5Th Chinese Workshop on
Machine Learning and Applications
Nanjing
Nov. 10, 2007

Learning to Rank: From Pairwise Approach to Listwise Approach

Hang Li
Microsoft Research Asia

Joint work with Tie-Yan Liu, Jun Xu, and others

Talk Outline

- Ranking in Information Retrieval
- Learning to Rank
- ListNet: Probabilistic Model for Ranking
- AdaRank: Boosting Algorithm for Ranking

Means for Information Access

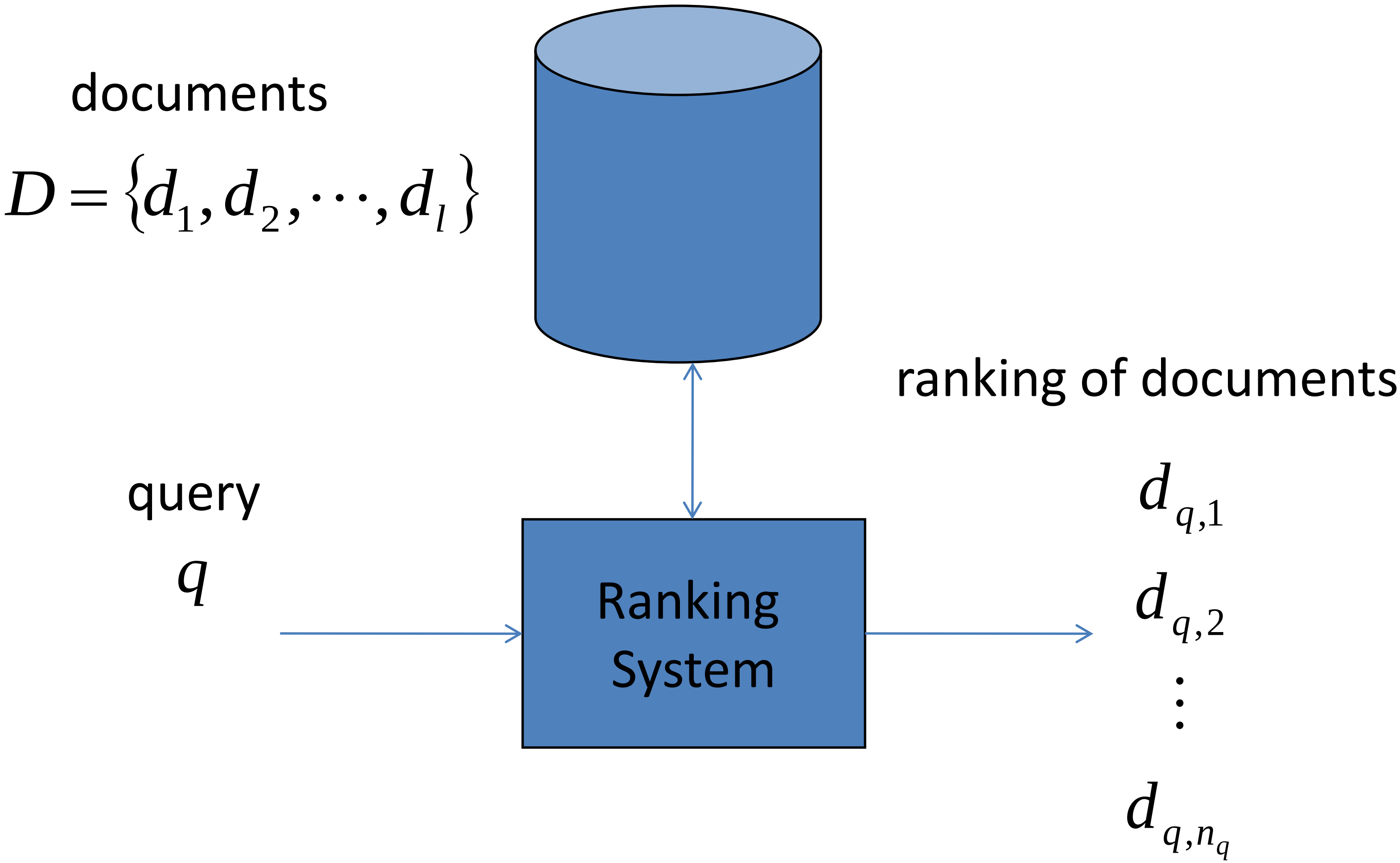
Information Extraction

Information Retrieval

Summarization

Current Approach
Information Retrieval = Ranking

Ranking Problem: Example = Document Retrieval



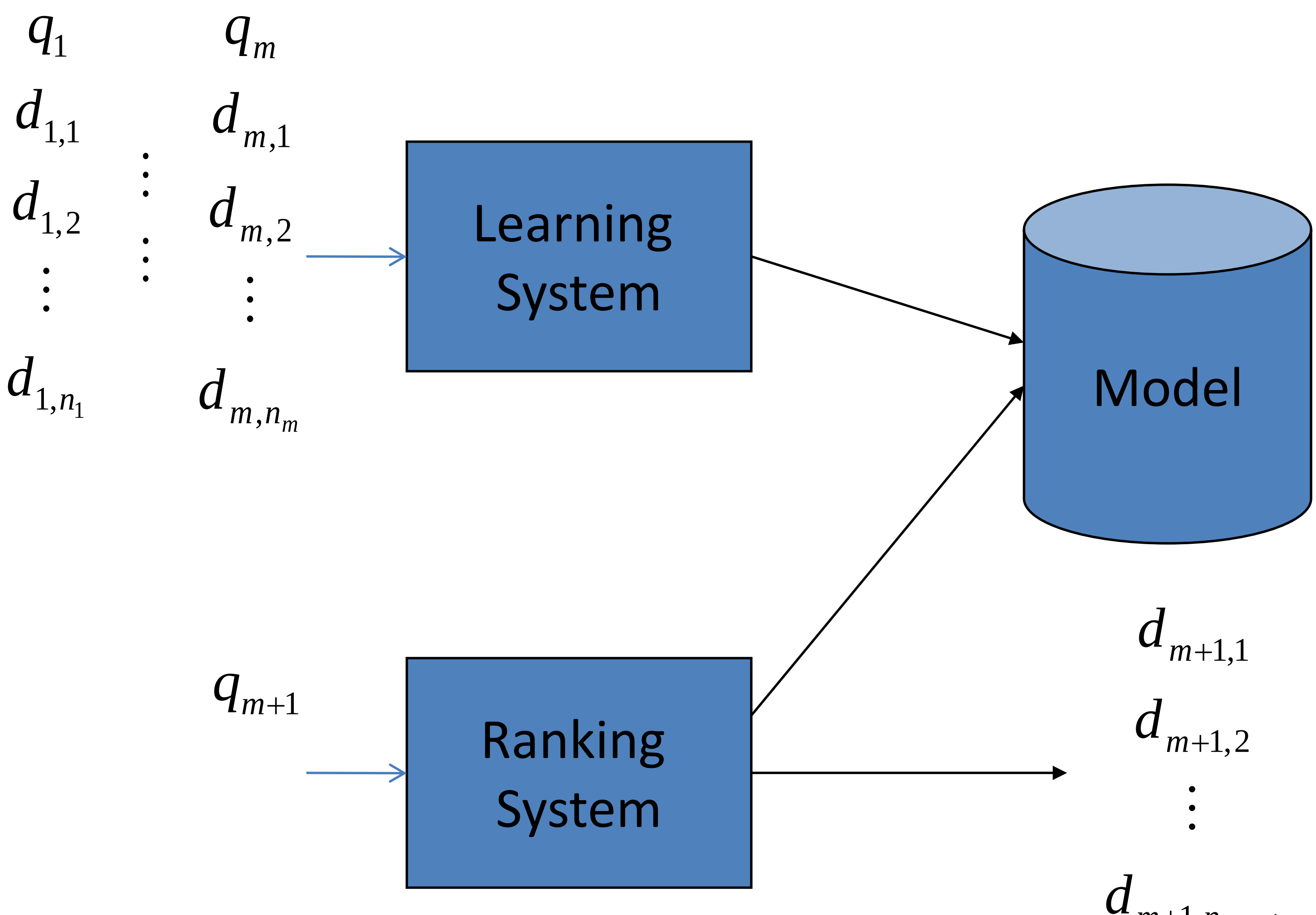
Ranking based on Relevance, Importance, Preference

