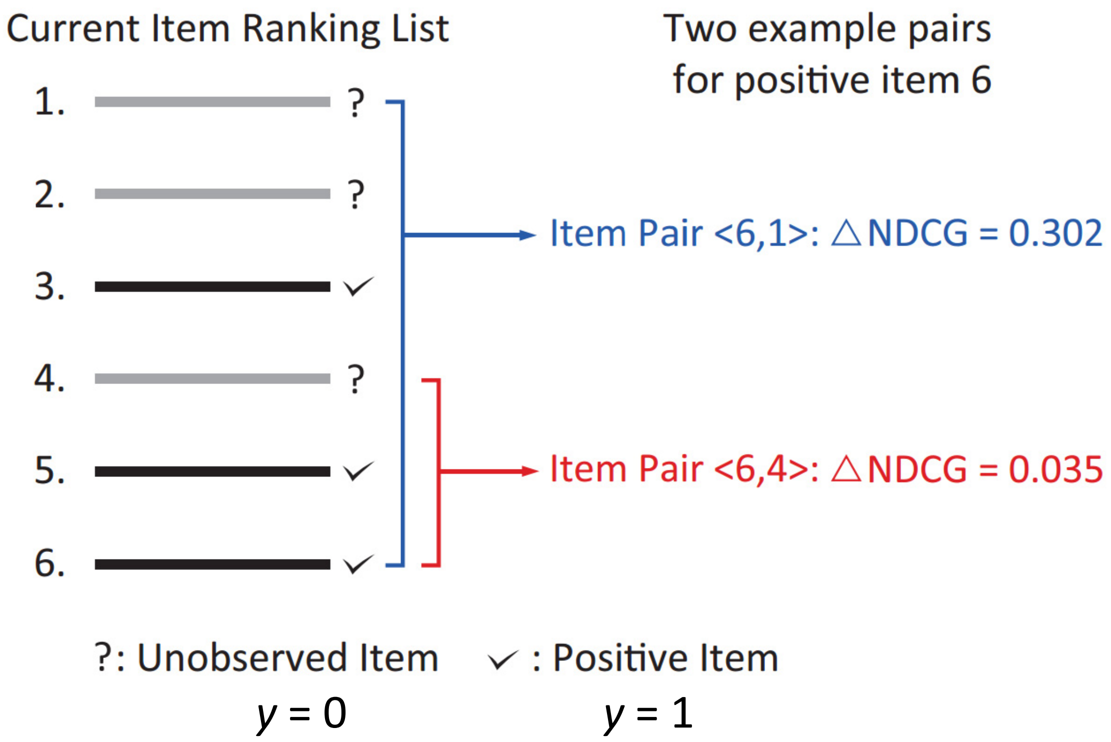
Normalizer

  - *i*(*j*) is the document id at *j*-th position
  - Z_k is set to normalize the DCG of the ground truth ranking as 1

Shortcomings of Pairwise Approaches

- Same pair-level error but different list-level error

$$NDCG@k = Z_k \sum_{j=1}^{k} \frac{2^{y_{i(j)}} - 1}{\log(j+1)}$$



Current Item Ranking List

Two example pairs for positive item 6

1. ?

2. ?

3. ✓

4. ?

5. ✓

6. ✓

Item Pair <6,1>: △NDCG = 0.302

Item Pair <6,4>: △NDCG = 0.035

?: Unobserved Item $y = 0$

✓ : Positive Item $y = 1$

Listwise Approaches

- Training loss is directly built based on the difference between the prediction list and the ground truth list

- Straightforward target
  - Directly optimize the ranking evaluation measures

- Complex model

ListNet

- Train the score function $y_i = f_\theta(x_i)$
- Rankings generated based on $\{y_i\}_{i=1\ldots n}$
- Each possible *k*-length ranking list has a probability

$$P_f([j_1, j_2, \ldots, j_k]) = \prod_{t=1}^{k} \frac{\exp(f(x_{j_t}))}{\sum_{l=t}^{n} \exp(f(x_{j_l}))}$$

- List-level loss: cross entropy between the predicted distribution and the ground truth

$$\mathcal{L}(\boldsymbol{y}, f(\boldsymbol{x})) = -\sum_{g \in \mathcal{G}_k} P_y(g) \log P_f(g)$$

- Complexity: many possible rankings

Cao, Zhe, et al. "Learning to rank: from pairwise approach to listwise approach." *Proceedings of the 24th international conference on Machine learning*. ACM, 2007.

