Learning to Rank
from heuristics to theoretic approaches

Hongning Wang
Congratulations

• Job Offer from Bing Core Ranking team
  – Design the ranking module for Bing.com
How should I rank documents?

Answer: Rank by relevance!
Relevance ?!
The Notion of Relevance

Relevance

\[ \Delta(\text{Rep}(q), \text{Rep}(d)) \]

Similarity

\[ P(r=1|q,d) \quad r \in \{0,1\} \]

Probability of Relevance

Relevance constraints

[ Fang et al. 04]

Div. from Randomness

(Amati & Rijsbergen 02)

P(d \rightarrow q) or P(q \rightarrow d)

Probabilistic inference

Different inference system

Prob. concept space model

(Wong & Yao, 95)

Inference network model

(Turtle & Croft, 91)

Learn. To Rank

(Joachims 02, Berges et al. 05)

Vector space model

(Salton et al., 75)

Prob. distr. model

(Wong & Yao, 89)

Regression Model

(Fuhr 89)

Generative Model

Doc generation

Query generation

Classical prob. Model

(Robertson & Sparck Jones, 76)

LM approach

(Ponte & Croft, 98)

(Lafferty & Zhai, 01a)

CS@UVa
Relevance Estimation

• Query matching
  – Language model
  – BM25
  – Vector space cosine similarity

• Document importance
  – PageRank
  – HITS
Did I do a good job of ranking documents?

Document as geometric objects: how to rank documents for full-text...
www.michaellih.org/.../documents-as-geometric-objects-how-to-...
Jul 7, 2011 – In this post I explain the basic ideas of how to rank different documents according to their relevance. The ideas used are very beautiful.

Information Retrieval: Ranking Documents
cciir.cs.umass.edu/~strohman/slides/IR-Intro-Ranking.pdf
File Format: PDF/Adobe Acrobat - View as HTML
Web features, implicit relevance indicators. • Evaluating ranking quality. • Test collections. • Quality metrics. • Training systems to rank documents better. 10 ...

Lucene net - Lucene: How to rank documents according to the...
stackoverflow.com/.../lucene-how-to-rank-documents-according-to-t...
1 answer - Mar 3
Top answer: This will require some work, but you can achieve this using payloads. See answers to this very similar question: How to get a better Lucene/Solr score ...

The Anatomy of a Search Engine
infoclab.stanford.edu/~backrub/google.html
We use font size relative to the rest of the document because when searching, you do not want to rank otherwise identical documents differently just because ...
Did I do a good job of ranking documents?

• IR evaluations metrics
  – Precision@K
  – MAP
  – NDCG
Take advantage of different relevance estimator?

- Ensemble the cues
  - Linear?
    - \( a_1 \times BM25 + a_2 \times LM + a_3 \times PageRank + a_4 \times HITS \)
  - Non-linear?
    - Decision tree
      - \( a_1 = 0.4, a_2 = 0.2, a_3 = 0.3, a_4 = 0.1 \)

- \( r = 1.0 \) | \( r = 0.7 \) | \( r = 0.4 \) | \( r = 0.1 \)

- \( BM25 > 0.5 \) | \( LM > 0.1 \) | \( PageRank > 0.3 \)
  - True | False | True | False

- \( BM25 > 0.5 \) | \( LM > 0.1 \) | \( PageRank > 0.3 \)
  - \( r = 0.2, NDCG = 0.6 \)
  - \( r = 0.12, NDCG = 0.5 \)
  - \( r = 0.18, NDCG = 0.7 \)
What if we have thousands of features?

• Is there any way I can do better?
  – Optimizing the metrics automatically!

Where to find those tree structures?

How to determine those $\alpha$s?
Rethink the task

• Given: (query, document) pairs represented by a set of relevance estimators, a.k.a., features

\[ \mathcal{f}(q, \{d_i\}_{i=1}^D) \rightarrow \text{ordered } \{d_i\}_{i=1}^D \]

• Needed: a way of combining the estimators

<table>
<thead>
<tr>
<th>DocID</th>
<th>BM25</th>
<th>LM</th>
<th>PageRank</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>1.6</td>
<td>1.1</td>
<td>0.9</td>
<td>0</td>
</tr>
<tr>
<td>0002</td>
<td>2.7</td>
<td>1.9</td>
<td>0.2</td>
<td>1</td>
</tr>
</tbody>
</table>

• Criterion: optimize IR metrics
  – P@k, MAP, NDCG, etc.