Learning to Rank

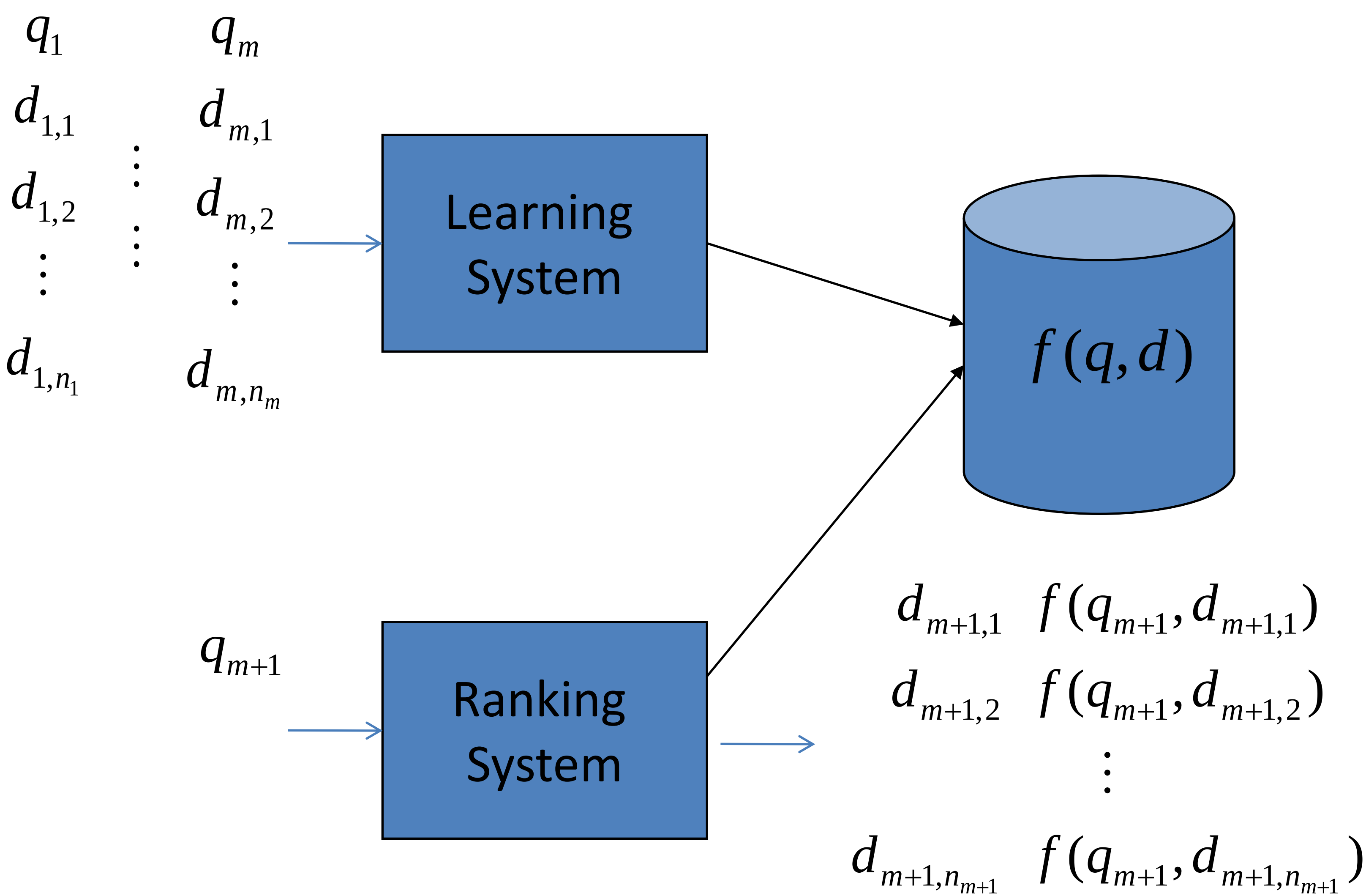
Issues of Learning to Rank

- General Framework
- Features
- Data Labeling
- Evaluation Measures
- Characteristics
- Existing Approaches

Learning to Rank



Score based Ranking



Learning Process

$$\begin{array}{c} q_1\left\{\begin{array}{l} d_{1,1} \\ d_{1,2} \\ \vdots \\ d_{1,n_1} \end{array}\right. \\ \vdots \\ \vdots \\ q_m\left\{\begin{array}{l} d_{m,1} \\ d_{m,2} \\ \vdots \\ d_{m,n_m} \end{array}\right. \end{array} \xrightarrow{\text{Data Labeling}} \begin{array}{c} q_1\left\{\begin{array}{ll} d_{1,1} & y_{1,1} \\ d_{1,2} & y_{1,2} \\ \vdots & \\ d_{1,n_1} & y_{1,n_1} \end{array}\right. \\ \vdots \\ \vdots \\ q_m\left\{\begin{array}{ll} d_{m,1} & y_{m,1} \\ d_{m,2} & y_{m,2} \\ \vdots & \\ d_{m,n_m} & y_{m,n_m} \end{array}\right. \end{array} \xrightarrow{\text{Feature Extraction}} \begin{array}{c} \left\{\begin{array}{ll} x_{1,1} & y_{1,1} \\ x_{1,2} & y_{1,2} \\ \vdots & \\ x_{1,n_1} & y_{1,n_1} \end{array}\right. \\ \vdots \\ \vdots \\ \left\{\begin{array}{ll} x_{m,1} & y_{m,1} \\ x_{m,2} & y_{m,2} \\ \vdots & \\ x_{m,n_m} & y_{m,n_m} \end{array}\right. \end{array} \xrightarrow{\text{Learning}} y = f(x)$$

