ACML 2009 Tutorial
Nov. 2, 2009
Nanjing

Learning to Rank

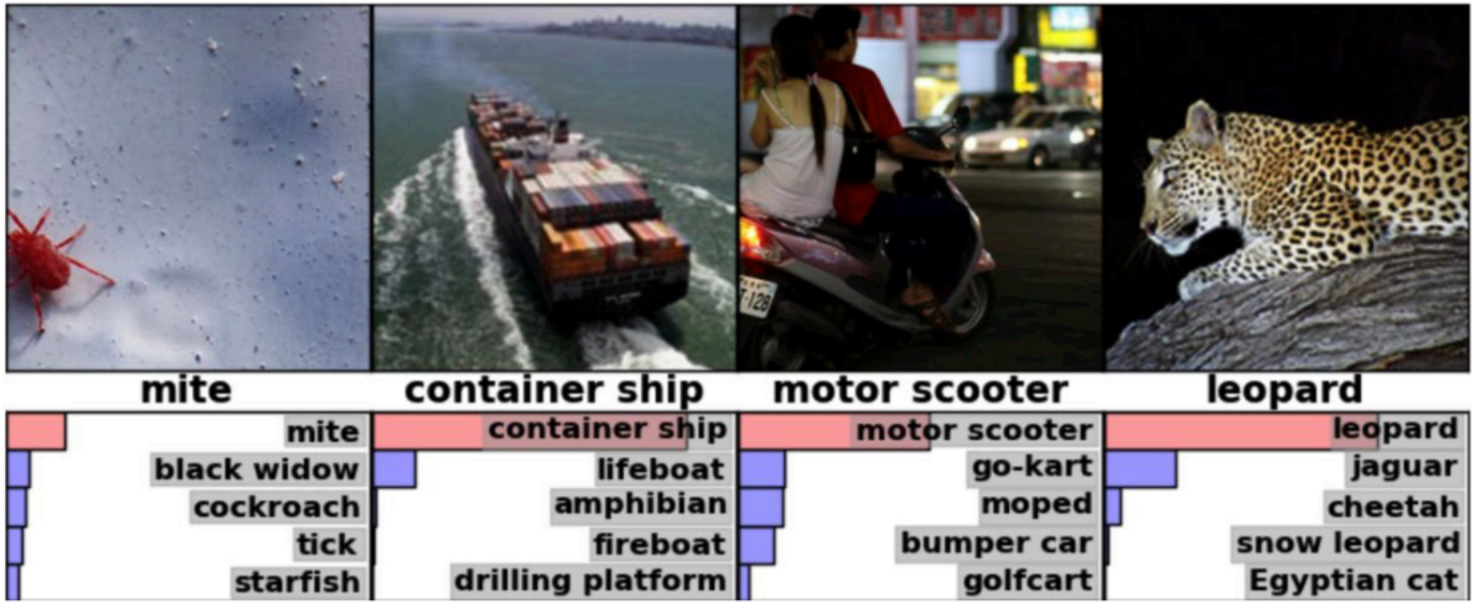
Hang Li
Microsoft Research Asia

Outline of Tutorial

1. Introduction
2. Learning to Rank Problem
3. Learning to Rank Methods
4. Learning to Rank Theory
5. Learning to Rank Applications
6. Future Directions of Learning to Rank Research
7. Summary

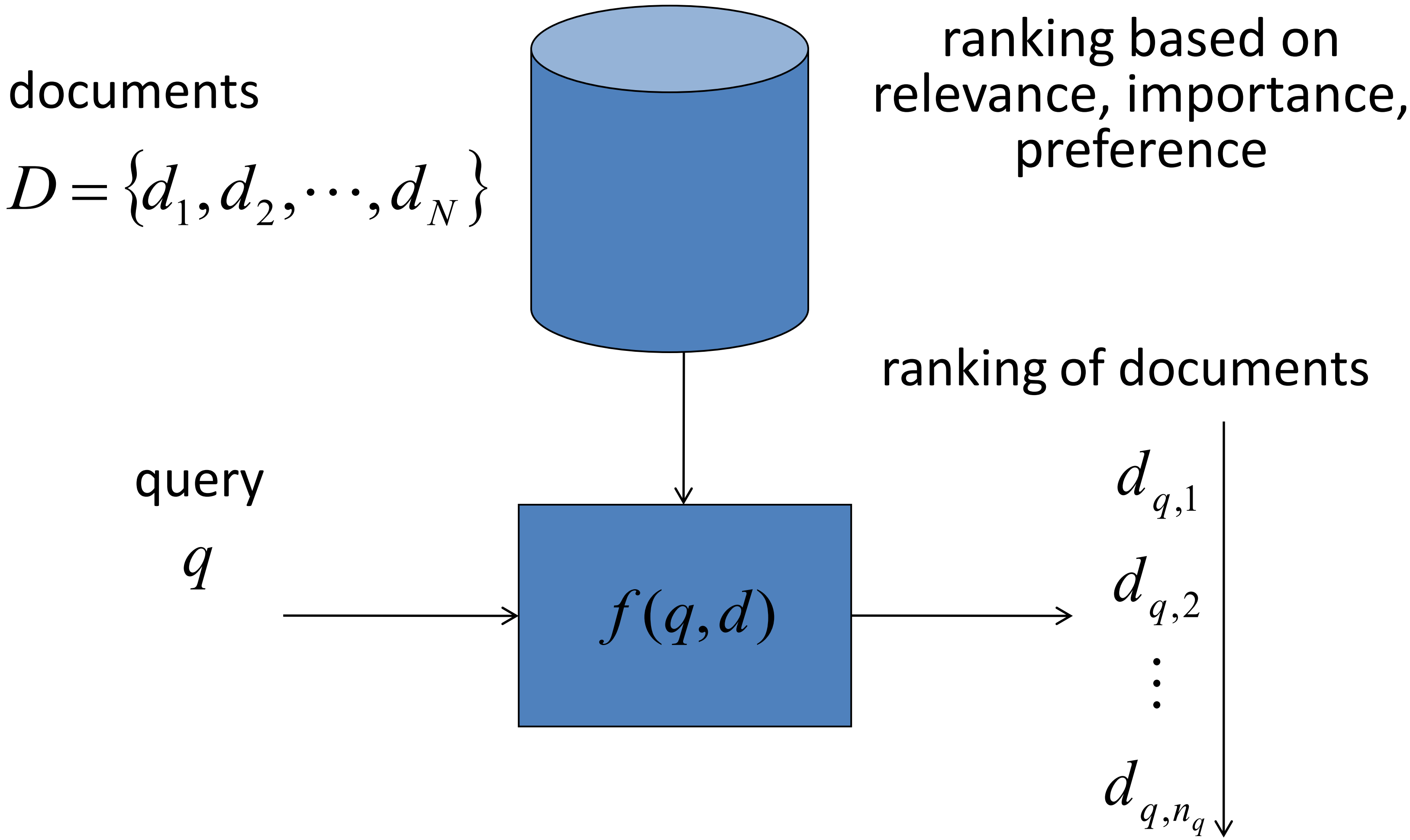
Deep Learning Success: Vision

Image Recognition IMAGENET



1. Introduction

Ranking Problem: Example = Document Search

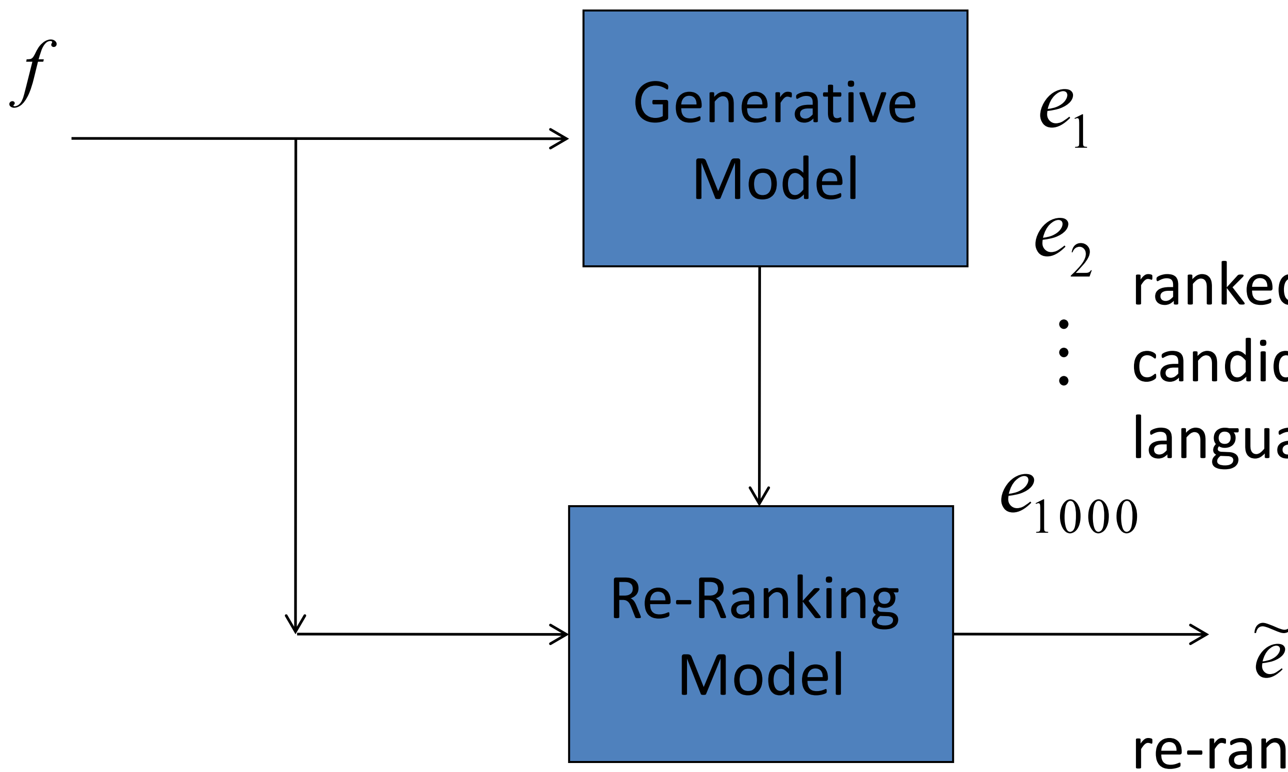


Ranking Problem
Example = Recommender System

	Item1	Item2	Item3	...	
User1	5	4			
User2	1		2		2
...		?	?	?	
UserM	4	3			

Ranking Problem
Example = Machine Translation

sentence source language



ranked sentence candidates in target language

re-ranked top sentence in target language

Ranking Problem

subjects

ranking of objects

$$S = \{s_1, s_2, \cdots, s_i, \cdots\}$$

s_i

$$F(s, O)$$

$o_{i,1}$

$o_{i,2}$

$\vdots$

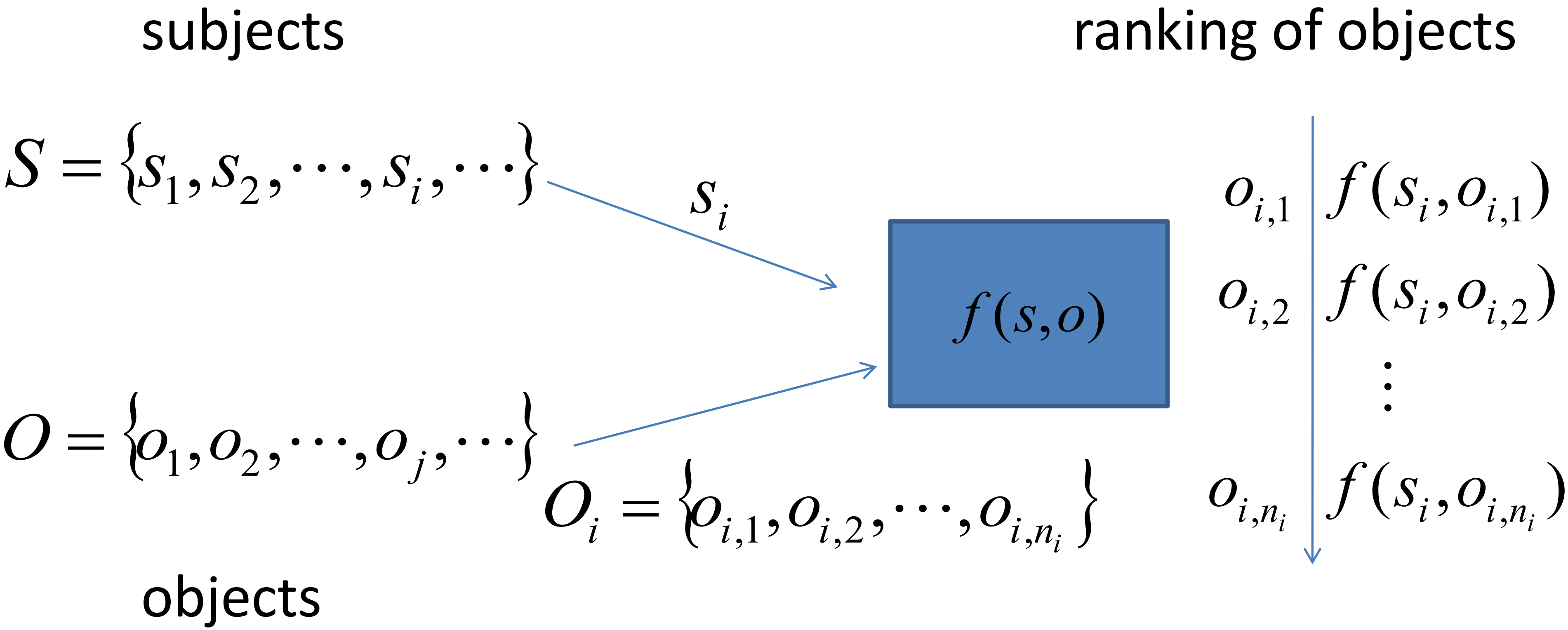
o_{i,n_i}

$$O = \{o_1, o_2, \cdots, o_j, \cdots\}$$

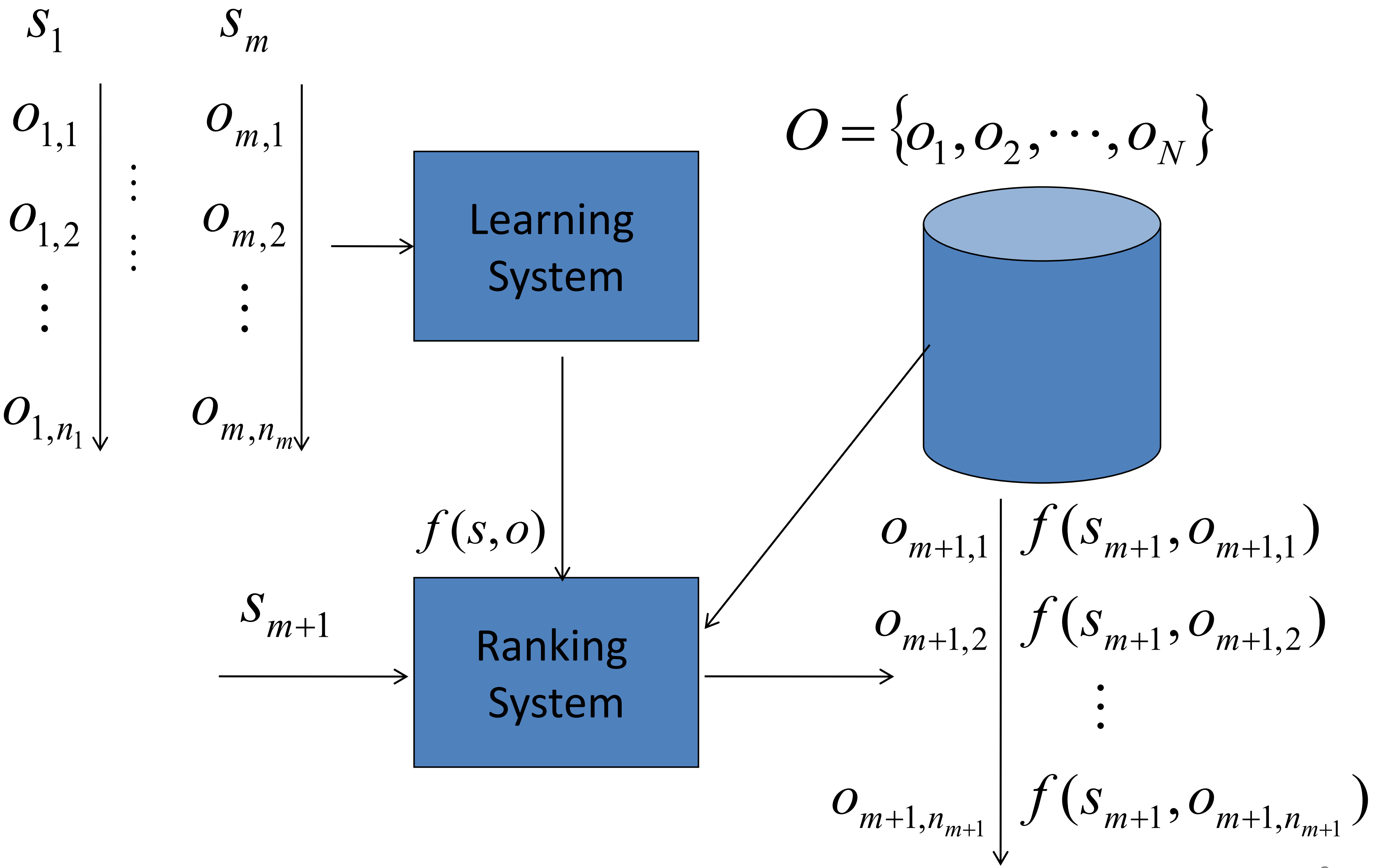
$$O_i = \{o_{i,1}, o_{i,2}, \cdots, o_{i,n_i}\}$$

objects

By Ranking Function



Learning to Rank



What is Learning to Rank?