Key!
Machine Learning

- Input: \( \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\} \), where \( X_i \in R^N, Y_i \in R^M \)
- Objective function: \( O(Y', Y) \)
- Output: \( f(X) \to Y \), such that \( f = \text{argmax}_{f' \subset F} O(f'(X), Y) \)

\[ O(Y', Y) = \delta(Y' = Y) \]

Classification

\[ O(Y', Y) = -||Y' - Y|| \]

Regression

NOTE: We will only talk about supervised learning.
Learning to Rank

- General solution in optimization framework
  - Input: \(\{(q_i, d_1, y_1), (q_i, d_2, y_2), \ldots, (q_i, d_n, y_n)\}\), where \(d_n \in \mathbb{R}^N\), \(y_i \in \{0, \ldots, L\}\)
  - Objective: \(O = \{P@k, \text{MAP, NDCG}\}\)
  - Output: \(f(q, d) \rightarrow Y\), s.t., \(f = \arg\max_{f' \subset F} O(f'(q, d), Y)\)

<table>
<thead>
<tr>
<th>DocID</th>
<th>BM25</th>
<th>LM</th>
<th>PageRank</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>1.6</td>
<td>1.1</td>
<td>0.9</td>
<td>0</td>
</tr>
<tr>
<td>0002</td>
<td>2.7</td>
<td>1.9</td>
<td>0.2</td>
<td>1</td>
</tr>
</tbody>
</table>
Challenge: how to optimize?

- Evaluation metric recap
  - Average Precision
    \[ \text{AveP} = \frac{\sum_{k=1}^{n} (P(k) \times \text{rel}(k))}{\text{number of relevant documents}} \]
  - DCG
    \[ \text{DCG}_p = \text{rel}_1 + \sum_{i=2}^{p} \frac{\text{rel}_i}{\log_2 i} \]
- Order is essential!
  - \( f \rightarrow \text{order} \rightarrow \text{metric} \)

Not continuous with respect to \( f(X) \)!
Approximating the objective function!

• Pointwise
  – Fit the relevance labels individually
• Pairwise
  – Fit the relative orders
• Listwise
  – Fit the whole order
Pointwise Learning to Rank

• Ideally perfect relevance prediction leads to perfect ranking
  – $f \rightarrow \text{score} \rightarrow \text{order} \rightarrow \text{metric}$

• Reducing ranking problem to
  – Regression
    • $O(f(Q,D), Y) = -\sum_i \|f(q_i, d_i) - y_i\|$
    • Subset Ranking using Regression, D.Cossock and T.Zhang, COLT 2006

  – (multi-)Classification
    • $O(f(Q,D), Y) = \sum_i \delta(f(q_i, d_i) = y_i)$
    • Ranking with Large Margin Principles, A. Shashua and A. Levin, NIPS 2002
Subset Ranking using Regression
D.Cossock and T.Zhang, COLT 2006

• Fit relevance labels via regression
  \[ \hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \left[ \sum_{j=1}^{m} (f(x_{i,j}, S_i) - y_{i,j})^2 \right] \]
  \[ - \text{Emphasize more on relevant documents} \]
  \[ \sum_{j=1}^{m} w(x_j, S)(f(x_j, S) - y_j)^2 + u \sup_{j} w'(x_j, S)(f(x_j, S) - \delta(x_j, S))^2 \]

Weights on each document
Most positive document

http://en.wikipedia.org/wiki/Regression_analysis
Ranking with Large Margin Principles
A. Shashua and A. Levin, NIPS 2002

• Goal: correctly placing the documents in the corresponding category and maximize the margin
Ranking with Large Margin Principles
A. Shashua and A. Levin, NIPS 2002

- Ranking lost is consistently decreasing with more training data
What did we learn

• Machine learning helps!
  – Derive something optimizable
  – More efficient and guided
Deficiency

• Cannot directly optimize IR metrics
  – $(0 \rightarrow 1, 2 \rightarrow 0)$  \hspace{1cm}  $(0 \rightarrow -2, 2 \rightarrow 4)$

• Position of documents are ignored
  – Penalty on documents at higher positions should be larger