Features

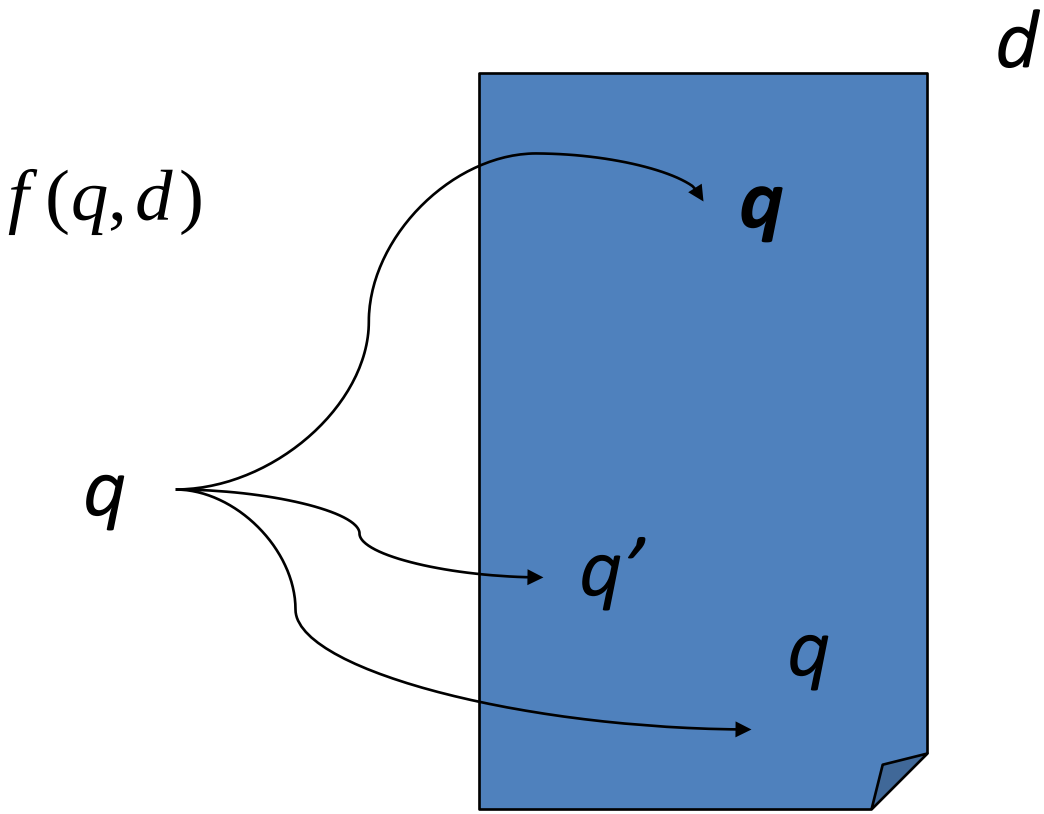
- Relevance: BM25
- Importance: PageRank

Relevance

- No rigorous definition
- Query = “soccer”, document = about soccer → document relevant
- Judgment by humans: several levels,

  e.g. “definitely relevant”, “partially relevant”

Key for Relevance: Matching between Query and Document

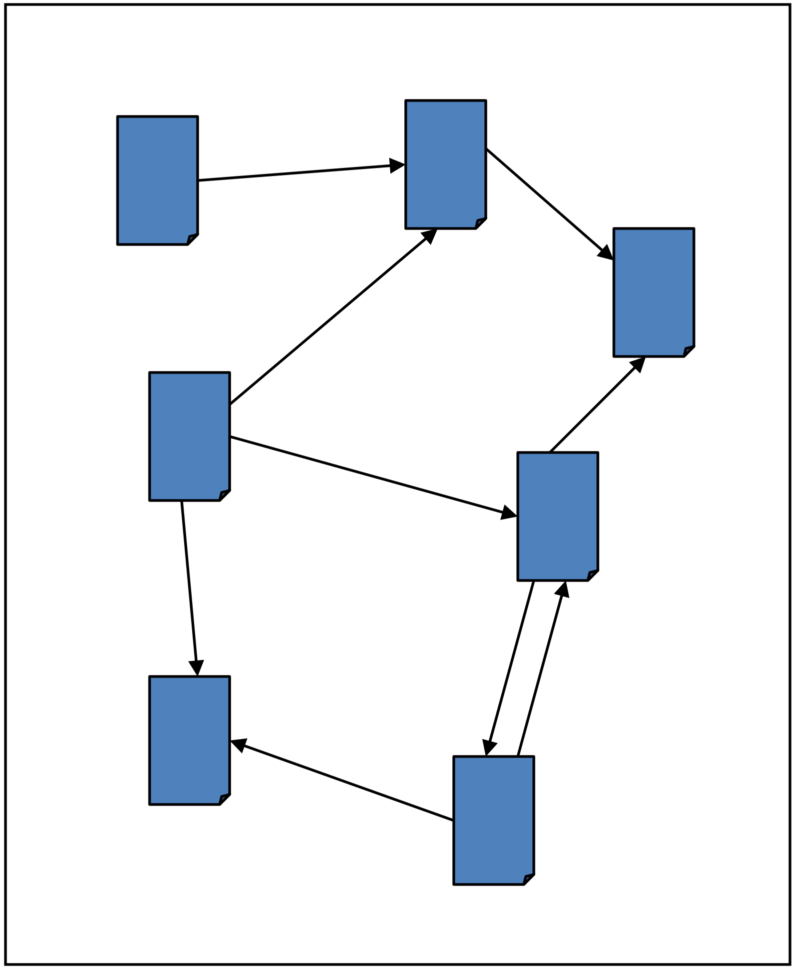


Importance

- No rigorous definition
- Citation, quality, freshness
- Independent of query

Key for Importance on Web Data: Link between Documents

$f(\cdot, d)$



Data Labeling Methods

- Rank
  - e.g., relevant, partially relevant, irrelevant
- Ordered Pair
  - e.g., A > B, B > C
- Others are impractical
  - List
  - Score

Evaluation Measures

- NDCG (Normalized Discounted Cumulative Gain)
- MAP (Mean Average Precision)
- MRR (Mean Reciprocal Rank)
- WTA (Winners Take All)

NDCG

- Example: perfect ranking for q_i
  - (3, 3, 2, 2, 1, 1, 1) rank *r = 3, 2, 1*
  - (7, 7, 3, 3, 1, 1, 1) gain $2^{r(j)} - 1$
  - (1, 0.63, 0.5, 0.43, 0.39, 0.36, 0.33) position discount $1/\log(1+j)$
  - (7, 18.11, 24.11, …) DCG $$\sum_{j=1}^{m} (2^{r(j)} - 1)/\log(1+j)$$
  - (1/7, 1/18.11, 1/24.11, …) normalizing factor n_i
  - (1, 1, 1,1,1,1,1) NDCG for perfect ranking

NDCG (cont')

- Example: imperfect ranking
  - (2, 3, 2, 3, 1, 1, 1)
  - (3, 7, 3, 7, 1, 1, 1) Gain
  - (1, 0.63, 0.5, 0.43, 0.39, 0.36, 0.33) Position discount
  - (3, 14.11, 20.11, …) DCG
  - (0.43, 0.78, 0.83, ….) NDCG