- Definition 1 (in broad sense)

  Learning to Rank = any machine learning technology for ranking problem
- Definition 2 (in narrow sense)

  Learning to Rank = machine learning technology for the above ranking problem (ranking of objects given subject)
- This tutorial takes Definition 2

Ranking Plays Key Role in Many Applications

Applications of Learning to Rank

- Search (Document Search, Entity Search, etc)
- Recommender System (Collaborative Filtering)
- Key Phrase Extraction
- Question Answering
- Document Summarization
- Opinion Mining
- Sentiment Analysis
- Machine Translation

Technologies on Learning to Rank

- Methods
  - Pointwise Methods
  - Pairwise Methods
  - Listwise Methods
- Theory
  - Generalization
  - Consistency
- Applications
  - Search
  - Collaborative Filtering
  - Key Phrase Extraction

Recent Trends on Learning to Rank

- Successfully applied to search
- Over 100 publications at SIGIR, ICML, NIPS, etc
- One book on learning to rank for information retrieval
- 2 sessions at SIGIR every year
- 3 SIGIR workshops
- Special issue at Information Retrieval Journal
- LETOR benchmark dataset, over 400 downloads

  http://research.microsoft.com/en-us/um/beijing/projects/letor/index.html

Scope of This Tutorial

- Overview of Learning to Rank technologies
- Focusing on Learning to Rank methods
- Touching theoretical issues
- Showing future directions
- Knowledge necessary for this tutorial: Machine Learning

Other ‘Learning to Rank’ Methods *Not Covered* in This Tutorial

- Rank Aggregation
- Ranking of Objects on Graph
  - Link Analysis (e.g, Page Rank)
- Unsupervised Learning of Ranking
  - Probabilistic Model in Information Retrieval (e.g., BM25, Language Model for IR)

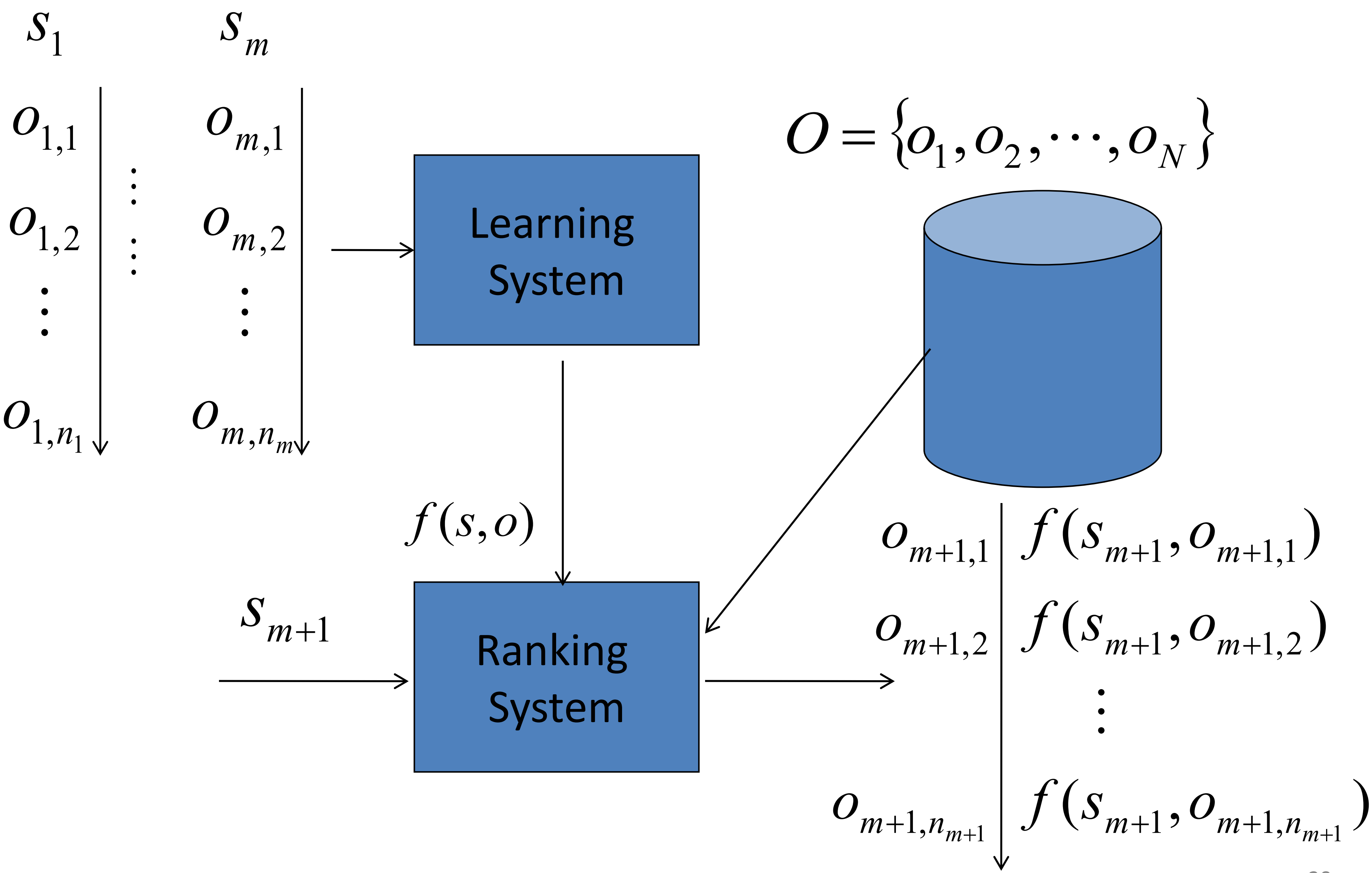
2. Learning to Rank Problem

Learning to Rank Problem

- 2.1 Problem Formulation
- 2.2 Example: Learning to Rank for Search
- 2.3 Issues in Learning to Rank
- 2.4 Relations with Other Learning Tasks

2.1 Problem Formulation

Learning to Rank

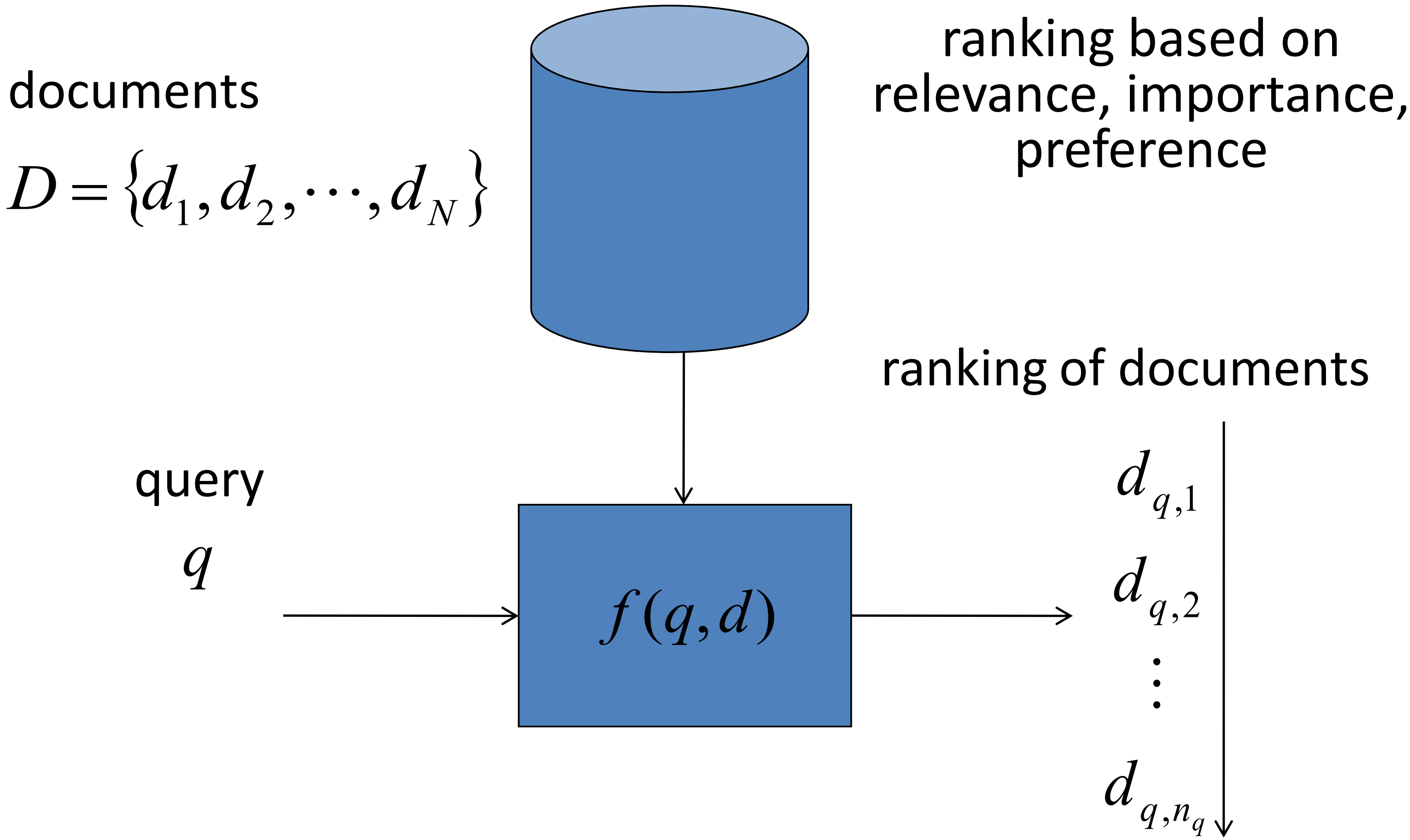


Characteristics of Learning to Rank

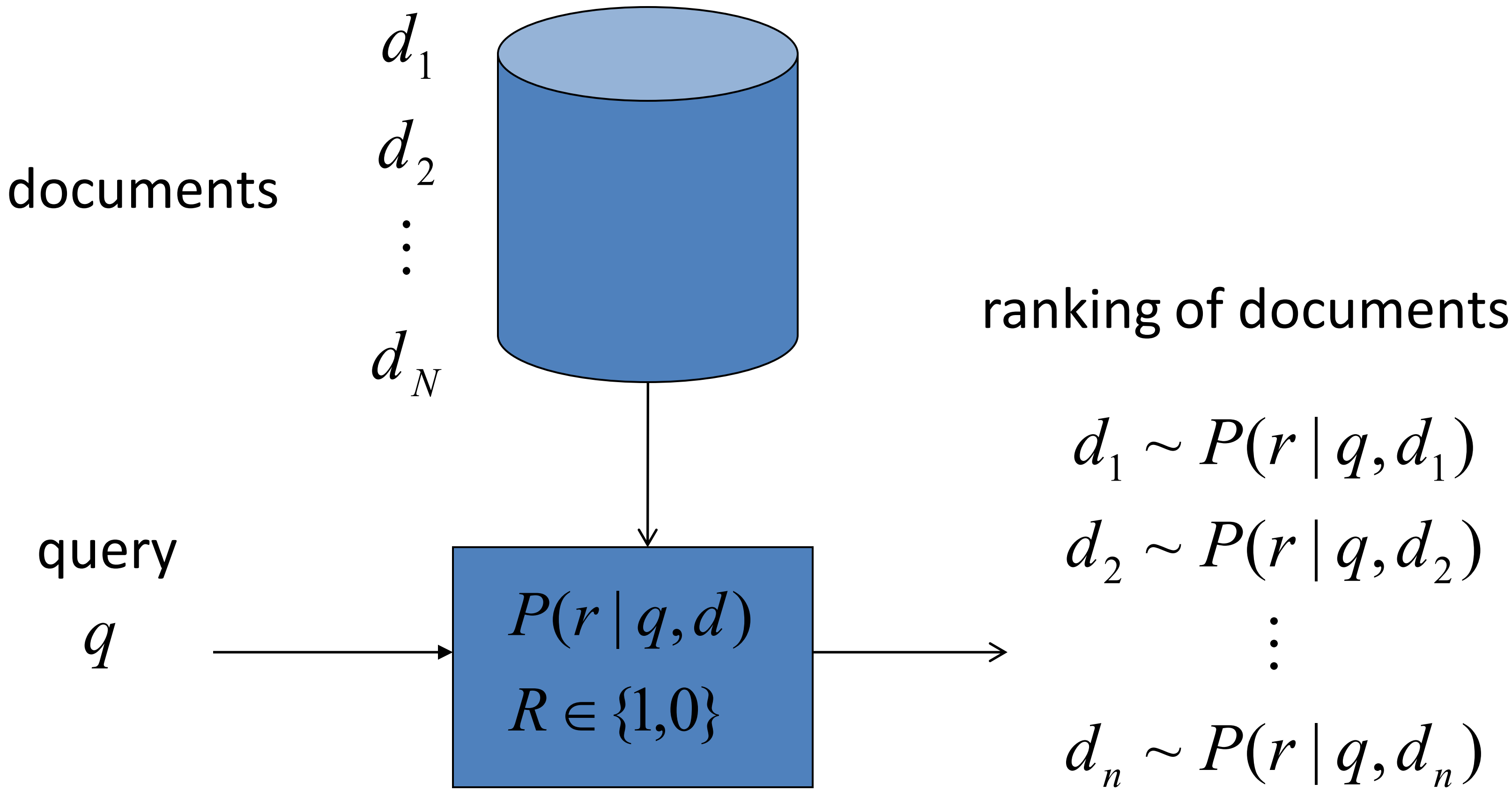
- Machine learning technologies
- Ranking of objects by subject
- Learning ranking function $f(s,o)$
- Using labeled data (supervised learning)
- Ranking function is feature based
- This tutorial mainly takes *document search* as example