• Favor the queries with more documents
Pairwise Learning to Rank

• Ideally perfect partial order leads to perfect ranking
  \[- f \rightarrow \textbf{partial order} \rightarrow \textbf{order} \rightarrow \textbf{metric}\]

• Ordinal regression
  \[- O(f(Q,D), Y) = \sum_{i \neq j} \delta(y_i > y_j)\delta(f(q_i, d_i) < f(q_i, d_i))\]
  • Relative ordering between different documents is significant
  • E.g., \((0 \rightarrow -2, 2 \rightarrow 4)\) is better than \((0 \rightarrow 1, 2 \rightarrow 0)\)

• Large body of research
• Minimizing the number of mis-ordered pairs

\[ y_1 > y_2, y_2 > y_3, y_1 > y_4 \]

linear combination of features

\[ f(q, d) = w^T X_{q,d} \]

Keep the relative orders

\[
\begin{align*}
\text{minimize:} & \quad V(\tilde{w}, \tilde{\xi}) = \frac{1}{2} \tilde{w} \cdot \tilde{w} + C \sum \xi_{i,j,k} \\
\text{subject to:} & \quad \forall (d_i, d_j) \in r_1^*: \tilde{w} \Phi(q_1, d_i) \geq \tilde{w} \Phi(q_1, d_j) + 1 - \xi_{i,j,1} \\
& \quad \ldots \\
& \quad \forall (d_i, d_j) \in r_n^*: \tilde{w} \Phi(q_n, d_i) \geq \tilde{w} \Phi(q_n, d_j) + 1 - \xi_{i,j,1} \\
& \quad \forall i \forall j \forall k: \xi_{i,j,k} \geq 0
\end{align*}
\]
Optimizing Search Engines using Clickthrough Data
Thorsten Joachims, KDD’02

• How to use it?
  – $f \rightarrow \text{score} \rightarrow \text{order}$
What did it learn from the data?

- Linear correlations

<table>
<thead>
<tr>
<th>weight</th>
<th>feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.60</td>
<td>query_abstract_cosine</td>
</tr>
<tr>
<td>0.48</td>
<td>top10_google</td>
</tr>
<tr>
<td>0.24</td>
<td>query_url_cosine</td>
</tr>
<tr>
<td>0.24</td>
<td>top1count_1</td>
</tr>
<tr>
<td>0.24</td>
<td>top10_msnsearch</td>
</tr>
<tr>
<td>0.22</td>
<td>host_citeseer</td>
</tr>
<tr>
<td>0.21</td>
<td>domain_nec</td>
</tr>
<tr>
<td>0.19</td>
<td>top10count_3</td>
</tr>
<tr>
<td>0.17</td>
<td>top1_google</td>
</tr>
<tr>
<td>0.17</td>
<td>country_de</td>
</tr>
<tr>
<td>...</td>
<td>abstract_contains_home</td>
</tr>
<tr>
<td>0.16</td>
<td>top1_hotbot</td>
</tr>
<tr>
<td>...</td>
<td>domain_name_in_query</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>-0.13</td>
<td>domain_tu-bs</td>
</tr>
<tr>
<td>-0.15</td>
<td>country_fi</td>
</tr>
<tr>
<td>-0.16</td>
<td>top50count_4</td>
</tr>
<tr>
<td>-0.17</td>
<td>url_length</td>
</tr>
<tr>
<td>-0.32</td>
<td>top10count_0</td>
</tr>
<tr>
<td>-0.38</td>
<td>top1count_0</td>
</tr>
</tbody>
</table>

Positive correlated features

Negative correlated features
Optimizing Search Engines using Clickthrough Data
Thorsten Joachims, KDD’02

• How good is it?
  – Test on real system
An Efficient Boosting Algorithm for Combining Preferences


• Smooth the loss on mis-ordered pairs

\[- \sum_{y_i > y_j} Pr(d_i, d_j) \exp[f(q, d_j) - f(q, d_i)]\]

- hinge loss (RankSVM)
- 0/1 loss
- square loss
- exponential loss

from Pattern Recognition and Machine Learning, P337
An Efficient Boosting Algorithm for Combining Preferences

• RankBoost: optimize via boosting
  – Vote by a committee

BM25
\( \{w_n^{(1)}\} \)

PageRank
\( \{w_n^{(2)}\} \)