- An imperfect ranking for the query will always decrease NDCG

Characteristics of Learning to Rank

- No need to predict category (or ordered category)

  vs Classification

- No need to predict value of $f(q,d)$

  vs Regression

- Ranking order is more important

  vs Ordinal regression

Ordinal Regression (Ordinal Classification)

- Categories are ordered
  - 5, 4, 3, 2, 1
  - e.g., rating restaurants
- Prediction
  - Map to ordered categories

Learning to Rank for Information Retrieval

- Important to ranking top results correctly
- Numbers of documents vary according to queries

Existing Approaches

- Pairwise Approach
  - Ranking SVM (Herbrich et al 2000)
  - RankBoost (Freund et al 2003)
  - Ranknet (Burges et al 2005)
- Listwise Approach
  - ListNet (Cao et al 2007)
  - AdaRank (Xu & Li 2007)
  - SVM Method for Maximizing MAP (Yue et al 2007)

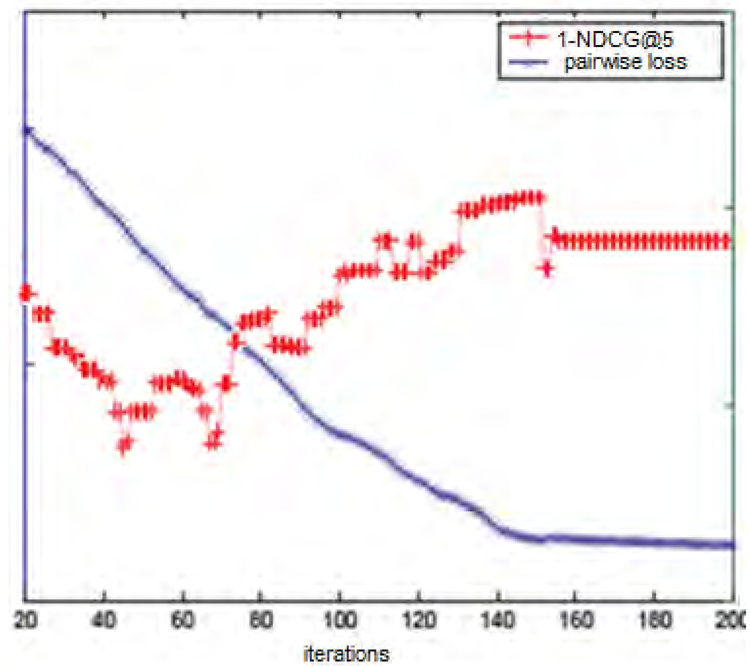
Previous Work: Pairwise Approach

- Transforming ranked list into document pairs
- Formalizing ranking as classification on document pairs
- Ranking SVM, RankNet, RankBoost

$$
\begin{array}{c} q^{(i)} \\ \begin{pmatrix} d_1^{(i)}, 5 \\ d_2^{(i)}, 3 \\ \vdots \\ d_{n^{(i)}}^{(i)}, 2 \end{pmatrix} \end{array}
\xrightarrow{\text{Transform}}
\begin{array}{c} q^{(i)} \\ \left\{ \left(d_1^{(i)}, d_2^{(i)}\right), \left(d_1^{(i)}, d_{n^{(i)}}^{(i)}\right), \ldots, \left(d_2^{(i)}, d_{n^{(i)}}^{(i)}\right) \right\} \\ 5 > 3 \qquad 5 > 2 \qquad 3 > 2 \end{array}
$$

Problems with Pairwise Approach

- Loss function is suboptimal
  - Not to optimize evaluation measures (e.g. NDCG and MAP)
  - Not to consider position in ranking and number of documents per query



Pairwise loss vs. (1-NDCG@5)
TREC Dataset

Our Proposal: Listwise Approach

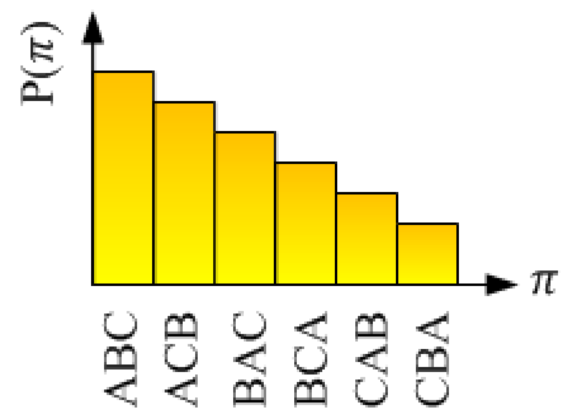
ListNet: Probabilistic Model for Ranking

Distance between Ranked Lists

- A Simple Example:
  - function *f*: $f(\mathrm{A})=3,\ f(\mathrm{B})=1,\ f(\mathrm{C})=0$ ABC
  - function *h*: $h(\mathrm{A})=5, h(\mathrm{B})=2, h(\mathrm{C})=3$ ACB
  - ground truth *g*: $g(\mathrm{A})=5, g(\mathrm{B})=3, g(\mathrm{C})=2$ ABC
- Question: which function is closer to ground truth?
- Our proposal:
  - Ranked list ↔ Permutation probability distribution

f: $f(\mathrm{A})=3, f(\mathrm{B})=1, f(\mathrm{C})=0$;
Ranking by *f*: ABC

↔



Permutation Probability

- Probability of permutation π is defined as

$$P_s(\pi) = \prod_{j=1}^{n} \frac{\varphi(s_{\pi(j)})}{\sum_{k=j}^{n} \varphi(s_{\pi(k)})}$$

- Example:

$$P_f(\text{ABC}) = \boxed{\frac{\varphi(f(\text{A}))}{\varphi(f(\text{A})) + \varphi(f(\text{B})) + \varphi(f(\text{A}))}} \cdot \boxed{\frac{\varphi(f(\text{B}))}{\varphi(f(\text{B})) + \varphi(f(\text{C}))}} \cdot \boxed{\frac{\varphi(f(\text{C}))}{\varphi(f(\text{C}))}}$$

P(A ranked No.1)

P(B ranked No.2 | A ranked No.1)
= P(B ranked No.1)/(1- P(A ranked No.1))

P(C ranked No.3 | A ranked No.1, B ranked No.2)

Properties of Permutation Probability

- Function f: $f(A)=3$, $f(B)=1$, $f(C)=0$ ABC
- Property 1: $P(ABC)$ is largest, $P(CBA)$ is smallest
- Property 2: swap B and C in ABC, $P(ABC) > P(ACB)$

Top-k Probability

- Computation of Permutation Probability is intractable
- Top-k Probability
  - Defining Top-k subgroup $G(j_1, \ldots, j_k)$ containing all permutations whose top-k documents are $j_1, \ldots, j_k$
  - $$P_s(G(j_1, \ldots, j_k)) = \prod_{t=1}^{k} \frac{\varphi(s_{\pi(j_t)})}{\sum_{u=t}^{n} \varphi(s_{\pi(j_u)})}$$
  - Time complexity of computation : from $n!$ to $n!/(n-k)!$