2.2 Example: Learning to Rank for Search

Ranking Problem: Example = Document Search



Traditional Approach = Probabilistic Model



BM25

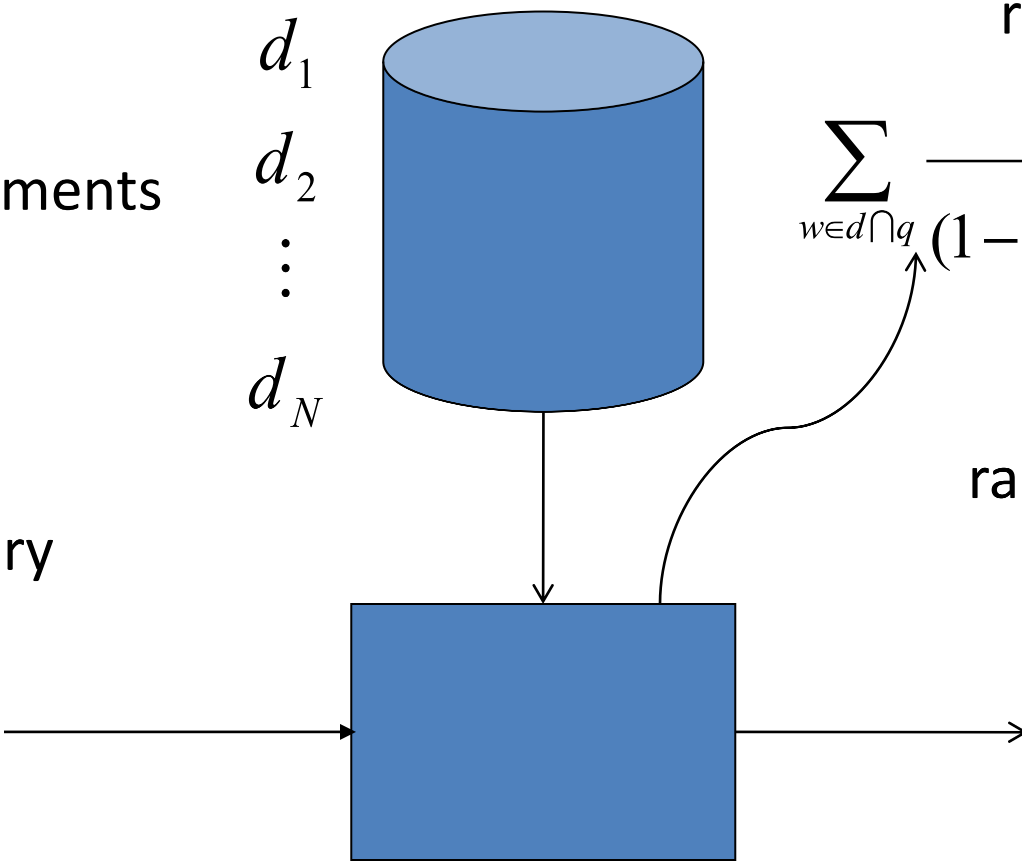
documents

d_1

d_2

$\vdots$

d_N



query

q

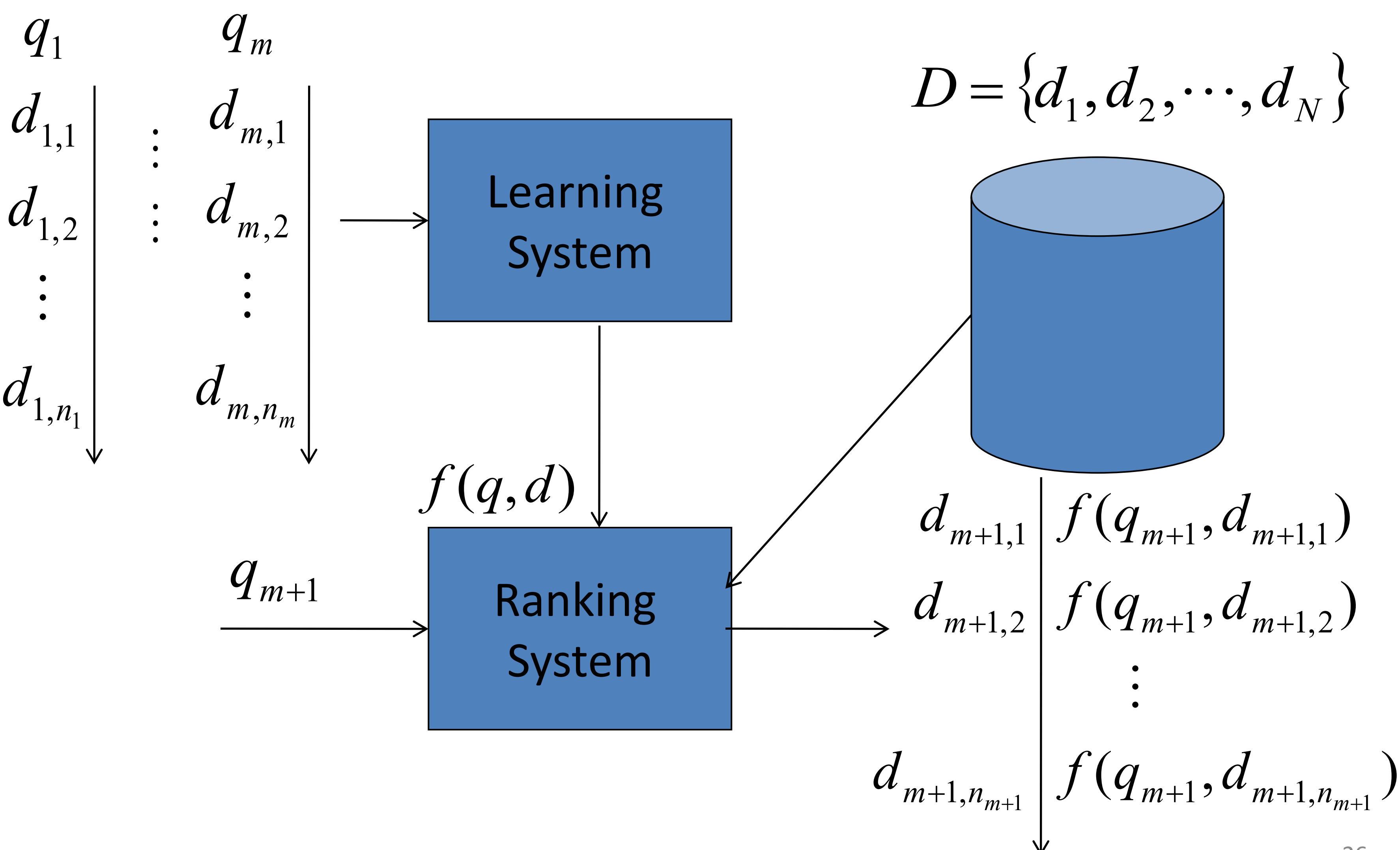
ranking function

$$\sum_{w \in d \cap q} \frac{(k+1)tf(w)}{(1-b)k + b\dfrac{dl}{avgdl} + tf(w)}$$

ranking of documents

$$d_1 \sim P(r \mid q, d_1)$$
$$d_2 \sim P(r \mid q, d_2)$$
$$\vdots$$
$$d_n \sim P(r \mid q, d_n)$$

New Approach = Learning to Rank



Training Process

$$
\begin{array}{ccccccc}
q_1 \begin{cases} d_{1,1} \\ d_{1,2} \\ \vdots \\ d_{1,n_1} \end{cases} & & q_1 \begin{cases} d_{1,1} & y_{1,1} \\ d_{1,2} & y_{1,2} \\ \vdots & \\ d_{1,n_1} & y_{1,n_1} \end{cases} & & \begin{cases} x_{1,1} & y_{1,1} \\ x_{1,2} & y_{1,2} \\ \vdots & \\ x_{1,n_1} & y_{1,n_1} \end{cases} & & \\
\vdots & & \vdots & & \vdots & & \\
& \xrightarrow{\text{1. Data Labeling (rank)}} & & \xrightarrow{\text{2. Feature Extraction}} & & \xrightarrow{\text{3. Learning}} & f(x) \\
\vdots & & \vdots & & \vdots & & \\
q_m \begin{cases} d_{m,1} \\ d_{m,2} \\ \vdots \\ d_{m,n_m} \end{cases} & & q_m \begin{cases} d_{m,1} & y_{m,1} \\ d_{m,2} & y_{m,2} \\ \vdots & \\ d_{m,n_m} & y_{m,n_m} \end{cases} & & \begin{cases} x_{m,1} & y_{m,1} \\ x_{m,2} & y_{m,2} \\ \vdots & \\ x_{m,n_m} & y_{m,n_m} \end{cases} & &
\end{array}
$$

Testing Process

$$q_{m+1}\begin{cases} d_{m+1,1} \\ d_{m+1,2} \\ \vdots \\ d_{m+1,n_{m+1}} \end{cases}$$

1. Data Labeling (rank)

$$q_{m+1}\begin{cases} d_{m+1,1} & y_{m+1,1} \\ d_{m+1,2} & y_{m+1,2} \\ \vdots & \\ d_{m+1,n_{m+1}} & y_{m+1,n_{m+1}} \end{cases}$$

2. Feature Extraction

$$\begin{cases} x_{m+1,1} & y_{m+1,1} \\ x_{m+1,2} & y_{m+1,2} \\ \vdots & \\ x_{m+1,n_{m+1}} & y_{m+1,n_{m+1}} \end{cases}$$

3. Ranking with $f(x)$

$$\begin{cases} x_{m+1,1} & f(x_{m+1,1}) & y_{m+1,1} \\ x_{m+1,2} & f(x_{m+1,2}) & y_{m+1,2} \\ \vdots & & \\ x_{m+1,n_{m+1}} & f(x_{m+1,n_{m+1}}) & y_{m+1,n_{m+1}} \end{cases}$$

4. Evaluation

Evaluation Result

Notes

- Features are functions of query and document
- Query and associated documents form a group
- Groups are i.i.d. data
- Feature vectors within group are not i.i.d. data
- Ranking model is function of features
- Several data labeling methods (here labeling of rank as example)

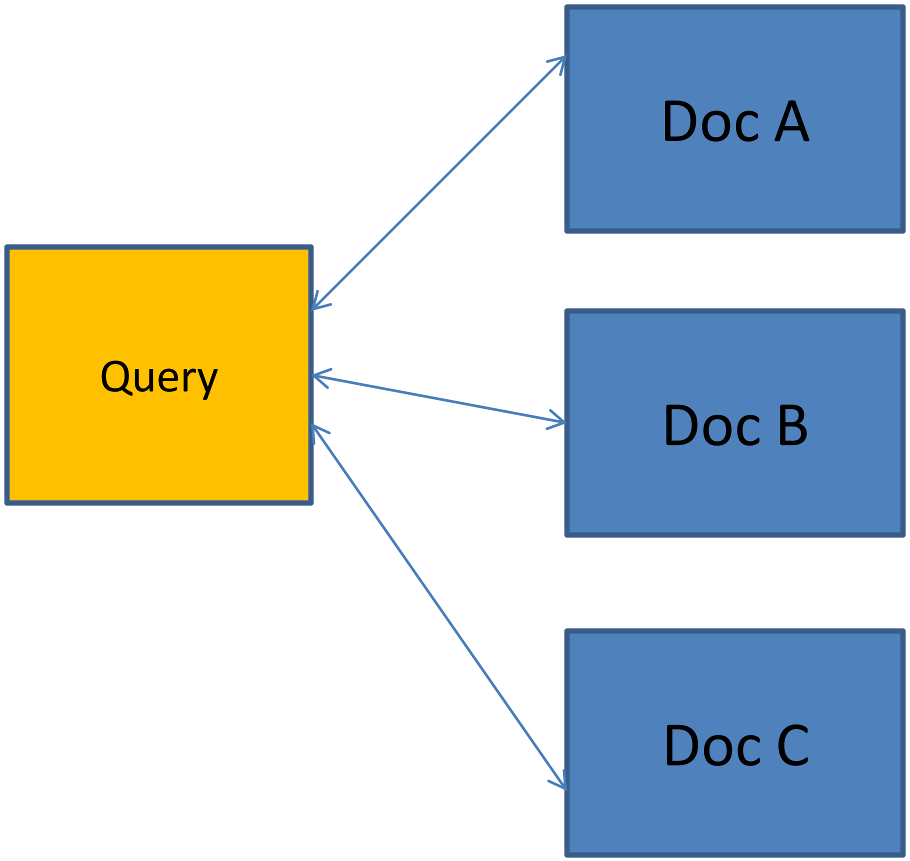
2.3 Issues in Learning to Rank

Issues in Learning to Rank

- Data Labeling
- Feature Extraction
- Evaluation Measure
- Learning Method (Model, Loss Function, Algorithm)

Data Labeling Problem

- E.g., relevance of documents w.r.t. query

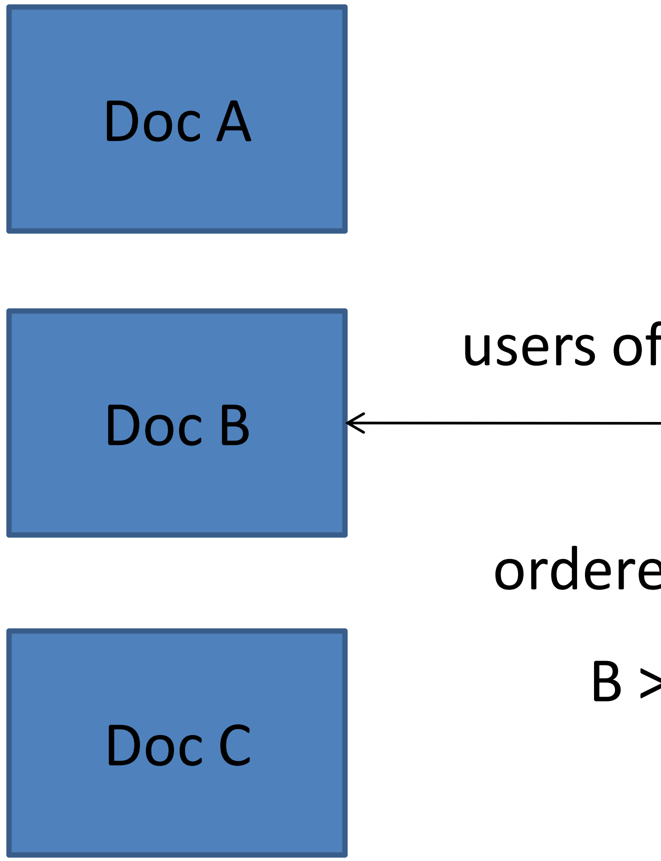


Data Labeling Methods

- Labeling of Ranks
  - Multiple levels (e.g., relevant, partially relevant, irrelevant)
  - Widely used in IR
- Labeling of Ordered Pairs
  - Ordered pairs between documents (e.g. A>B, B>C)
  - Implicit relevance judgment: derived from click-through data
- Creation of List
  - List (or permutation) of documents is given
  - Ideal but difficult to implement