Cosine
\( \{w_n^{(M)}\} \)

Updating
\( \Pr(d_i, d_j) \)

Credibility of each committee member (ranking feature)

\[
Y_M(x) = \text{sign}\left( \sum_{m=1}^{M} \alpha_m y_m(x) \right)
\]
An Efficient Boosting Algorithm for Combining Preferences

• How good is it?
A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgments
Zheng et al. SIRIG’07

• Non-linear ensemble of features
  – Object: $\sum_{y_i > y_j} (\max\{0, f(q, d_j) - f(q, d_i)\})^2$
  – Gradient descent boosting tree
    • Boosting tree
      – Using regression tree to minimize the residuals
      – $r^t(q, d, y) = O^t(q, d, y) - f^{(t-1)}(q, d, y)$

BM25 > 0.5
  True
  False
  LM > 0.1
    True
    False
    r = 1.0
    r = 0.7
  PageRank > 0.3
    True
    False
    r = 0.4
    r = 0.1

True
False

CS@UVa
A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgments
Zheng et al. SIRIG’07

- Non-linear v.s. linear
  - Comparing with RankSVM

![DCG for GBRank, GBT, and RankSVM in 5-fold cross validation](chart.png)
Where do we get the relative orders

• Human annotations
  – Small scale, expensive to acquire

• Clickthroughs
  – Large amount, easy to acquire
What did we learn

• Predicting relative order
  – Getting closer to the nature of ranking

• Promising performance in practice
  – Pairwise preferences from click-throughs
Listwise Learning to Rank

• Can we directly optimize the ranking?
  – $f \rightarrow \text{order} \rightarrow \text{metric}$

• Tackle the challenge
  – Optimization without gradient
From RankNet to LambdaRank to LambdaMART: An Overview
Christopher J.C. Burges, 2010

- Minimizing mis-ordered pair => maximizing IR metrics?

Mis-ordered pairs: 6
AP: $\frac{5}{8}$
DCG: 1.333

Mis-ordered pairs: 4
AP: $\frac{5}{12}$
DCG: 0.931

*Position is crucial!*
From RankNet to LambdaRank to LambdaMART: An Overview
Christopher J.C. Burges, 2010

- Weight the mis-ordered pairs?
  - Some pairs are more important to be placed in the right order
  - Inject into object function
    \[ \sum y_i > y_j \Omega d_i, d_j \exp^{-w \Delta \Phi q_n}, d_i, d_j \]
  - Inject into gradient
    \[ \lambda_{ij} = \frac{\partial O_{approx}}{\partial \Delta O} \]

Gradient with respect to approximated objective, i.e., exponential loss on mis-ordered pairs
Change in original object, e.g., NDCG, if we switch the documents i and j, leaving the other documents unchanged
Depend on the ranking of document i, j in the whole list
From RankNet to LambdaRank to LambdaMART: An Overview
Christopher J.C. Burges, 2010

• Lambda functions
  – Gradient?
    • Yes, it meets the sufficient and necessary condition of being partial derivative
  – Lead to optimal solution of original problem?
    • Empirically
From RankNet to LambdaRank to LambdaMART: An Overview
Christopher J.C. Burges, 2010

- Evolution

<table>
<thead>
<tr>
<th></th>
<th>RankNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>Cross entropy over the pairs</td>
</tr>
<tr>
<td>Gradient</td>
<td>Gradient of cross entropy</td>
</tr>
<tr>
<td>Optimization</td>
<td>neural network</td>
</tr>
</tbody>
</table>

As we discussed in RankBoost
Optimize solely by gradient
Non-linear combination
From RankNet to LambdaRank to LambdaMART: An Overview
Christopher J.C. Burges, 2010

• A Lambda tree

```
<tree id="8" weight="0.1">
    <split>
        <feature> 811 </feature>
        <threshold> 5.0 </threshold>
        <split pos="left">
            <feature> 33 </feature>
            <threshold> 20.0 </threshold>
            <split pos="left">
                <feature> 589 </feature>
                <threshold> 43493.125 </threshold>
                <split pos="left">
                    <feature> 1094 </feature>
                    <threshold> 302.73438 </threshold>
                    <split pos="left">
                        <feature> 108 </feature>
                        <threshold> 9881.824 </threshold>
                        <split pos="left">
                            <output> -0.66917753 </output>
                        </split>
                        <split pos="right">
                            <feature> 151 </feature>
                            <threshold> 9072276.0 </threshold>
                        </split>
                    </split>
                </split>
            </split>
        </split>
    </split>
</tree>
```
A Support Vector Machine for Optimizing Average Precision
Yisong Yue, et al., SIGIR’07