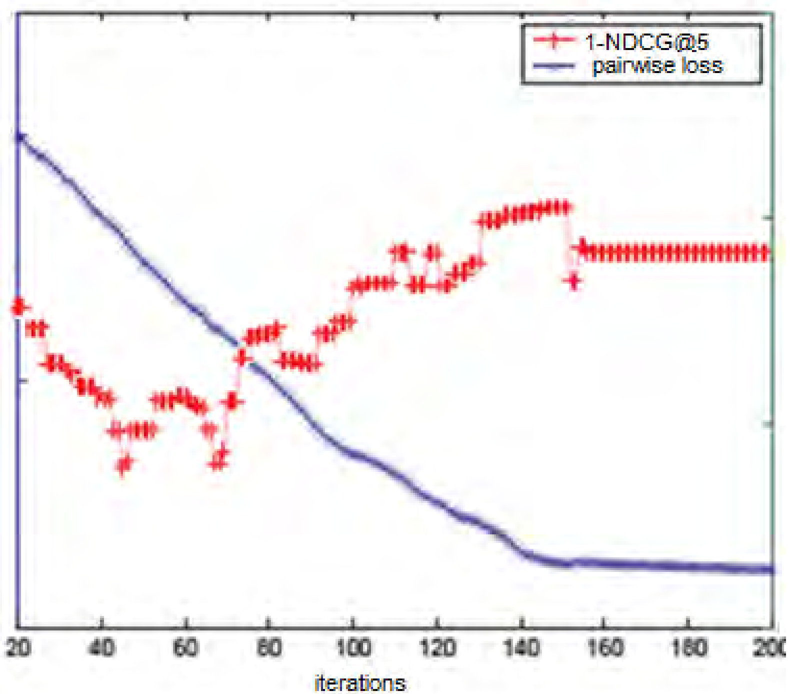
ListNet Method (*ICML'07*)

- Loss function = KL-divergence between two Top-k probability distributions ($\varphi = \exp$)

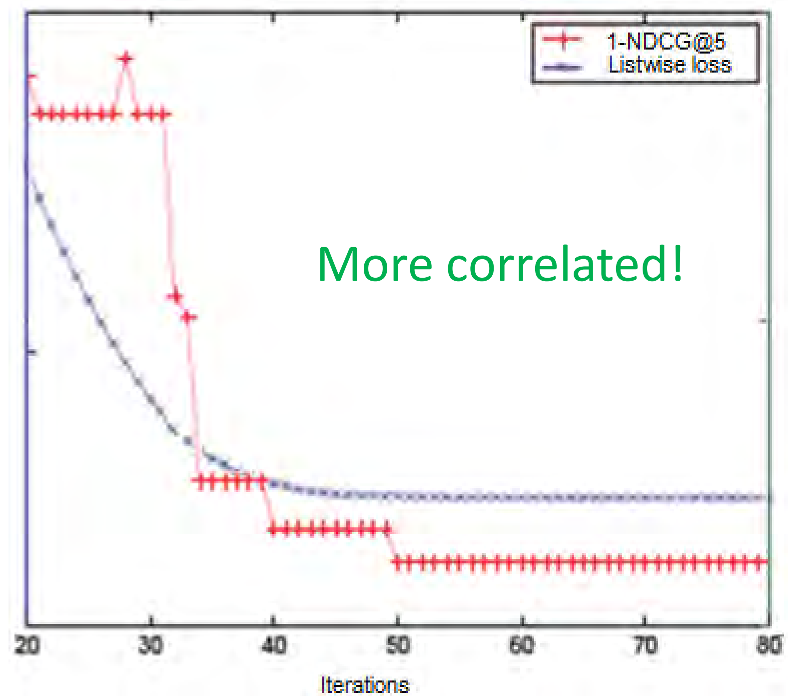
$$L(w) \propto -\sum_{q \in Q} \sum_{G(j_1,\ldots,j_k)} \left(\prod_{t=1}^{k} \frac{\exp(s_{y(j_t)})}{\sum_{u=t}^{n} \exp(s_{y(j_u)})} \right) \log \left(\prod_{t=1}^{k} \frac{\exp(w \cdot X_{y(j_t)})}{\sum_{u=t}^{n} \exp(w \cdot X_{y(j_u)})} \right)$$

- Model = Neural Network
- Algorithm = Gradient Descent

Experimental Results



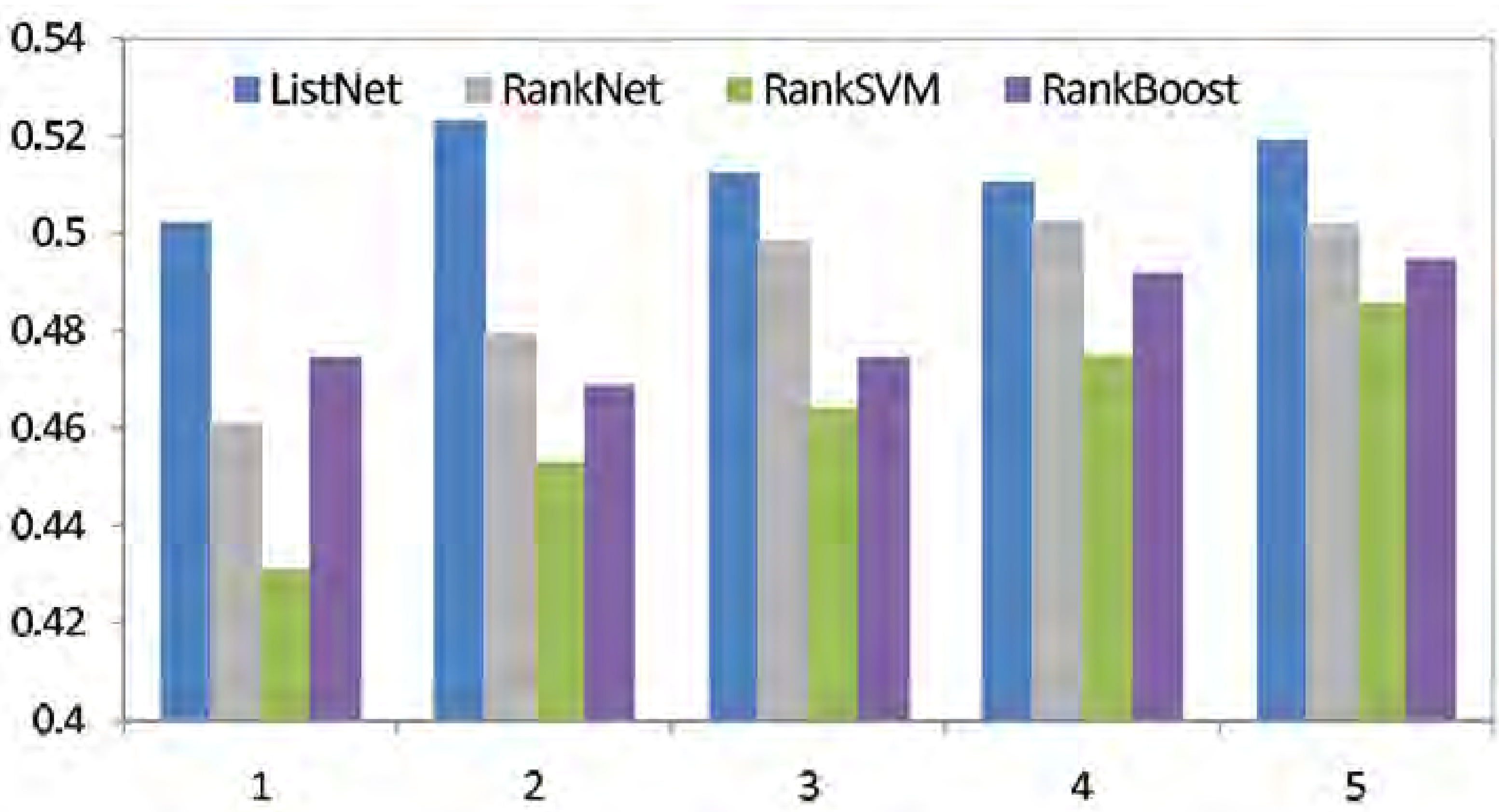
Pairwise (RankNet)