Implicit Relevance Judgment

ranking of documents at search system

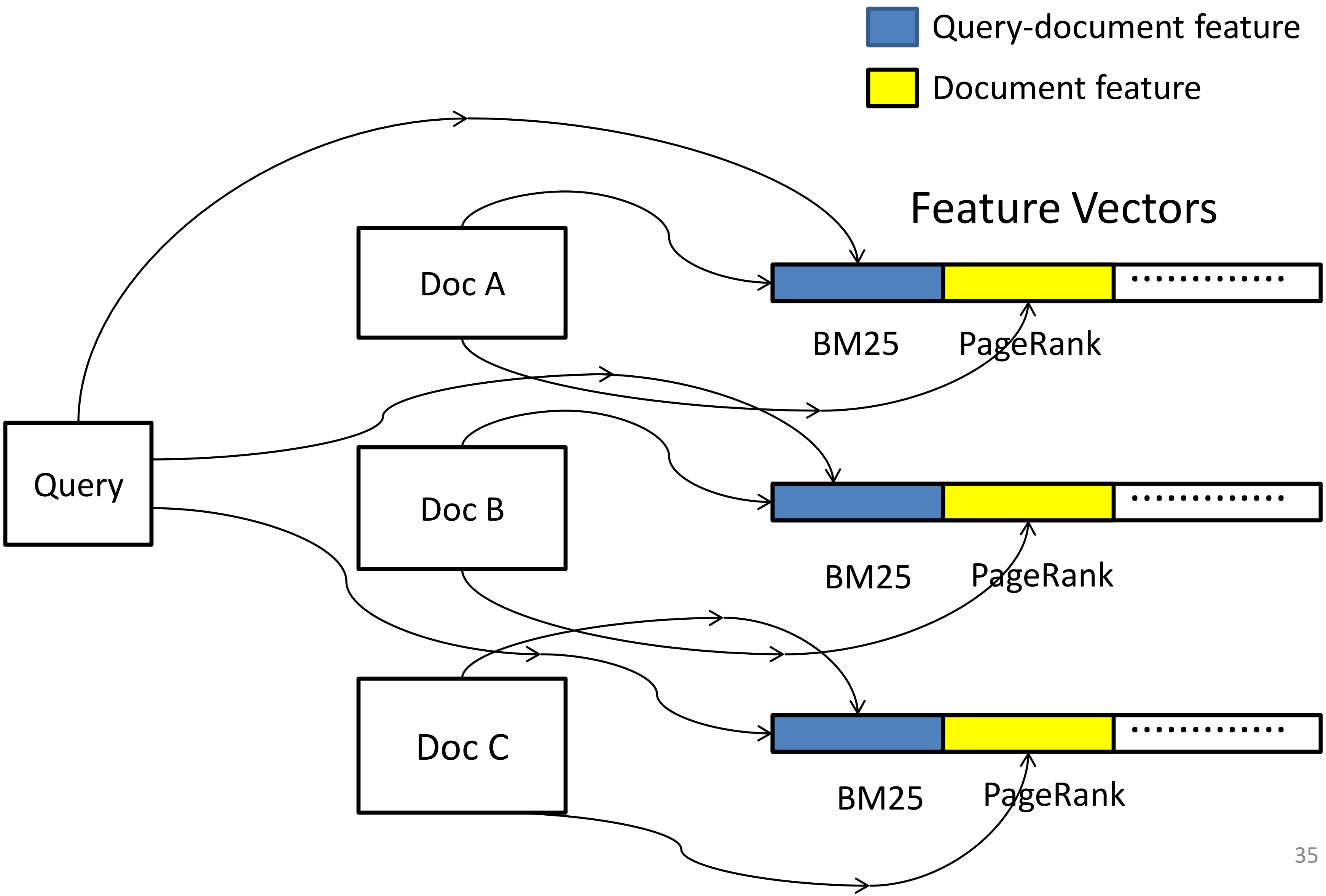


users often clicked on Doc B

ordered pair

B > A

Feature Extraction



Example Features

- Relevance: BM25
- Relevance: proximity
- Relevance: query exactly occurs in document
- Importance: PageRank

Evaluation Measures

- Important to rank top results correctly
- Measures
  - NDCG (Normalized Discounted Cumulative Gain)
  - MAP (Mean Average Precision)
  - MRR (Mean Reciprocal Rank)
  - WTA (Winners Take All)
  - Kendall's Tau

NDCG

- Evaluating ranking using labeled ranks
- NDCG at position *j*

$$\frac{1}{n_j}\sum_{i=1}^{j}(2^{r(i)}-1)/\log(1+i)$$

NDCG (cont')

- Example: perfect ranking
  - (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

$$\sum_{i=1}^{j}(2^{r(i)}-1)/\log(1+i)$$

  - (1/7, 1/18.11, 1/24.11, …) normalizing factor n_j
  - (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

- Imperfect ranking decreases NDCG

2.4 Relations with Other Learning Tasks

Relations with Other Learning Tasks

- No need to predict category
  vs Classification
- No need to predict value of $f(q,d)$
  vs Regression
- Relative ranking order is more important
  vs Ordinal regression
- *Learning to rank can be approximated by classification, regression, ordinal regression*

Ordinal Regression (Ordinal Classification)

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

3. Learning to Rank Methods

Learning to Rank Methods

- 3.1 Overview of Learning to Rank Methods
- 3.2 Pairwise Classification
- 3.3 Cost-sensitive Pairwise Classification
- 3.4 Probabilistic Model for Ranking
- 3.5 Direct Optimization of Evaluation Measure
- 3.6 Approximation of Evaluation Measure
- 3.7 Evaluation Results

3.1 Overview of Learning to Rank Methods

Three Major Approaches

- Pointwise approach
- Pairwise approach
- Listwise approach

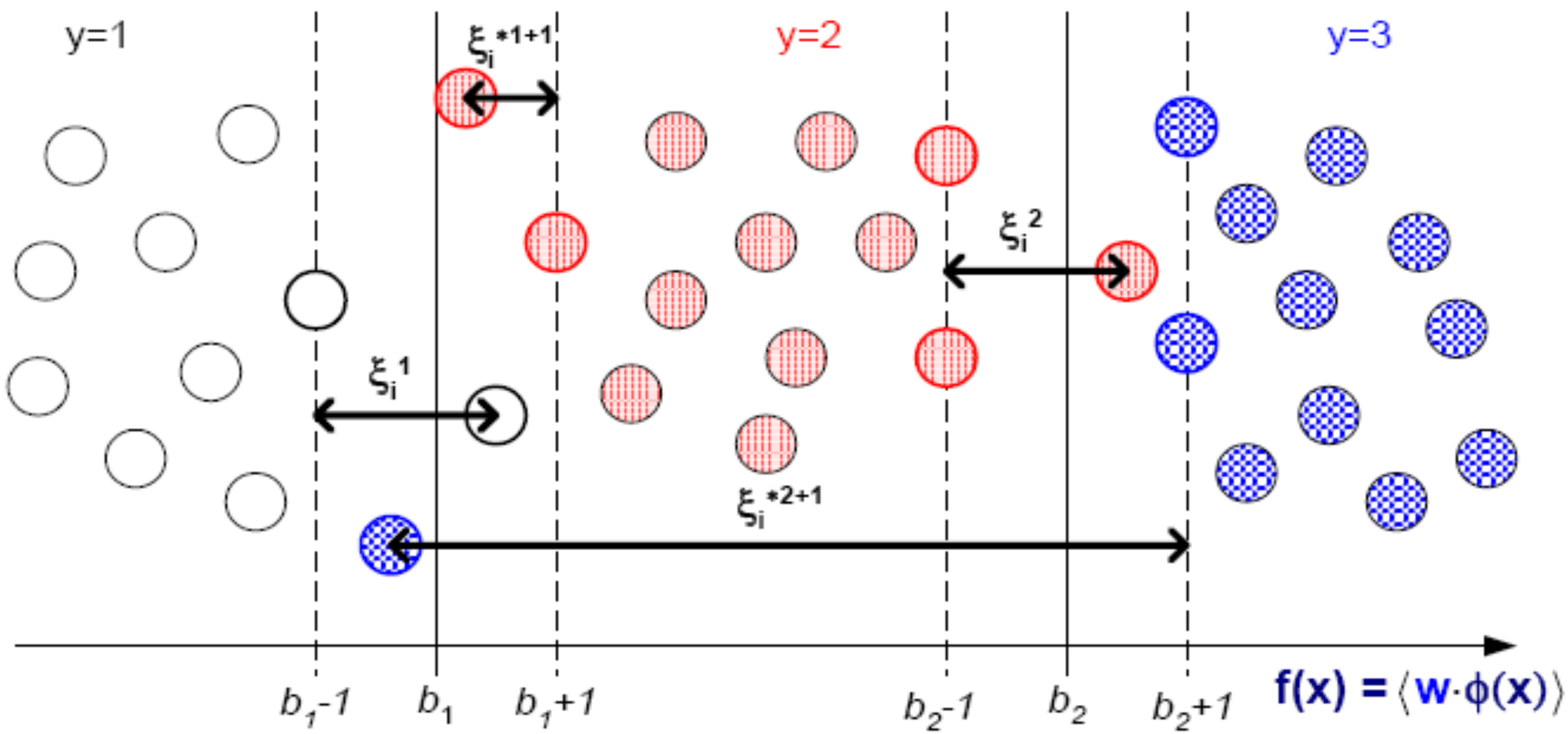
- First is suitable to ordinal regression
- Latter two are more suitable to learning to rank (e.g., ranking in search)

Pointwise Approach

- Transforming ranking to regression, classification, or ordinal regression
- Query-document group structure is ignored

Point-wise learning

- Goal is to learn a threshold to separate each rank



Pointwise Approach

	Learning		
	Regression	**Classification**	**Ordinal Regression**
Input	Feature vector x		
Output	Real number $y = f(x)$	Category $y = \text{classifier}(x)$	Ordered category $y = \text{thresold}(f(x))$
Model	Ranking function $f(x)$		
Loss Function	Regression loss	Classification loss	Ordinal regression loss

Pointwise Approach

	Ranking
Input	$\mathbf{x} = \{x_i\}_{i=1}^{n}$
Output	Permutation on vectors $\boldsymbol{\pi} = \mathrm{sort}(\{f(x_i)\}_{i=1}^{n})$
Model	Ranking function $f(x)$
Loss Function	Ranking evaluation measure

Pairwise Approach

- Transforming ranking to pairwise classification
- Query-document group structure is ignored

Pairwise Approach

	Learning	Ranking
Input	Ordered feature vector pair $(x_i, x_j), x_i > x_j$	Feature vectors $\mathbf{x} = \{x_i\}_{i=1}^{n}$
Output	Classification on order of vector pair $y_{i,j} = \text{classifier}\,(x_i - x_j)$	Permutation on vectors $\boldsymbol{\pi} = \text{sort}(\{f(x_i)\}_{i=1}^{n})$
Model	Ranking function $f(x)$	
Loss Function	Pairwise classification loss	Ranking evaluation measure

Listwise Approach

- List as instance
- Query-document group structure is used
- Straightforwardly represents learning to rank problem

Listwise Approach

	Learning & Ranking
Input	Feature vectors $\mathbf{x} = \{x_i\}_{i=1}^{n}$
Output	Permutation on feature vectors $\boldsymbol{\pi} = \mathrm{sort}(\{f(x_i)\}_{i=1}^{n})$
Model	Ranking function $f(x)$
Loss Function	Listwise loss function (ranking evaluation measure)

Learning to Rank Methods

- Pointwise Approach
  - Subset Ranking [Cossock and Zhang, 2006]: Regression
  - SVM [Nallapati, 2004]: Binary Classification Using SVM
  - McRank [Li et al 2007]: Multi-Class Classification Using Boosting Tree
  - Prank [Crammer and Singer 2002]: Ordinal Regression Using Perceptron
  - Large Margin [Shashua & Levin 2002]: Ordinal Regression Using SVM

Learning to Rank Methods

- Pairwise Approach
  - Ranking SVM: Pairwise Classification Using SVM
  - RankBoost [Freund et al 2003]: Pairwise Classification Using Boosting
  - RankNet [Burges et al 2005]: Pairwise Classification Using Neural Net
  - Frank [Tsai et al 2007]: Pairwise Classification Using Fidelity Loss and Neural Net
  - GBRank [Zheng et al 2007]: Pairwise Regression Using Boosting Tree
  - IR SVM [Cao et al 2006]: Cost-sensitive Pairwise Classification Using SVM
  - Multiple SVMs [Qin et al 2007]: Multiple SVMs

Learning to Rank Methods

- Listwise Approach
  - ListNet [Cao et al 2007]: Probabilistic Ranking Model
  - ListMLE [Xia et al 2008]: Probabilistic Ranking Model
  - AdaRank [Xu and Li 2007]: Direct Optimization of Evaluation Measure
  - SVM Map [Yue et al 2007]: Direct Optimization of Evaluation Measure
  - PermuRank [Xu et al 2008]: Direct Optimization of Evaluation Measure
  - Soft Rank [Taylor et al 2008]: Approximation of Evaluation Measure
  - Lambda Rank [Burges et al 2007]: Using Implicit Loss Function

Learning to Rank Methods

- Other Methods
  - K-Nearest Neighbor Ranker [Geng et al 2008]
  - Semi-Supervised Learning [Jin et al 2008]