Distance between Ranked Lists

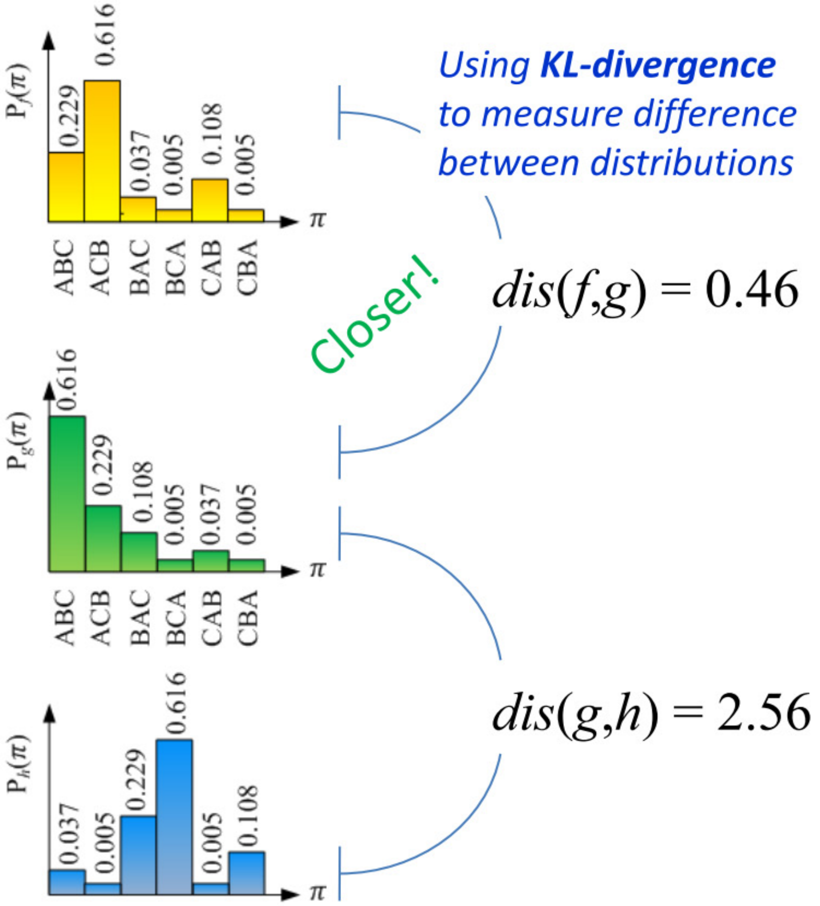
- A similar distance: KL divergence

$\varphi = exp$

f: $f(A) = 3, f(B)=0, f(C)=1$; ↔
Ranking by f: ABC

g: $g(A) = 6, g(B)=4, g(C)=3$; ↔
Ranking by g: ABC

h: $h(A) = 4, h(B)=6, h(C)=3$; ↔
Ranking by h: ACB



*Using **KL-divergence** to measure difference between distributions*

$dis(f,g) = 0.46$

$dis(g,h) = 2.56$

Pairwise vs. Listwise

- Pairwise approach shortcoming
  - Pair-level loss is away from IR list-level evaluations
- Listwise approach shortcoming
  - Hard to define a list-level loss under a low model complexity

- A good solution: LambdaRank
  - Pairwise training with listwise information

Burges, Christopher JC, Robert Ragno, and Quoc Viet Le. "Learning to rank with nonsmooth cost functions." *NIPS*. Vol. 6. 2006.

LambdaRank

- Pairwise approach gradient

$$o_{i,j} \equiv f(x_i) - f(x_j)$$

$$\frac{\partial \mathcal{L}(q, d_i, d_j)}{\partial \theta} = \underbrace{\frac{\partial \mathcal{L}(q, d_i, d_j)}{\partial P_{i,j}} \frac{\partial P_{i,j}}{\partial o_{i,j}}}_{\lambda_{i,j}} \Big(\frac{\partial f_\theta(x_i)}{\partial \theta} - \frac{\partial f_\theta(x_i)}{\partial \theta} \Big)$$