Listwise (ListNet)

Training Performance on TREC Dataset

Experimental Results

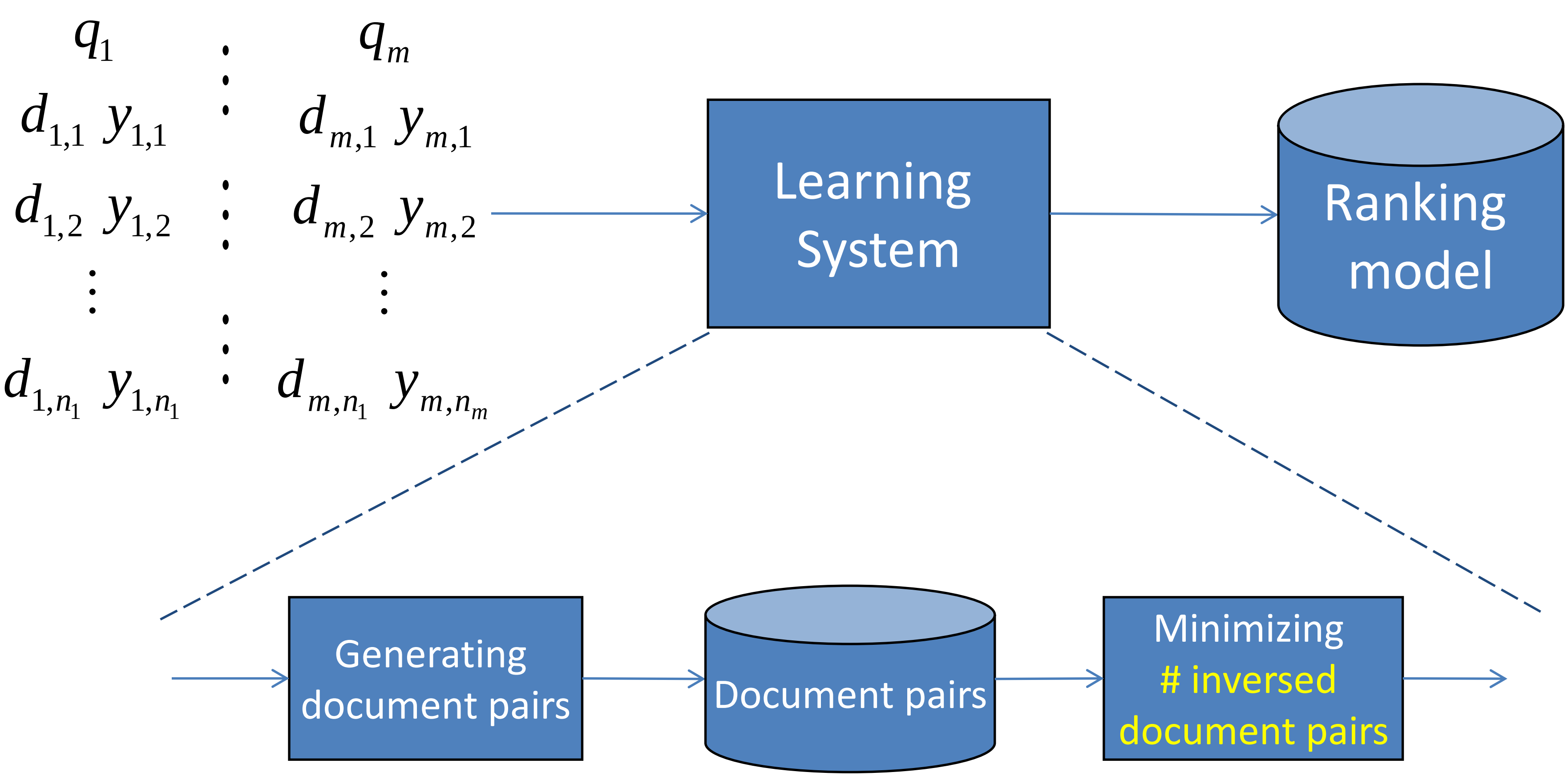


NDCG@

Testing Performance on TREC Dataset

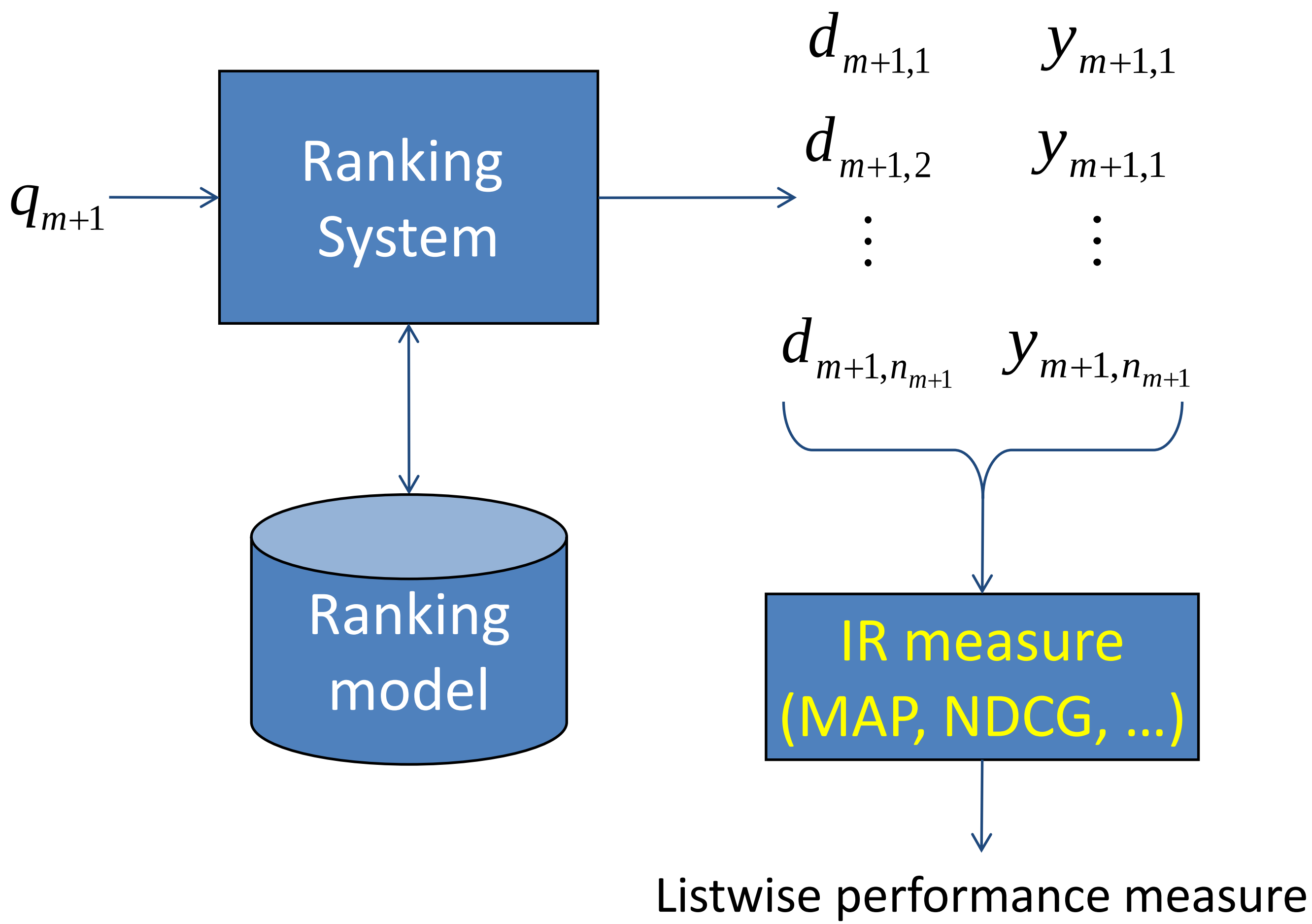
AdaRank: Boosting Algorithm for Ranking

Traditional Pairwise Approach (Training)



Minimizing loss function based on document pairs

Traditional Pairwise Approach (Evaluation)



Loss Function of AdaRank

$$\max_{f \in \mathcal{F}} \sum_{i=1}^{m} \boxed{E(\pi(q_i, \mathbf{d}_i, f), \mathbf{y}_i)}$$

Any evaluation measure taking value between [-1,+1]

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{m} (1 - E(\pi(q_i, \mathbf{d}_i, f), \mathbf{y}_i))$$

$$e^{-x} \geq 1 - x$$

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{m} \exp\{-E(\pi(q_i, \mathbf{d}_i, f), \mathbf{y}_i)\}$$

$$f(\vec{x}) = \sum_{t=1}^{T} \alpha_t h_t(\vec{x})$$

$$\min_{h_t \in \mathcal{H}, \alpha_t \in \Re^+} L(h_t, \alpha_t) = \sum_{i=1}^{m} \exp\{-E(\pi(q_i, \mathbf{d}_i, f_{t-1} + \alpha_t h_t), \mathbf{y}_i)\}$$

AdaRank Algorithm

Input: $S = \{(q_i, \mathbf{d}_i, \mathbf{y}_i)\}_{i=1}^{m}$, and parameters E and T