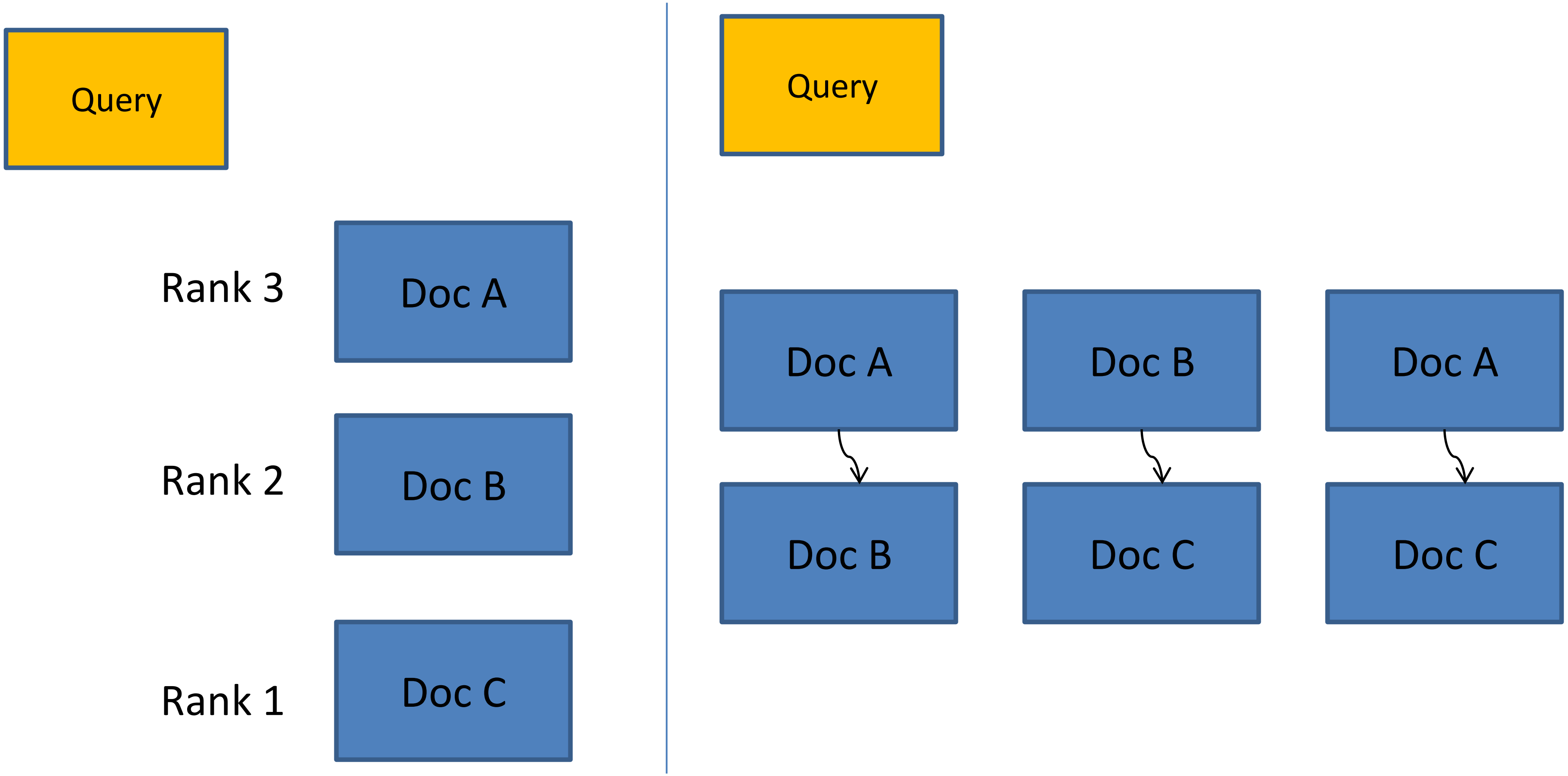
3.2 Pairwise Classification

Pairwise Classification Methods

- Ranking SVM
- RankBoost
- RankNet

Pairwise Classification

- Converting document list to document pairs



Ranking SVM

Transforming Ranking to Pairwise Classification

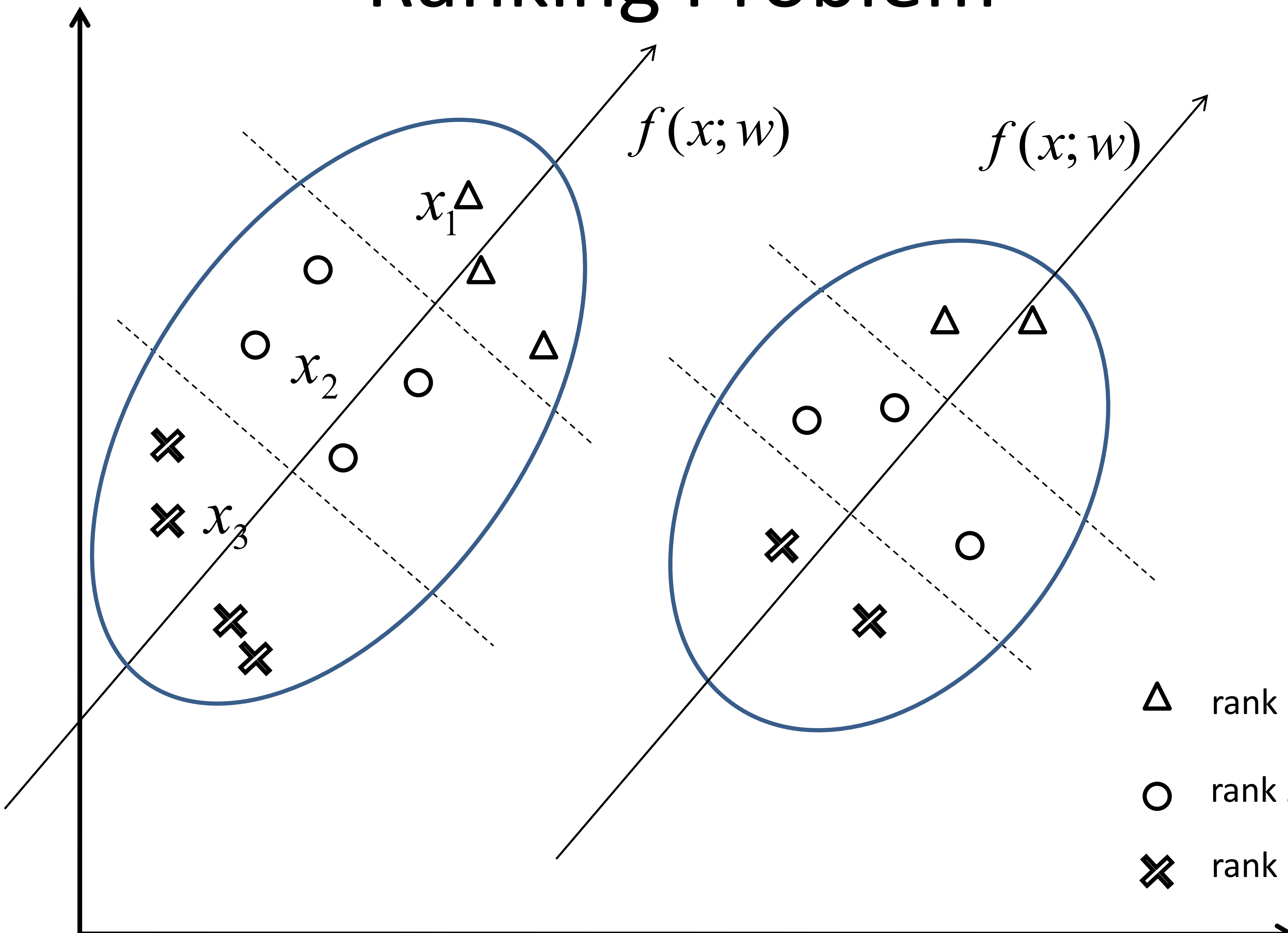
- Input space: *X*
- Ranking function $f : X \rightarrow R$
- Ranking: $x_i \succ x_j \;\Leftrightarrow\; f(x_i; w) > f(x_j; w)$
- Linear ranking function: $f(x; w) = \langle w, x \rangle$

$$\langle w, x_i - x_j \rangle > 0 \quad \Leftrightarrow \quad f(x_i; w) > f(x_j; w)$$

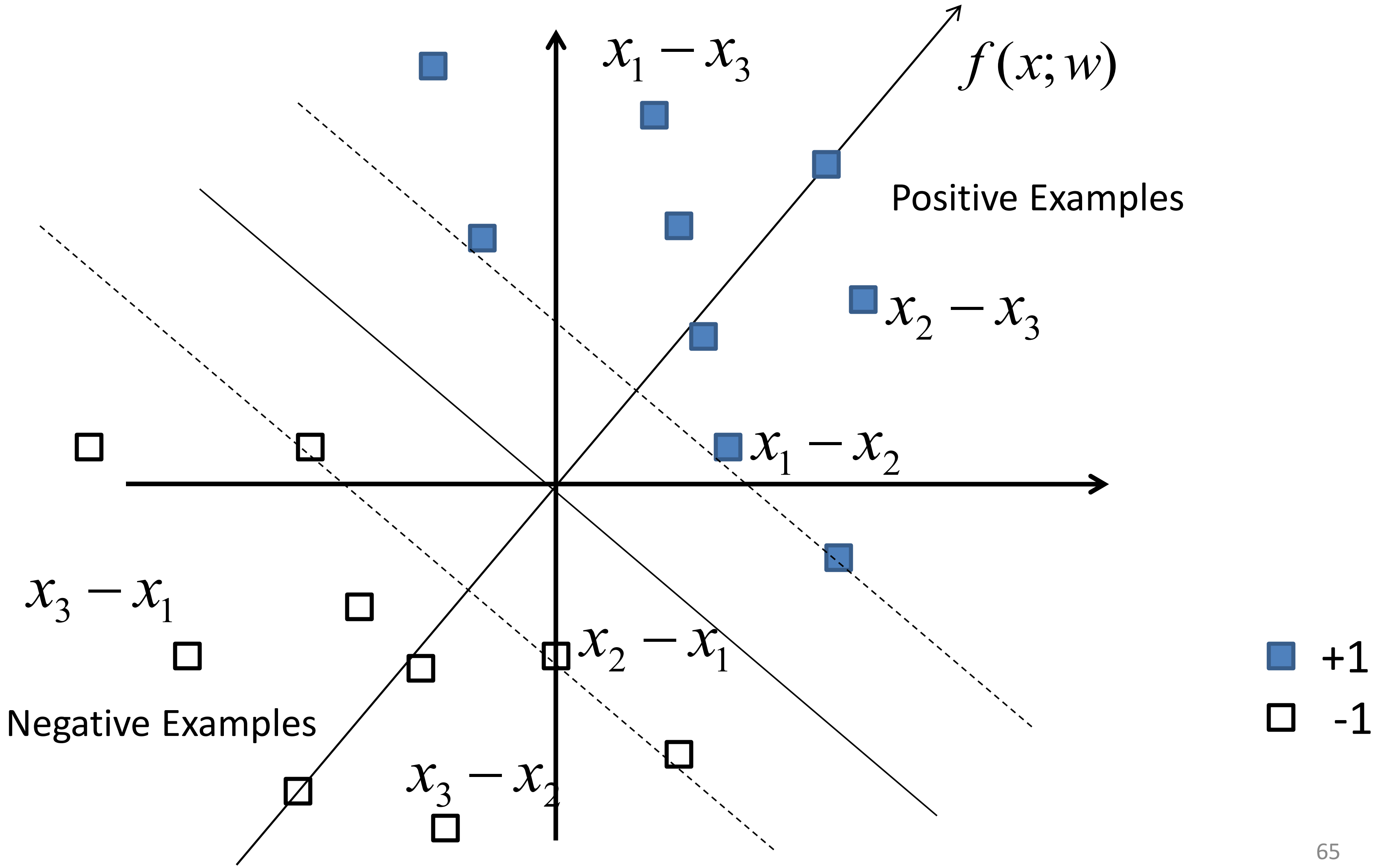
- Transforming to pairwise classification:

$$(x_i - x_j, z), \;\; z = \begin{cases} +1 & x_i \succ x_j \\ -1 & x_j \succ x_i \end{cases}$$

Ranking Problem



Transformed Pairwise Classification Problem



Ranking SVM

- Pairwise classification on differences of feature vectors
- Corresponding positive and negative examples
- Negative examples are redundant and can be discarded
- Hyper plane passes the origin
- Soft Margin and Kernel can be used
- *Ranking SVM =* pairwise classification SVM

Learning of Ranking SVM

$$\min_{w,\xi} \frac{1}{2} \| w \|^2 + C\sum_{i=1}^{l} \xi_i$$

$$z_i \left\langle w, x_i^{(1)} - x_i^{(2)} \right\rangle \geq 1 - \xi_i \quad i = 1, \cdots, l$$

$$\xi_i \geq 0$$

$$\Updownarrow$$

$$\min_{w} \sum_{i=1}^{l} \left[1 - z_i \left\langle w, x_i^{(1)} - x_i^{(2)} \right\rangle \right]_+ + \lambda \| w \|^2$$

$$[s]_+ = \max(0, s) \quad \lambda = \frac{1}{2C}$$

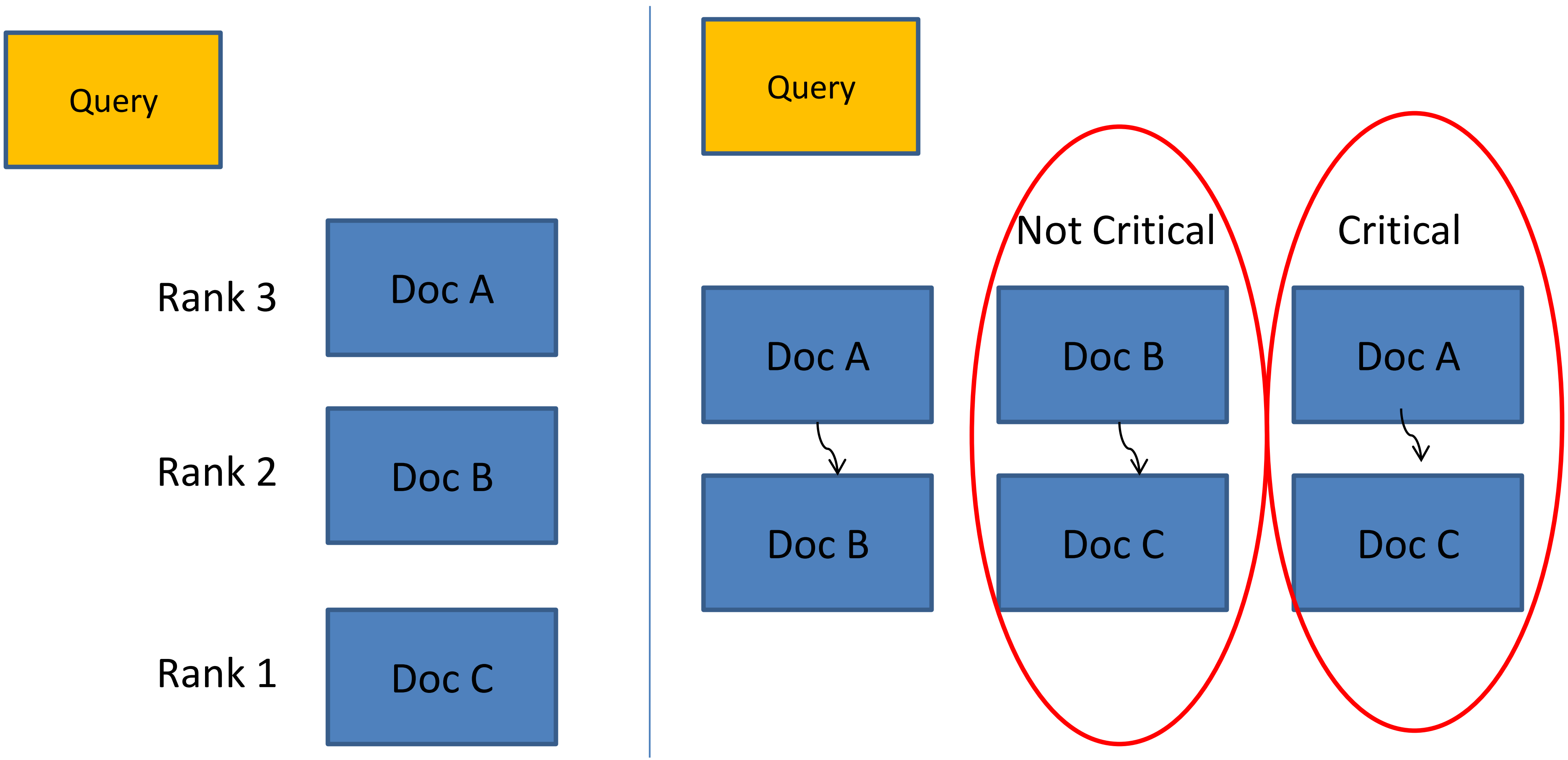
3.3 Cost-sensitive Pairwise Classification

Cost-Sensitive Pairwise Classification Methods

- IR SVM
- Multiple SVMs

Cost-sensitive Pairwise Classification

- Converting to document pairs



IR SVM

Problems with Ranking SVM

- Not sufficient emphasis on correct ranking on top

  Ranks: 3, 2, 1

  ranking 1: 2 3 2 1 1 1 1

  ranking 2: 3 2 1 2 1 1 1

  ranking 2 should be better than ranking 1

  Ranking SVM views them as the same
- Numbers of pairs vary according to queries

  q1: 3 2 2 1 1 1 1

  q2: 3 3 2 2 2 1 1 1 1 1

  number of pairs for q1 : 2*(2-2) + 4*(3-1) + 8*(2-1) = 14

  number of pairs for q2: 6*(3-2) + 10*(3-1) + 15*(2-1) = 31

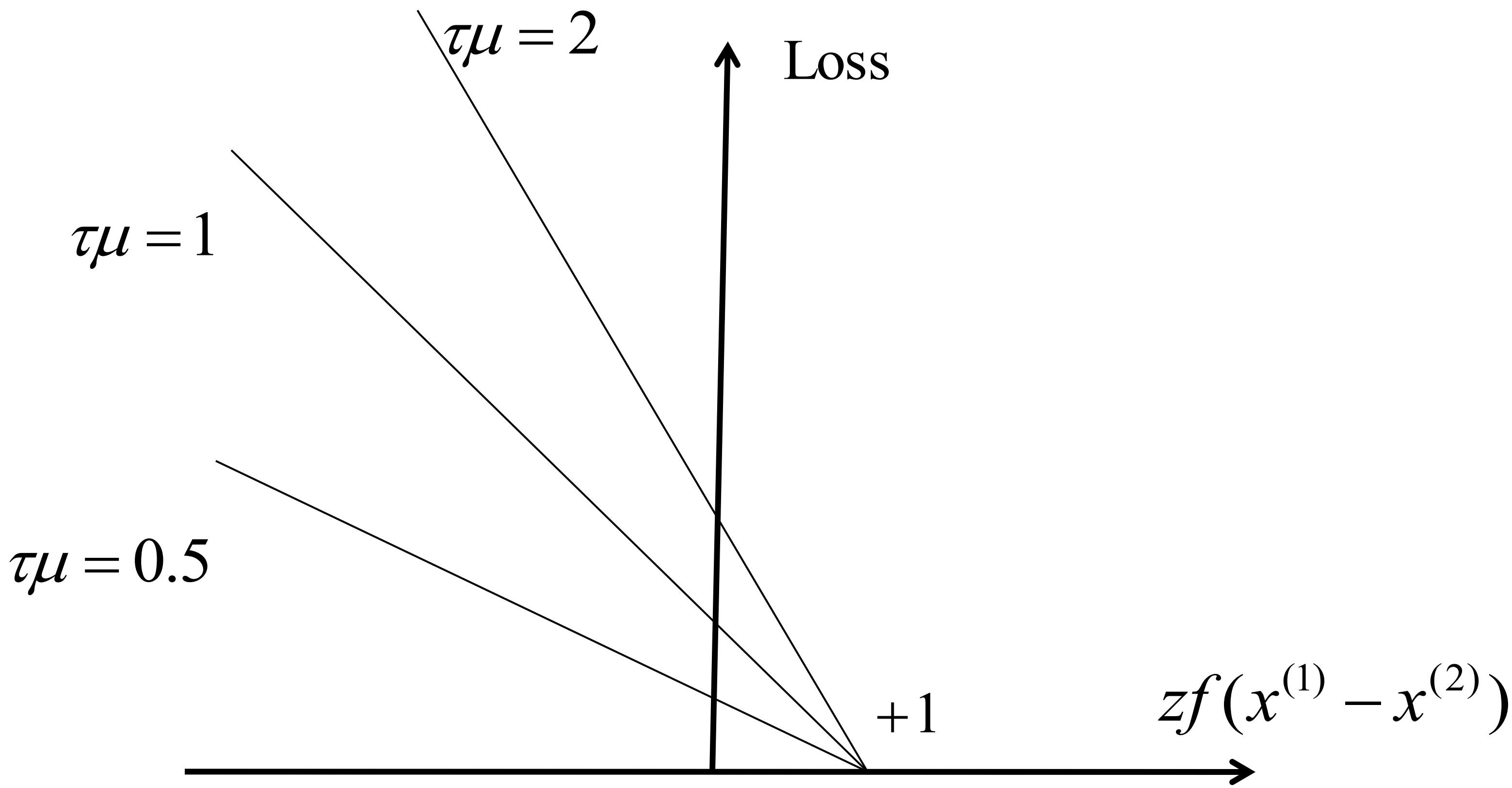
  Ranking SVM is biased toward q2

IR SVM

- Solving the two problems of Ranking SVM
- Higher weight on important rank pairs $\tau_{k(i)}$
- Normalization weight on pairs in query $\mu_{q(i)}$
- IR SVM = Ranking SVM using modified hinge loss