Pairwise ranking loss

Scoring function itself

Current ranking list

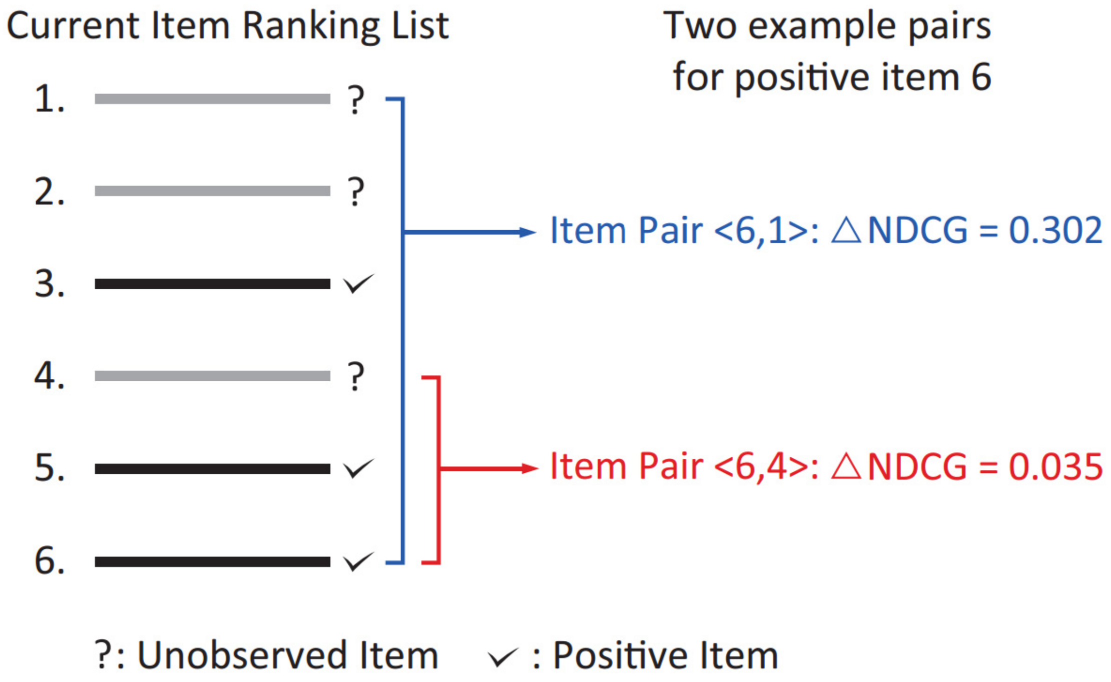
- LambdaRank basic idea
  - Add listwise information into $\lambda_{i,j}$ as $h(\lambda_{i,j}, g_q)$

$$\frac{\partial \mathcal{L}(q, d_i, d_j)}{\partial \theta} = h(\lambda_{i,j}, g_q) \Big(\frac{\partial f_\theta(x_i)}{\partial \theta} - \frac{\partial f_\theta(x_j)}{\partial \theta} \Big)$$