Initialize $P_1(i) = 1/m$.
For $t = 1, \cdots, T$

- Create weak ranker h_t with weighted distribution $\boldsymbol{P}_t$ on training data S.
- Choose α_t

$$\alpha_t = \frac{1}{2} \cdot \ln \frac{\sum_{i=1}^{m} P_t(i)\{1 + E(\pi(q_i, \mathbf{d}_i, h_t), \mathbf{y}_i)\}}{\sum_{i=1}^{m} P_t(i)\{1 - E(\pi(q_i, \mathbf{d}_i, h_t), \mathbf{y}_i)\}}.$$

- Create f_t

$$f_t(\vec{x}) = \sum_{k=1}^{t} \alpha_k h_k(\vec{x}).$$

- Update $\boldsymbol{P}_{t+1}$

$$P_{t+1}(i) = \frac{\exp\{-E(\pi(q_i, \mathbf{d}_i, f_t), \mathbf{y}_i)\}}{\sum_{j=1}^{m} \exp\{-E(\pi(q_j, \mathbf{d}_j, f_t), \mathbf{y}_j)\}}.$$

End For
Output ranking model: $f(\vec{x}) = f_T(\vec{x})$.

Theoretical Results on AdaRank

- Training error will be continuously reduced during learning phase.

THEOREM 1. *The following bound holds on the ranking accuracy of the AdaRank algorithm on training data:*

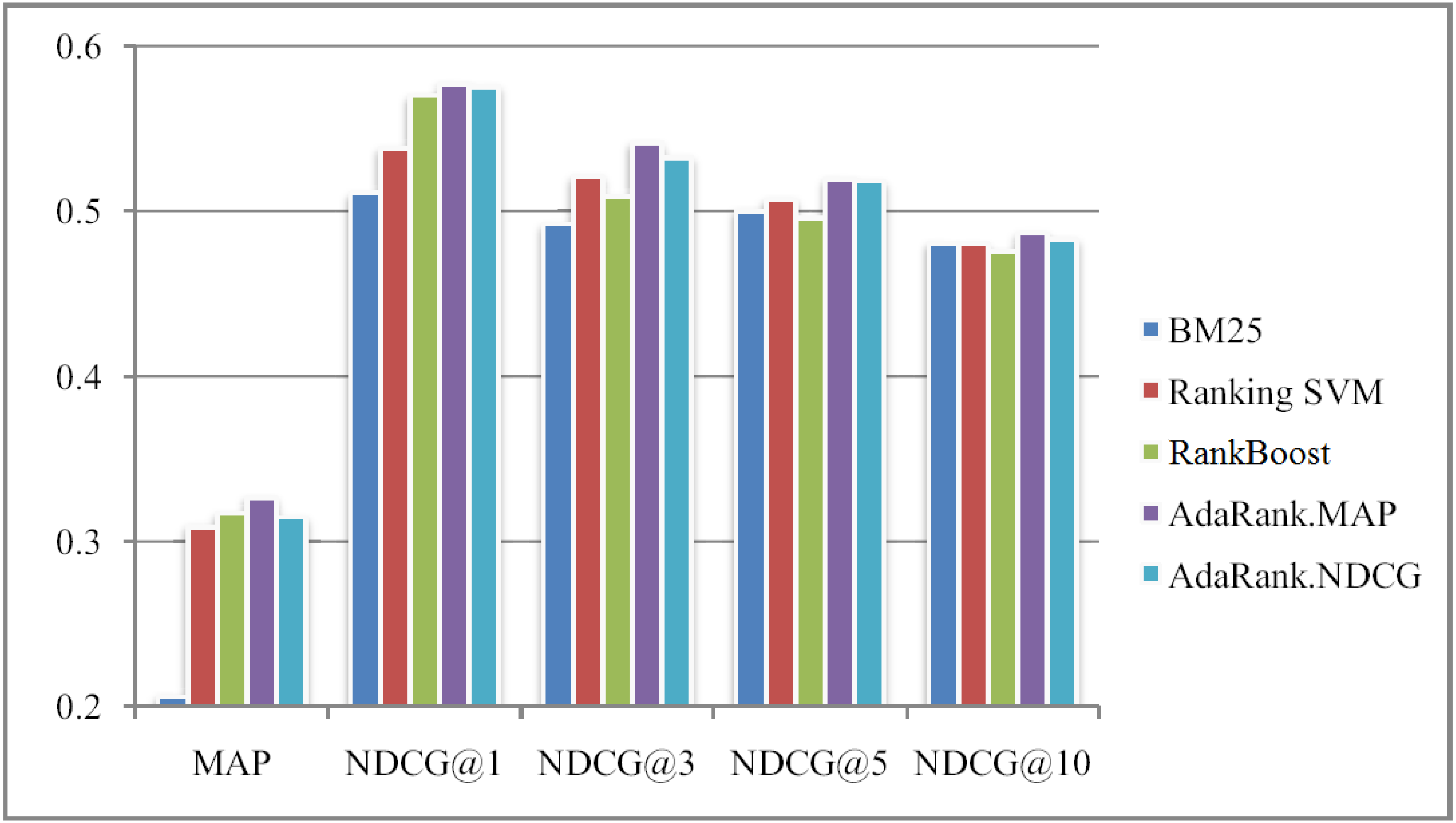
$$\frac{1}{m}\sum_{i=1}^{m} E(\pi(q_i, \mathbf{d}_i, f_T), \mathbf{y}_i) \geq 1 - \prod_{t=1}^{T} e^{-\delta^t_{min}} \sqrt{1 - \varphi(t)^2},$$