Modified Hinge Loss function

$$\min_{w} \sum_{i=1}^{l} \tau_{k(i)} \mu_{q(i)} \left[1 - z_i \left\langle w, x_i^{(1)} - x_i^{(2)} \right\rangle \right]_+ + \lambda \| w \|^2$$



Learning of IR SVM

$$\min_{w} \sum_{i=1}^{l} \tau_{k(i)} \mu_{q(i)} \left[1 - z_i \left\langle w, x_i^{(1)} - x_i^{(2)} \right\rangle\right]_+ + \lambda \| w \|^2$$

$$\Updownarrow$$

$$\min_{w,\xi} \frac{1}{2} \| w \|^2 + \sum_{i=1}^{l} C_i \xi_i$$

$$z_i \left\langle w, x_i^{(1)} - x_i^{(2)} \right\rangle \geq 1 - \xi_i \quad i = 1, \cdots, l$$

$$\xi_i \geq 0$$

$$C_i = \frac{\tau_{k(i)} \mu_{q(i)}}{2\lambda}$$

3.4 Probabilistic Model for Ranking

Methods of Using Probabilistic Models

- ListNet
- ListMLE

ListNet and ListMLE

Plackett-Luce Model (Permutation Probability)

- Probability of permutation π is defined as

$$P(\pi) = \prod_{i=1}^{n} \frac{s_{\pi(i)}}{\sum_{j=i}^{n} s_{\pi(j)}}$$

- Example:

$$P(\text{ABC}) = \boxed{\frac{s_A}{s_A + s_B + s_C}} \cdot \boxed{\frac{s_B}{s_B + s_C}} \cdot \boxed{\frac{s_C}{s_C}}$$

P(A ranked No.1)

P(B ranked No.2 | A ranked No.1)

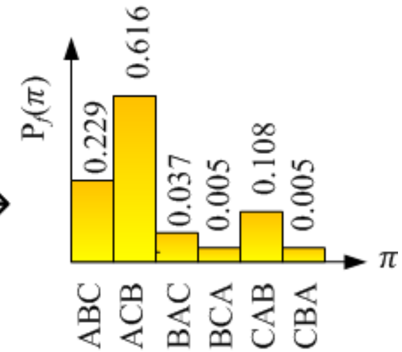
P(C ranked No.3 | A ranked No.1, B ranked No.2)

Properties of Plackett-Luce Model

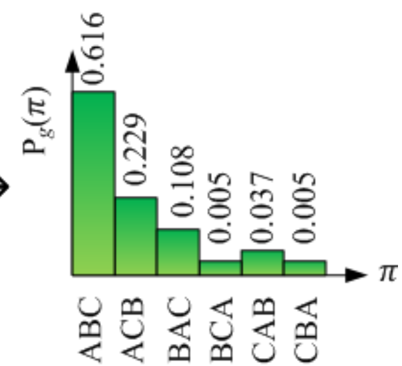
- Objects: ABC
- Scores: $s_{\mathrm{A}} = 5, s_{\mathrm{B}} = 3, s_{\mathrm{C}} = 1$
- Property 1: *P*(ABC) is largest, *P*(CBA) is smallest
- Property 2: swap B and C in ABC, *P*(ABC) > *P*(ACB)

KL Divergence between Permutation Probability Distributions

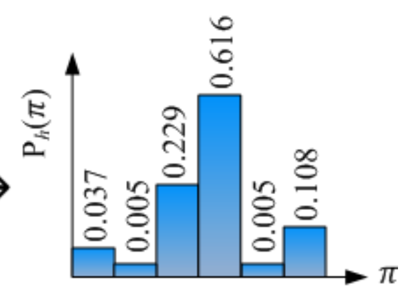
f: $f(\mathrm{A}) = 3, f(\mathrm{B})=0, f(\mathrm{C})=1$;
Ranking by f: ABC $\leftrightarrow$



g: $g(\mathrm{A}) = 6, g(\mathrm{B})=4, g(\mathrm{C})=3$;
Ranking by g: ABC $\leftrightarrow$



h: $h(\mathrm{A}) = 4, h(\mathrm{B})=6, h(\mathrm{C})=3$;
Ranking by h: ACB $\leftrightarrow$



ListNet

- Parameterized Plackett-Luce Model

$$s = \exp(f(x;w))$$

$$P(G(x_1 \cdots x_k)) = \prod_{i=1}^{k} \frac{s_{x_i}}{\sum_{j=i}^{n} s_{x_j}}$$

- Ranking Model: $f(x;w)$ = Neural Net

ListNet (cont')

- Loss function = KL-divergence between two Top-k probability distributions from ground truth and ranking model

$$L(w) = -\sum_{q \in Q} \sum_{\pi \in \Omega^k} \left(\prod_{i=1}^{k} \frac{\exp(y_i))}{\sum_{j=i}^{n} \exp(y_j)} \right) \log \left(\prod_{i=1}^{k} \frac{\exp(f(x_i; w))}{\sum_{j=i}^{n} \exp(f(x_j; w))} \right)$$

- Algorithm = Gradient Descent

ListMLE

- Parameterized Plackett-Luce Model

$$s = \exp(f(x;w))$$

$$P(G(x_1 \cdots x_k)) = \prod_{i=1}^{k} \frac{s_{x_i}}{\sum_{j=i}^{n} s_{x_j}}$$

- Maximum Likelihood Estimation

$$L(w) = -\sum_{q \in Q} \log\left(\prod_{i=1}^{k} \frac{\exp(f(x_i;w))}{\sum_{j=i}^{n} \exp(f(x_j;w))}\right)$$

Plackett-Luce Model (Top-k Probability)

- Computation of permutation probabilities is intractable
- Top-k probability
  - Defining Top-k subgroup $G(o_1 \ldots o_k)$ containing all permutations whose top-k objects are $o_1, \ldots, o_k$
  - $P\big(G(o_1 \cdots o_k)\big) = \prod_{i=1}^{k} \frac{s_{o_i}}{\sum_{j=i}^{n} s_{o_j}}$
  - Time complexity of computation : from $n!$ to $n!/(n-k)!$
- Example: $P\big(G(\mathrm{A})\big) = \frac{s_A}{s_A + s_B + s_C}$

3.5 Direct Optimization of Evaluation Measures

Direction Optimization Methods

- AdaRank
- PermuRank
- SVM MAP

Listwise Loss

$$
\begin{array}{lll}
q_1 \left\{ \begin{array}{l} x_{1,1} \\ x_{1,2} \\ \vdots \\ x_{1,n_1} \end{array} \right. &
\begin{array}{l} \pi_{1,1} \\ \pi_{1,2} \\ \\ \pi_{1,n_1} \end{array} &
\begin{array}{l} y_{1,1} \\ y_{1,2} \\ \\ y_{1,n_1} \end{array} \\
& \vdots & \\
& \vdots & \\
q_m \left\{ \begin{array}{l} x_{m,1} \\ x_{m,2} \\ \vdots \\ x_{m,n_m} \end{array} \right. &
\begin{array}{l} \pi_{m,1} \\ \pi_{m,2} \\ \\ \pi_{m,n_m} \end{array} &
\begin{array}{l} y_{m,1} \\ y_{m,2} \\ \\ y_{m,n_m} \end{array}
\end{array}
$$

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

$$\Downarrow$$

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

AdaRank, SVM-MAP, PermuRank

- Optimizing different upper bounds (surrogate loss functions)

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

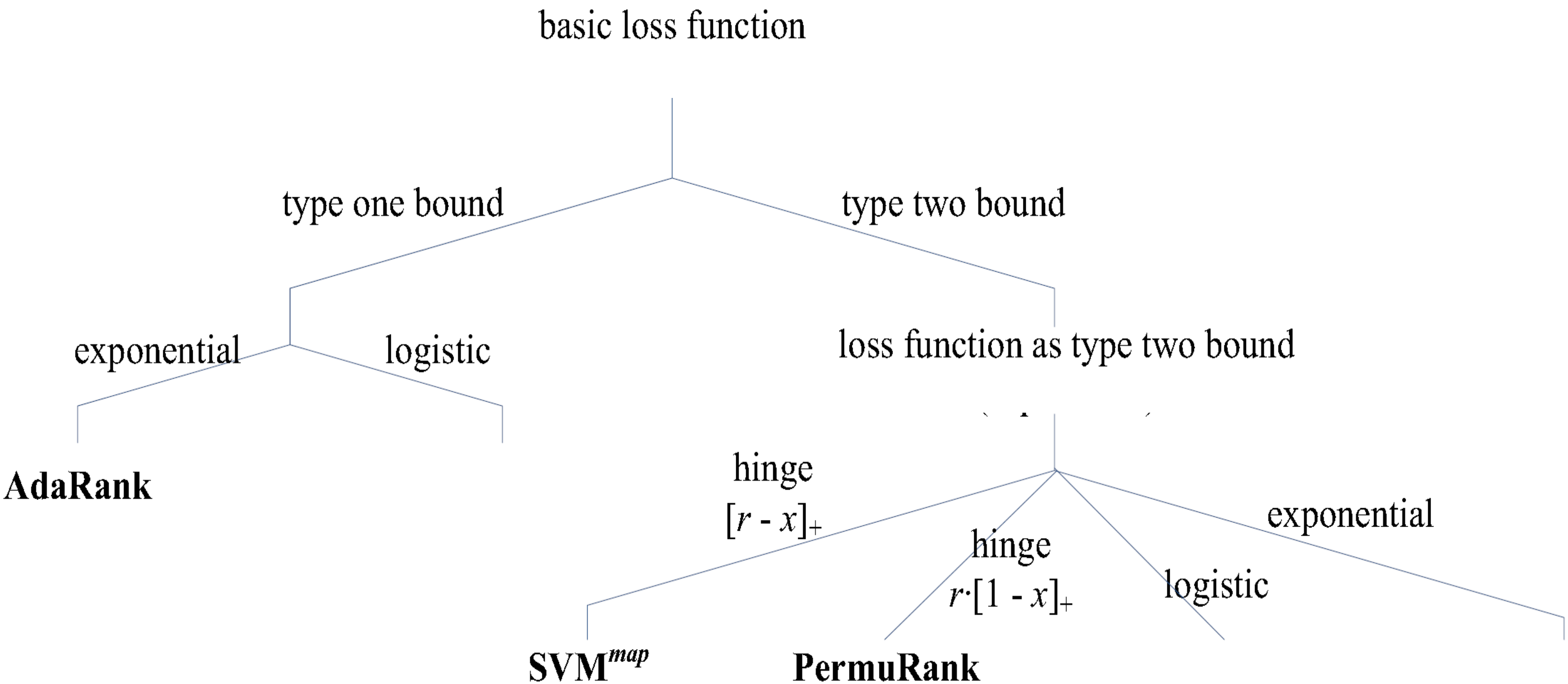
- Type One Upper Bound

$$\sum_{i=1}^{m} \exp\{-E(\pi_i, \mathbf{y}_i)\} \quad \sum_{i=1}^{m} \log_2 \left(1 + e^{-E(\pi_i, \mathbf{y}_i)}\right)$$

- Type Two Upper Bound

$$\sum_{i=1}^{m} \max_{\pi_i^* \in \Pi_i^*; \pi_i \in \Pi_i \setminus \Pi_i^*} ((E(\pi_i^*, \mathbf{y}_i) - E(\pi_i, \mathbf{y}_i)) \cdot [\![(F(q_i, \mathbf{d}_i, \pi_i^*) \leq F(q_i, \mathbf{d}_i, \pi_i))]\!])$$

Relations between Upper Bounds



AdaRank

AdaRank

- Optimizing exponential loss function
- Algorithm: AdaBoost-like algorithm for ranking

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]

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

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

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

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

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

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* $t = 1, 2, \cdots, T$.