LambdaRank for Optimizing NDCG

- A choice of Lambda for optimize NDCG

$$h(\lambda_{i,j}, g_q) = \lambda_{i,j} \Delta NDCG_{i,j}$$



Current Item Ranking List

Two example pairs for positive item 6

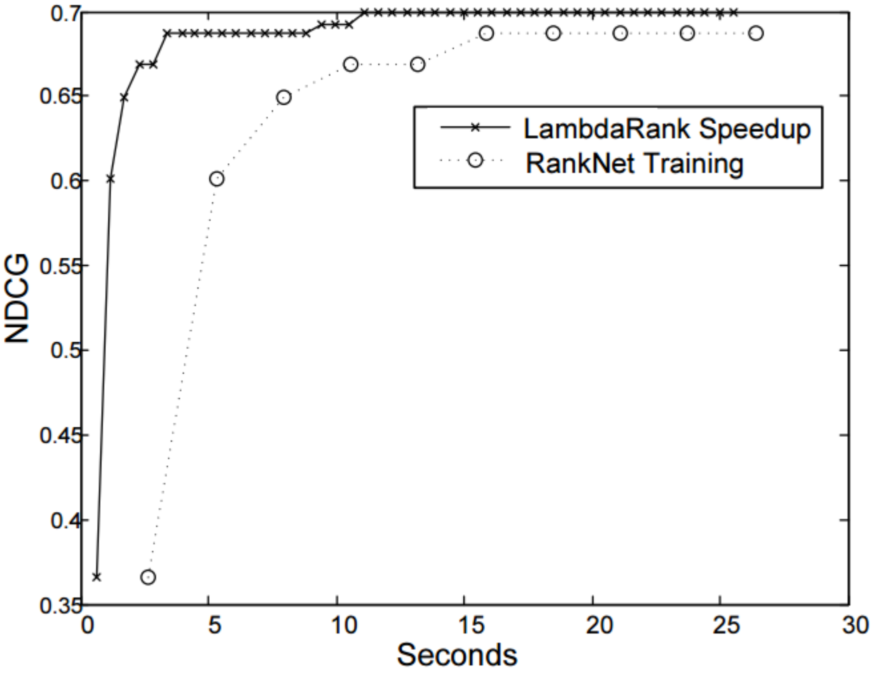
1. ?
2. ?
3. ✓
4. ?
5. ✓
6. ✓

Item Pair <6,1>: △NDCG = 0.302

Item Pair <6,4>: △NDCG = 0.035

?: Unobserved Item ✓ : Positive Item

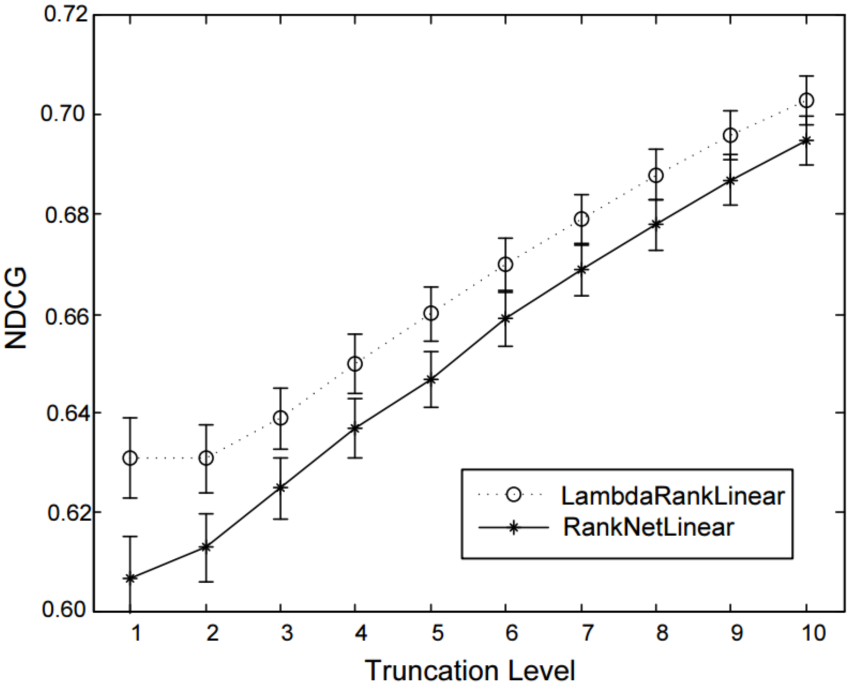
LambdaRank vs. RankNet

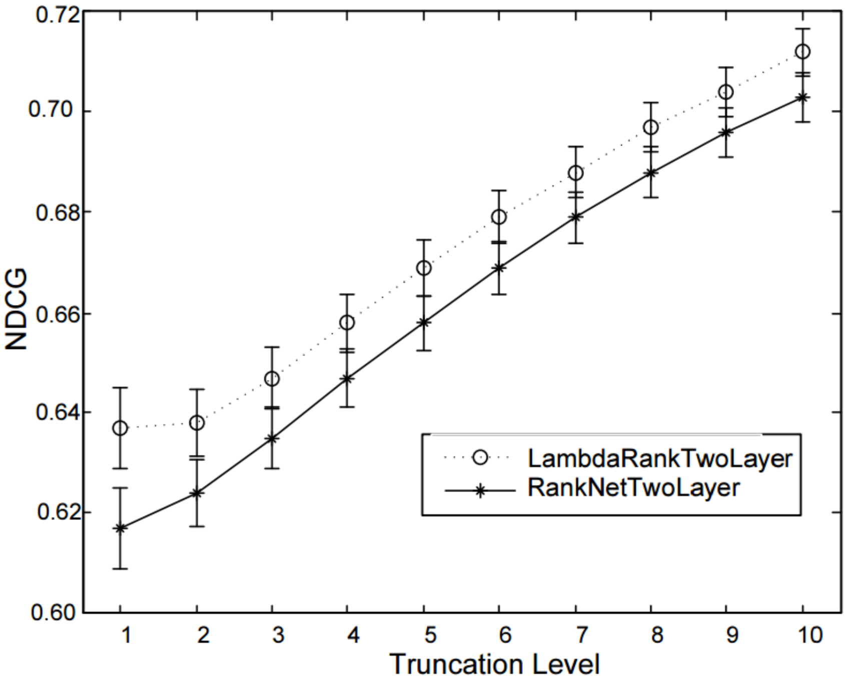


Linear nets

Burges, Christopher JC, Robert Ragno, and Quoc Viet Le. "Learning to rank with nonsmooth cost functions." *NIPS*. Vol. 6. 2006.

LambdaRank vs. RankNet





Burges, Christopher JC, Robert Ragno, and Quoc Viet Le. "Learning to rank with nonsmooth cost functions." *NIPS*. Vol. 6. 2006.

Summary of Learning to Rank

- Pointwise, pairwise and listwise approaches for learning to rank

- Pairwise approaches are still the most popular
    - A balance of ranking effectiveness and training efficiency

- LambdaRank is a pairwise approach with list-level information
    - Easy to implement, easy to improve and adjust