where $\varphi(t) = \sum_{i=1}^{m} P_t(i) E(\pi(q_i, \mathbf{d}_i, h_t), \mathbf{y}_i)$, $\delta^t_{min} = \min_{i=1,\cdots,m} \delta^t_i$, *and*

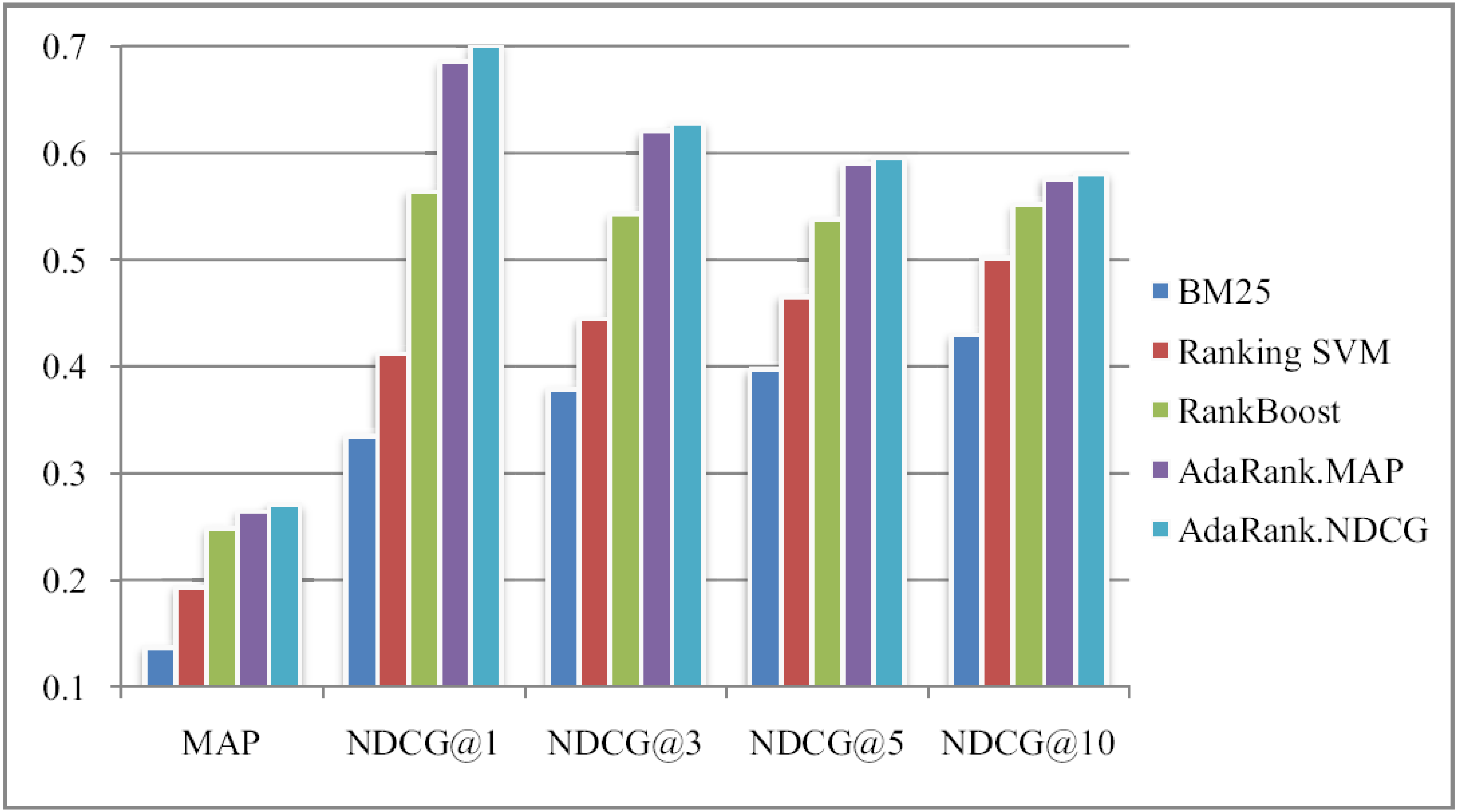
$$\delta^t_i = E(\pi(q_i, \mathbf{d}_i, f_{t-1} + \alpha_t h_t), \mathbf{y}_i) - E(\pi(q_i, \mathbf{d}_i, f_{t-1}), \mathbf{y}_i) \\ -\alpha_t E(\pi(q_i, \mathbf{d}_i, h_t), \mathbf{y}_i),$$

for all $i = 1, 2, \cdots, m$ *and* $t = 1, 2, \cdots, T$.

Ranking Accuracies on OHSUMED Data



Ranking Accuracies on .Gov Data



Summary

Summary

- Ranking in Information Retrieval
- Learning to Rank
- ListNet: Probabilistic Model for Ranking
- AdaRank: Boosting Algorithm for Ranking

References

- Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, Hang Li, Learning to Rank: From Pairwise Approach to Listwise Approach, Proc. of ICML 2007, 129-136. (pdf)
- Jun Xu, Hang Li, AdaRank: A Boosting Algorithm for Information Retrieval, Proc. of SIGIR 2007, 391-398. (pdf)

Thanks!