SVM MAP

Global Ranking Function

$$\sigma_i = \arg\max_{\sigma \in \Pi_i} F(q_i, \mathbf{d}_i, \sigma).$$

$$F(q_i, \mathbf{d}_i, \pi_i) = w^\top \Phi(q_i, \mathbf{d}_i, \pi_i),$$

$$\Phi(q_i, \mathbf{d}_i, \pi_i) = \frac{1}{n(q_i) \cdot (n(q_i) - 1)} \sum_{k,l:k<l} [z_{kl}(\phi(q_i, d_{ik}) - \phi(q_i, d_{il}))],$$

Equivalent to ranking with local ranking function (plus sorting)

$$f(q_i, d_{ij}) = w^T \phi(q_i, d_{ij})$$

Ranking with Global Function

$$f_A = w^T \phi_A, \quad f_B = w^T \phi_B, \quad f_C = w^T \phi_C$$

$$f_A > f_B > f_C$$

- ABC $\quad F_{ABC} = \frac{1}{6}\left((f_A - f_B) + (f_B - f_C) + (f_A - f_C)\right)$

- ACB $\quad F_{ACB} = \frac{1}{6}\left((f_A - f_C) + (f_C - f_B) + (f_A - f_B)\right)$

$$F_{ABC} > F_{ACB}$$

Type Two Bound

$$R(F) = \sum_{i=1}^{m} (E(\pi_i^*, \mathbf{y}_i) - E(\pi_i, \mathbf{y}_i)) = \sum_{i=1}^{m} (1 - E(\pi_i, \mathbf{y}_i)),$$

$$\sum_{i=1}^{m} \max_{\pi_i^* \in \Pi_i^*; \pi_i \in \Pi_i \setminus \Pi_i^*} ((E(\pi_i^*, \mathbf{y}_i) - E(\pi_i, \mathbf{y}_i)) \cdot [\![(F(q_i, \mathbf{d}_i, \pi_i^*) \leq F(q_i, \mathbf{d}_i, \pi_i))]\!]),$$

SVM MAP

$$\sum_{i=1}^{m}\Big[\max_{\pi_i^* \in \Pi_i^*; \pi_i \in \Pi_i \setminus \Pi_i^*} ((E(\pi_i^*, \mathbf{y}_i) - E(\pi_i, \mathbf{y}_i))$$

$$- (F(q_i, \mathbf{d}_i, \pi_i^*) - F(q_i, \mathbf{d}_i, \pi_i)))\Big]_+ + \lambda \parallel \vec{w} \parallel^2 .$$

$$\min_{\vec{w}; \xi \geq 0} \frac{1}{2}\|\vec{w}\|^2 + \frac{C}{m}\sum_{i=1}^{m}\xi_i$$

$$s.t. \quad \forall i, \forall \pi_i^* \in \Pi_i^*, \forall \pi_i \in \Pi_i \setminus \Pi_i^* :$$

$$F(q_i, \mathbf{d}_i, \pi_i^*) - F(q_i, \mathbf{d}_i, \pi_i) \geq E(\pi_i^*, \mathbf{y}_i) - E(\pi_i, \mathbf{y}_i) - \xi_i,$$

3.6 Approximation of Evaluation Measures

Method based on Approximation of Evaluation Measure

- SoftRank
- LambdaRank

SoftRank

SoftRank

- Evaluation measures are not smooth, due to use of sorting
- Probabilistically approximate evaluation measures (Soft NDCG)
- Model: Neural Net
- Algorithm: Gradient Descent

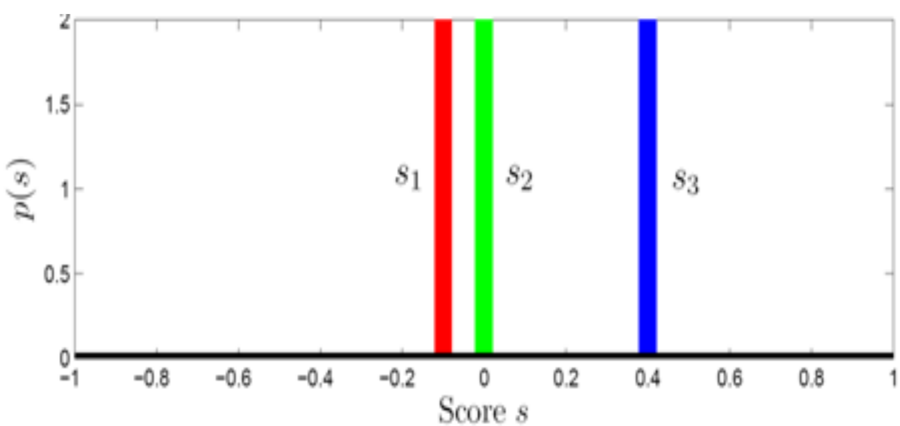
SoftRank Algorithm

- Calculate score distributions of documents
- Calculate position distributions of documents
- Compute SoftNDCG
- Optimize SoftNDCG by Gradient Descent

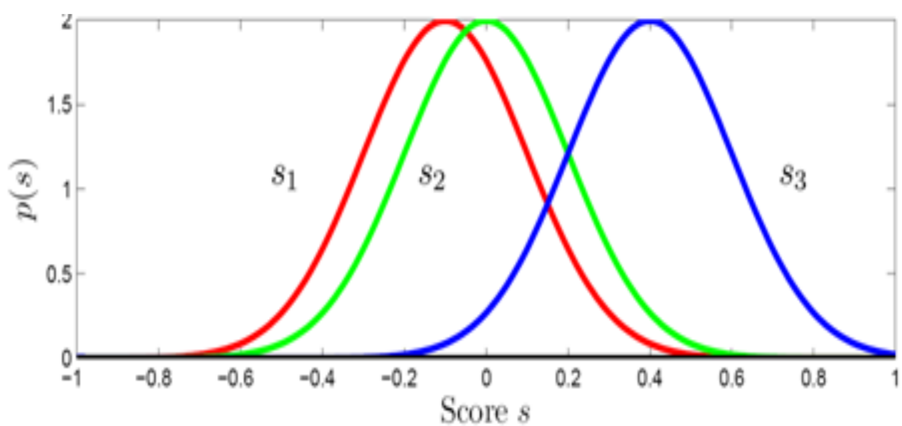
Score Distribution of Document

- Assuming that score of document s_j follows Gaussian distribution

$$p(s_j) = N(s_j \mid f(x_j), \sigma_s^2)$$



Deterministic Score



Probabilistic Score

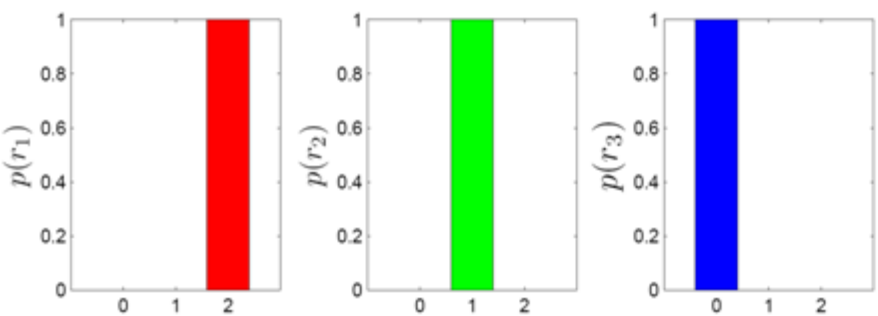
Position Distribution of Document

- Probability of document *i* being ranked before document *j*

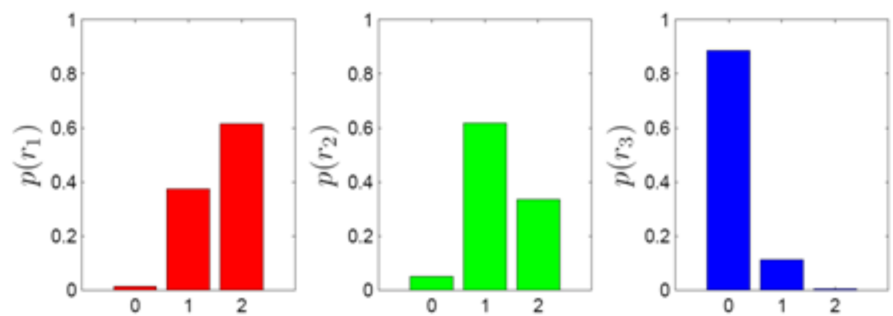
$$\pi_{ij} = \int_0^{\infty} N(s \mid f(x_i) - f(x_j), 2\sigma^2) ds$$

- Position distribution of document (recursive definition)

$$p_j^{(i)}(r) = p_j^{(i-1)}(r-1)\pi_{ij} + p_j^{(i-1)}(r)(1-\pi_{ij})$$



Deterministic Position Distribution



Probabilistic Position Distribution

Soft NDCG

- Re-writing NDCG as document (feature vector) indexed

$$G = G_{max}^{-1} \sum_{j=1}^{N} g(j) D(r_j).$$

- Only position discount is affected by ranking model
- Assume that positions of document are probabilistically determined by ranking model, rather than deterministically

Soft NDCG (cont')

- Soft NDCG

$$\mathcal{G} = G_{max}^{-1} \sum_{j=1}^{N} g(j) E[D(r_j)].$$

$$\mathcal{G} = G_{max}^{-1} \sum_{j=1}^{N} g(j) \sum_{r=0}^{N-1} D(r) p_j(r)$$

- Only need to calculate probabilities of positions of document $p_j(r)$

3.7 Evaluation Results

LETOR Data Set

- Available at
  - http://research.microsoft.com/~letor/
- Data Corpora: TREC, OHSUMED
- Training/Validation/Test split
- Standard IR Features

Evaluation Results

- Pairwise approach and listwise approach perform better than pointwise approach
- Listwise approach performs better than pairwise approach in most cases
- Listwise approach
  - ListMLE, ListNet, AdaRank, PermuRank, SVM-MAP
- Pairwise approach
  - Ranking SVM, RankNet, RankBoost
- Pointwise approach
  - Linear Regression

4. Learning to Rank Theory

Learning to Rank Theory

- 4.1 Generalization Analysis of Pairwise Methods
- 4.2 Generalization Analysis of Listwise Methods

Learning to Rank Theory

- Pairwise Approach
  – Generalization Analysis [Lan et al 2008]
- Listwise Approach
  – Generalization Analysis [Lan et al 2009]
  – Consistency Analysis [Xia et al 2008]

4.1 Generalization Analysis of Pairwise Methods

Pairwise Learning Framework (Query Level)

- Data is represented as $(z^{(q)}, y^{(q)})$, where $z^{(q)}$ is pair of feature vectors $(x_1^{(q)}, x_2^{(q)})$ and $y^{(q)}$ is ground-truth on their order, w.r.t. query *q*
- (z, y) are random variables according to distribution $P(\cdot,\cdot)$
- Training Data: $S = (q_1, S_1)\cdots(q_m, S_m)$

$$S_i = \left\{(z_1^{(i)}, y_1^{(i)}), \cdots, (z_{n_i}^{(i)}, y_{n_i}^{(i)})\right\}, i = 1, \cdots m$$

- Expected Risk:

$$R_l(f) = \int_Q L(f, q) P_Q(dq) \quad L(f, q) = \int_{X^2 \times Y} l(f; z^{(q)}, y^{(q)}) D_q(dz^{(q)}, dy^{(q)})$$

- Empirical Risk:

$$\hat{R}_l(f; S) = \frac{1}{m} \sum_{i=1}^{m} \hat{L}(f; q_i) \qquad \hat{L}(f; q_i) = \frac{1}{n_i} \sum_{j=1}^{n_i} l(f; z_j^{(i)}, y_j^{(i)})$$

Generalization Analysis

- Goal = to minimize expected risk $R_l(f)$
- Distribution is unknown we instead minimize empirical risk $\hat{R}_l(f;S)$
- Generalization analysis is concerned with upper bound of difference between expected and empirical risks

Generalization Analysis Using Stability

- With probability at least $1-\delta$

$$R_l(f) \le \hat{R}_l(f;S) + 2\tau(m) + \left(4m\tau(m) + B\right)\sqrt{\frac{\ln 1/\delta}{2m}}$$

- The bound is related to number of training queries *m*, and stability of algorithm
- If $\tau(m)$ goes to 0 very fast when *m* approaches infinity, then the bound will be tight.

Stability of Ranking SVM and IRSVM

- Ranking SVM

$$\tau(m) = \frac{4\kappa^2}{\lambda m} \frac{1 + \frac{\sigma}{\mu\sqrt{\delta/m}}}{1 - \frac{\varepsilon}{\mu}} = O\left(\frac{1}{\sqrt{m}}\right)$$

- IR SVM

$$\tau(m) = \frac{4\kappa^2}{\lambda m} = O\left(\frac{1}{m}\right)$$

- IR SVM > Ranking SVM

4.2 Generalization Analysis of Listwise Methods

Listwise Learning Framework

- Data is represented as (z, y), where z is feature vector set $z = (x_1, \cdots, x_n)$ and y is ground-truth permutation
- (z, y) are random variables according to distribution $P(\cdot,\cdot)$
- Training Data: $(z_1, y_1), \cdots, (z_m, y_m)$
- Expected Risk:

$$R_l(f) = \int_{Z \times Y} l(f; z, y) P(dz, dy)$$

- Empirical Risk:

$$R_l(f; S) = \frac{1}{m} \sum_{i=1}^{m} l(f; z_i, y_i)$$

Generalization Analysis

- Goal = to minimize expected risk $R_{l_A}(f)$
- Distribution is unknown we instead minimize empirical risk $\hat{R}_{l_A}(f;S)$
- Generalization analysis is concerned with upper bound of difference between expected and empirical risks

$$\sup(R_{l_A}(f)-\hat{R}_{l_A}(f;S))$$

Generalization Analysis Using Rademacher Complexity

- With probability at least $1-\delta$

$$\sup(R_{l_A}(f)-\hat{R}_{l_A}(f;S)) \le \frac{4BM}{\sqrt{m}} C_A(\phi)N(\phi)+\sqrt{\frac{2\ln 2/\delta}{m}}$$

- The bound is related to:
  - $C_A(\phi)$, algorithm-dependent factor, determined by loss function and transformation function ϕ
  - $N(\phi)$, algorithm-independent factor, only determined by transformation function ϕ
  - when n approaches infinity, generalization bound vanishes at rate of $O(\frac{1}{\sqrt{m}})$.

Generalization Analysis Using Rademacher Complexity (cont')

- ListMLE

$$C_{ListMLE}(\phi) = \frac{2}{\phi(-BM)(\log n + \log \frac{\phi(BM)}{\phi(-BM)}}$$

- ListNet

$$C_{ListNet}(\phi) = \frac{2n!}{\phi(-BM)(\log n + \log \frac{\phi(BM)}{\phi(-BM)}}$$

- ListMLE > ListNet

5. Learning to Rank Applications

Learning to Rank Applications

- Search [Burges et al 2005]
- Recommender System [Freund et al 2003]
- Key Phrase Extraction [Jiang et al 2009]

Recommender System (Collaborative Filtering)

- Problem formulation
  - Input: users’ ratings on some items
  - Output: users’ ratings on other items
  - Assumption: users sharing same ratings on input items tend to agree on new items
- Solutions
  - Classification
  - Ordinal Regression
  - Learning to Rank

Recommender System

	Item1	**Item2**	**Item3**	**...**	
User1	5	4			
User2	1		2		2
...		?	?	?	
UserM	4	3			

Recommender System Using RankBoost

- Ranking items according to users
- Justification: users tend to rate on different scales
- Method: RankBoost
- Result: RankBoost > Nearest Neighbor

Key Phrase Extraction

- Problem formulation
  - Input: document
  - Output: keyphrases of document
  - Two steps: phrase extraction and keyphrase identification
- Traditional approach
  - Classification: keyphrase vs non-keyphrase

Key Phrase Extraction Using Ranking SVM

- Ranking of phrases as keyphrases
- Justification: keyphrase or non-keyphrase is relative
- Method: Ranking SVM
- Result: Ranking SVM > SVM

6. Future Directions of Learning to Rank Research

New Issues to be Further Studied

- Learning from implicit data
  - Automatically generate labeled data from implicit feedback
- Model (feature) learning
  - Automatically learn features such as BM25
- Global ranking
  - Using features of current document as well as relations with other documents

New Issues to be Further Studied (cont')

- Query-dependent ranking
  - Creating different ranking models for different queries (in search)
- New applications
  - Machine Translation, etc

7. Summary

Content of Tutorial

- Introduction
- Learning to Rank Problem
- Learning to Rank Methods
- Learning to Rank Theory
- Learning to Rank Applications
- Future Directions of Learning to Rank Research

Takeaway Message

- Learning to Rank = branch of Machine Learning
- Learning to Rank = learning for ranking objects given subject
- Learning to Rank is different from ordinal regression
- Learning to Rank has been successfully applied to search
- Existing approaches: pointwise, pairwise, listwise

Acknowledgements

- I am grateful to Wei-Ying Ma, Hsiao-Wuen Hon, and Harry Shum for their encouragement and guidance to our work on learning to rank.
- I thank my colleagues Tie-Yan Liu, Jun Xu, Tao Qin, Yunbo Cao, Yunhua Hu, our interns Zhe Cao, Ming-Feng Tsai, Xiubo Geng, Yanyan Lan, Fen Xia, Ming Li, Xin Jiang, our internal collaborators Chris Burges, Michael Taylor, Stephen Robertson, Dmitriy Meyerzon, Victor Poznanski, Rangan Majumder, Girish Kumar, Steven Yao, and our external collaborators Rong Jin, Zhi-Hua Zhou, Hamed Valizadegan, Cheng Xiang Zhai, Thorsten Joachims. We have been working together on Learning to Rank. This made possible for me to give this tutorial.
- I appreciate Tie-Yan Liu and Jun Xu for their helping creating some of the slides of this tutorial.

Contact: hangli@microsoft.com

References

- S. Agarwal, T. Graepel, T. Herbrich, S. Har-Peled, and D. Roth. Generalization bounds for the area under the roc curve. Journal of Machine Learning, 6:393-425, 2005.
- S. Agarwal and P. Niyogi. Stability and generalization of bipartite ranking algorithms. In COLT 2005, pages 32-47, 2005.
- E. Agichtein, E. Brill, S. T. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In SIGIR 2006, pages 3-10, 2006.
- N. Ailon and MehryarMohri. An efficient reduction from ranking to classification. In COLT, 2008.
- H. Almeida, M. Goncalves, M. Cristo, and P. Calado. A combined component approach for finding collection-adapted ranking functions based on genetic programming. In SIGIR 2007, pages 399-406, 2007.
- M.-R. Amini, T.-V. Truong, and C. Goutte. A boosting algorithm for learning bipartite ranking functions with partially labeled data. In SIGIR 2008, pages 99-106, 2008.
- J.-A. Aslam, E. Kanoulas, V. Pavlu, S. Savev, E. Yilmaz. Document Selection Methodologies for Efficient and Effective Learning-to-Rank. In SIGIR 2009.
- M.-F. Balcan, N. Bansal, A. Beygelzimer, D. Coppersmith, J. Langford, and G. B. Sorkin. Robust reductions from ranking to classification. In COLT, 2007.
- S. Banerjee, S. Chakrabarti, and G. Ramakrishnan. Learning to Rank for Quantity Consensus Queries. In SIGIR 2009.
- B. Bartell, G. W. Cottrell, and R. Belew. Learning to retrieve information. In SCC 1995, 1995.

References

- C. J. Burges, R. Ragno, and Q. V. Le. Learning to rank with nonsmooth cost functions. In NIPS 2006, pages 395-402, 2006.
- C. J. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In ICML 2005, pages 89-96, 2005.
- Y. Cao, J. Xu, T-Y Liu, H. Li, Y. Huang, H.-W. Hon. Adapting Ranking SVM to Document Retrieval. In SIGIR'06, pages 186-193, 2006.
- G. Cao, J. Nie, L. Si, J. Bai. Learning to Rank Documents for Ad-Hoc Retrieval with Regularized Models, SIGIR 2007 workshop: Learning to Rank for Information Retrieval, 2007
- Z. Cao, T. Qin, T.Y. Liu, M.F. Tsai, and H. Li: Learning to Rank: From Pairwise Approach to Listwise Approach, In ICML 2007, pages 129-136, 2007.
- V. R. Carvalho, J. L. Elsas, W. W. Cohen, and J. G. Carbonell. A metalearning approach for robust rank learning. In SIGIR 2008 workshop on Learning to Rank for Information Retrieval, 2008.
- S. Chakrabarti, R. Khanna, U. Sawant, and C. Bhattacharyya. Structured learning for non-smooth ranking losses. In SIGKDD 2008, pages 88-96, 2008.
- O. Chapelle, Q. Le, and A. Smola. Large margin optimization of ranking measures. In NIPS workshop on Machine Learning for Web Search, 2007.
- O. Chapelle and Y. Zhang. A Dynamic Bayesian Network Click Model for Web Search Ranking. In WWW 2009, pages 1-10, 2009.
- S. J. M. Clémençon and N. Vayatis. Overlaying classifiers: a practical approach for optimal ranking . In NIPS 2008, pages 305-312, 2008.

References

- S. J.M. Clémençon and N. Vayatis. Empirical performance maximization for linear rank statistics. In NIPS 2008, pages 313-320, 2008.
- R. Cummins and C. O'Riordan. Learning in a Pairwise Term-Term Proximity Framework for Information. In SIGIR 2009.
- W. Cheng, J. Huehn, and E. Huellermeier. Decision Tree and Instance-Based Learning for Label Ranking. In ICML 2009.
- W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. Journal of Machine Learning Research, 6:1019-1041, 2005.
- W. Chu and Z. Ghahramani. Preference learning with Gaussian processes. In ICML 2005, pages 137-144, 2005.
- W. Chu and S. S. Keerthi. New approaches to support vector ordinal regression. In ICML 2005, pages 145-152, 2005.
- S. Clemenson, G. Lugosi, and N. Vayatis. Ranking and scoring using empirical risk minimization. In COLT 2005, 2005.
- C. Cortes, M. Mohri, and A. Rastogi. Magnitude-preserving ranking algorithms. In ICML 2007, pages 169-176, 2007.
- W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. In NIPS 1998, volume 10, pages 243-270, 1998.
- D. Cossock and T. Zhang. Subset ranking using regression. In COLT 2006, pages 605-619, 2006.

References

- K. Crammer and Y. Singer. Pranking with ranking. In NIPS 2002, pages 641-647, 2002.
- Pinar Donmez, Krysta M. Svore , Christopher J.C. Burges. On the Local Optimality of LambdaRank . In SIGIR 2009.
- K. Duh and K. Kirchhoff. Learning to rank with partially-labeled data. In SIGIR 2008, pages 251-258, 2008.
- J. L. Elsas, V. R. Carvalho, and J. G. Carbonell. Fast Learning of Document Ranking Functions with the Committee Perceptron, In WSDM 2008, 2008.
- W. Fan, M. Gordon, and P. Pathak. On linear mixture of expert approaches to information retrieval. Decision Support System, 42(2):975-987, 2006.
- W. Fan, M. Gordon, and P. Pathak. Genetic programming-based discovery of ranking functions for effective web search. Journal of Management of Information Systems, 21(4):37-56, 2005.
- W. Fan, E. A. Fox, P. Pathak, and H. Wu. The effects of fitness functions on genetic programming based ranking discovery for web search. Journal of American Society for Information Science and Technology, 55(7):628-636, 2004.
- W. Fan, M. Gordon, and P. Pathak. Discovery of context-specific ranking functions for effective information retrieval using genetic programming. IEEE Transactions on Knowledge and Data Engineering, 16(4):523-527, 2004.
- W. Fan, M. Gordon, and P. Pathak. A generic ranking function discovery framework by genetic programming for information retrieval. Information Processing and Management, 40(4):587-602, 2004.
- W. Fan, M. D. Gordon, W. Xi, and E. A. Fox. Ranking function optimization for effective web search by genetic programming: an empirical study. In HICSS 2004, page 105-112, 2004.

References

- Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research, 4:933-969, 2003.
- N. Fuhr. Optimum polynomial retrieval functions based on the probability ranking principle. ACM Transactions on Information Systems, 7(3):183-204, 1989.
- G. Fung, R. Rosales, and B. Krishnapuram, Learning Rankings via Convex Hull Separation, In NIPS 2005 workshop on Learning to Rank, 2005
- J. Gao, H. Qi, X. Xia, and J. Nie. Linear discriminant model for information retrieval. In SIGIR 2005, pages 290-297, 2005.
- X. Geng, T.Y. Liu, T. Qin, and H. Li: Feature selection for ranking. IN SIGIR 2007, pages 407-414, 2007.
- X. Geng, T.Y. Liu, T. Qin, and H. Li: Query Adaptive Ranking, In SIGIR 2008.
- X. Geng, T.Y. Liu, T. Qin, A. Arnold, H. Li, and H.Y. Shum. Query Dependent Ranking with K Nearest Neighbor. In SIGIR2008, pages 115-122, 2008.
- J. Guiver and E. Snelson, Learning to Rank with SoftRank and Gaussian Processes, In SIGIR 2008, 2008.
- J. Guiver and E. Snelson. Bayesian inference for Plackett-Luce ranking models. ICML 2009.
- E. F. Harrington. Online ranking/collaborative filtering using the perceptron algorithm. In ICML, pages 250-257, 2003.

References

- R. Herbrich, K. Obermayer, and T. Graepel. Large margin rank boundaries for ordinal regression. In Advances in Large Margin Classifiers, pages 115-132, 2000.
- J. C. Huang and B. J. Frey Structured ranking learning using cumulative distribution networks. In NIPS 2008, pages 697-704, 2008.
- S. Ji, K. Zhou, C. Liao, Z. Zheng, G.R. Xue, O. Chapelle, G. Sun, and H. Zha. Global Ranking by Exploiting User Clicks. In SIGIR 2009.
- R. Jin, H. Valizadegan, and H. Li: Ranking Refinement and Its Application to Information Retrieval. In WWW2008, pages 397-406, 2008.
- T. Joachims. Optimizing search engines using clickthrough data. In KDD 2002, pages 133-142, 2002.
- S. Kramer, G. Widmer, B. Pfahringer, and M. D. Groeve. Prediction of ordinal classes using regression trees. Funfamenta Informaticae, 34:1-15, 2000.
- Y. Lan, T.Y. Liu, T. Qin, Z. Ma, and H. Li. Query Level Stability and Generalization in Learning to Rank, In ICML2008, pages 512-519, 2008.
- Y. Lan, T.Y. Liu, Z. Ma, and H. Li: Generalization Analysis of Listwise Learning to Rank Algorithms. In ICML 2009
- Q. V. Le and A. J. Smola. Direct Optimization of Ranking Measures. Tech Report CoRR. 2007.
- M. Lease. An Improved Markov Random Field Model for Supporting Verbose Queries. In SIGIR 2009.

References

- G. Lebanon and J. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In ICML 2002, pages 363-370, 2002.
- I. Matveeva, C. Burges, T. Burkard, A. Laucius, and L. Wong. High accuracy retrieval with multiple nested ranker. In SIGIR 2006, pages 437-444, 2006.
- P. Li, C. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In NIPS 2007, 2007.
- T.Y. Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. LETOR: Benchmark Dataset for Research on Learning to Rank for Information Retrieval. In SIGIR 2007 Workshop, 2007.
- Y.T. Liu, T.Y. Liu, T. Qin, Z. Ma, and Hang Li: Supervised rank aggregation. In WWW 2007 pages 481-490, 2007
- M. Volkovs and R. Zemel. BoltzRank: Learning to Maximize Expected Ranking Gain. In ICML 2009.
- D. A. Metzler, W. B. Croft, and A. McCallum. Direct maximization of rank based metrics for information retrieval. In CIIR Technical Report, 2005.
- T. Minka and S. Robertson. Selection bias in the LETOR datasets. In SIGIR 2008 workshop on Learning to Rank for Information Retrieval, 2008
- R. Nallapati. Discriminative models for information retrieval. In SIGIR 2004, pages 64-71, 2004.
- T. Pahikkala, E. Tsivtsivadze, A. Airola, J. Boberg, T. Salakoski, Learning to Rank with Pairwise Regularized Least-Squares. In SIGIR 2007 workshop: Learning to Rank for Information Retrieval, 2007.

References

- T. Qin, X. Zhang, D. Wang, T.Y. Liu, W. Lai, and H. Li: Ranking with multiple hyperplanes. In SIGIR 2007, pages 279-286, 2007.
- T. Qin, T.Y. Liu, X. Zhang, D. Wang, W. Xiong, and H. Li: Learning to Rank Relational Objects and Its Application to Web Search. In WWW 2008.
- T. Qin, T.Y. Liu, X. Zhang, D. Wang, and H. Li. Global Ranking Using Continuous Conditional Random Fields. In NIPS2008, pages 1281-1288, 2008.
- T. Qin, T.Y. Liu, X. Zhang, D. Wang, W. Xiong, and H. Li: Learning to Rank Relational Objects and Its Application to Web Search. In WWW2008, pages 407-416, 2008.
- T. Qin, X. Zhang, M. Tsai, D. Wang, T.Y. Liu, and H. Li. Query-level Loss Functions for Information Retrieval. Information Processing and Management, 44:838-855, Elsevier Science, 2008.
- T. Qin, T. Y. Liu, W. Lai, X. Zhang, D. Wang, and H. Li. Ranking with Multiple Hyperplanes. In SIGIR2007, pages 279-286, 2007.
- F. Radlinski and T. Joachims. Active exploration for learning rankings from clickthrough data. In KDD 2007.
- F. Radlinski and T. Joachims. Query chain: Learning to rank from implicit feedback. In KDD 2005, pages 239-248, 2005.
- F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In ICML 2008, pages 784-791, 2008.
- S. Rajaram and S. Agarwal. Generalization bounds for k-partite ranking. In NIPS 2005 WorkShop on Learning to Rank, 2005.

References

- L. Rigutini, T. Papini, M. Maggini, and F. Scarselli. Learning to rank by a neural-based sorting algorithm. In SIGIR 2008 workshop on Learning to Rank for Information Retrieval, 2008.
- S. Robertson and H. Zaragoza. On rank-based effectiveness measures and optimization. Information Retrieval, 10(3):321-339, 2007.
- C. Rudin, C. Cortes, M. Mohri, and R. E. Schapire, Margin-Based Ranking Meets Boosting in the Middle, In COLT 2005.
- A. Shashua and A. Levin. Ranking with large margin principles: Two approaches. In NIPS2002, pages 937-944, 2002.
- M. Talyor, J. Guiver, and etc. Softrank: Optimising non-smooth rank metrics. In WSDM 2008, pages 77-86, 2008.
- M. Taylor, H. Zaragoza, N. Craswell, S. Robertson, and C. J. Burges. Optimisation methods for ranking functions with multiple parameters. In CIKM 2006, pages 585-593, 2006.
- A. Trotman. Learning to rank. Information Retrieval, 8(3):359-381, 2005.
- M.F. Tsai, T.Y. Liu, T. Qin, H.H. Chen, W.Y. Ma: FRank: a ranking method with fidelity loss. In SIGIR 2007, pages 383-390, 2007.
- N. Usunier, D. Buffoni and P. Gallinari. Ranking with Ordered Weighted Pairwise Classification. In ICML 2009.
- N. Usunier, V. Truong, M. R. Amini, and P. Gallinari, Ranking with Unlabeled Data: A First Study, In NIPS 2005 workshop:Learning to Rank, 2005.

References

- A. Veloso, H. M. de Almeida, M. A. Gon?alves, and W. M. Jr. Learning to rank at query-time using association rules. In SIGIR 2008, pages 267-274, 2008.
- W. Xi, J. Lind, and E. Brill, Learning effective ranking functions for newsgroup search, In SIGIR 2004.
- F. Xia, T.Y. Liu, J. Wang, W. Zhang, and H. Li: Listwise Approach to Learning to Rank –Theory and Algorithm, In ICML2008, pages 1192-1199, 2008.
- J. Xu and H. Li: AdaRank: a boosting algorithm for information retrieval. In SIGIR 2007, pages 391-398, 2007.
- J. Xu, T.Y. Liu, M. Lu, H. Li, and W.Y. Ma: Directly Optimizing Evaluation Measures in Learning to Rank, SIGIR'08, pages 107-114, 2008.
- J. Xu, Y. Cao, H. Li, and Y. Huang. Cost Sensitive Learning of SVM for Ranking. In ECML2006, pages 833-840, 2006.
- J.-Y. Yeh, J.-Y. Lin, and etc. Learning to rank for information retrieval using genetic programming. In LR4IR 2007, 2007.
- H. Yu. SVM selective sampling for ranking with application to data retrieval. In KDD 2005, pages 354-363, 2005.
- Y. Yue and T. Joachims. Predicting diverse subsets using structural SVM. In ICML 2008, pages 1224-1231, 2008.
- Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In SIGIR 2007, pages 271-278, 2007.

References

- Z. Zheng, K. Chen, G. Sun, and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In SIGIR 2007, pages 287-294, 2007.
- Z. Zheng, H. Zha, and G. Sun. Query-level learning to rank using isotonic regression. In SIGIR 2008 workshop on Learning to Rank for Information Retrieval, 2008.
- Z. Zheng, H. Zha, Tong Zhang, Olivier Chapelle, Keke Chen, Gordon Sun. A general boosting method and its application to learning ranking functions for web search. In NIPS 2007.
- K. Zhou, G.-R. Xue, H. Zha, and Y. Yu. Learning to rank with ties. In SIGIR 2008, pages 275-282, 2008.
- O. Zoeter, M. Taylor, E. Snelson, J. Guiver, N. Craswell, and M. Szummer. A decision theoretic framework for ranking using implicit feedback. In SIGIR 2008 workshop on Learning to Rank for Information Retrieval, 2008.