\textbf{RankSVM}

\begin{itemize}
  \item Minimizing the pairwise loss
\end{itemize}

\[\text{minimize: } V(\vec{w}, \xi) = \frac{1}{2} \vec{w} \cdot \vec{w} + C \sum \xi_{i,j,k}\]

\[\text{subject to: }\]
\[\forall (d_i, d_j) \in r^*_1 : \vec{w} \Phi(q_1, d_i) \geq \vec{w} \Phi(q_1, d_j) + 1 - \xi_{i,j,1}\]
\[\text{...}\]
\[\forall (d_i, d_j) \in r^*_n : \vec{w} \Phi(q_n, d_i) \geq \vec{w} \Phi(q_n, d_j) + 1 - \xi_{i,j,n}\]
\[\forall i \forall j \forall k : \xi_{i,j,k} \geq 0\]

Loss defined on the number of mis-ordered document pairs

\textbf{SVM-MAP}

\begin{itemize}
  \item Minimizing the structural loss
\end{itemize}

\[\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \| \mathbf{w} \|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i\]

\[\text{s.t. } \forall i, \forall y \in Y \setminus y_i : \]
\[\mathbf{w}^T \Psi(x_i, y) \geq \mathbf{w}^T \Psi(x_i, y) + \Delta(y_i, y) - \xi_i\]

MAP difference

Loss defined on the quality of the whole list of ordered documents
A Support Vector Machine for Optimizing Average Precision
Yisong Yue, et al., SIGIR’07

- Max margin principle
  - Push the ground-truth far away from any mistakes one might make
  - Finding the most likely violated constraints
A Support Vector Machine for Optimizing Average Precision
Yisong Yue, et al., SIGIR’07

- Finding the most violated constraints
  - MAP is invariant to permutation of (ir)relevant documents
  - Maximize MAP over a series of swaps between relevant and irrelevant documents

\[
\arg\max_{y \in \mathcal{Y}} \Delta(y_i, y) + w^T \Psi(x_i, y)
\]

Right-hand side of constraints

Greedy solution

Start from the reverse order of ideal ranking
A Support Vector Machine for Optimizing Average Precision
Yisong Yue, et al., SIGIR’07

• Experiment results

<table>
<thead>
<tr>
<th>Model</th>
<th>TREC 9</th>
<th>TREC 10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
<td>W/L</td>
</tr>
<tr>
<td>SVM$^\Delta_{map}$</td>
<td>0.290</td>
<td>–</td>
</tr>
<tr>
<td>SVM$^\Delta_{roc}$</td>
<td>0.282</td>
<td>29/21</td>
</tr>
<tr>
<td>SVM$_{acc}$</td>
<td>0.213</td>
<td>49/1 **</td>
</tr>
<tr>
<td>SVM$_{acc2}$</td>
<td>0.270</td>
<td>34/16 **</td>
</tr>
<tr>
<td>SVM$_{acc3}$</td>
<td>0.133</td>
<td>50/0 **</td>
</tr>
<tr>
<td>SVM$_{acc4}$</td>
<td>0.233</td>
<td>47/3 **</td>
</tr>
</tbody>
</table>
Other listwise solutions

• Soften the metrics to make them differentiable
  – Michael Taylor et al., SoftRank: optimizing non-smooth rank metrics, WSDM'08

• Minimize a loss function defined on permutations
  – Zhe Cao et al., Learning to rank: from pairwise approach to listwise approach, ICML'07
What did we learn

• Taking a list of documents as a whole
  – Positions are visible for the learning algorithm
  – Directly optimizing the target metric

• Limitation
  – The search space is huge!
Summary

• Learning to rank
  – Automatic combination of ranking features for optimizing IR evaluation metrics

• Approaches
  – Pointwise
    • Fit the relevance labels individually
  – Pairwise
    • Fit the relative orders
  – Listwise
    • Fit the whole order
Experimental Comparisons

- My experiments
  - 1.2k queries, 45.5K documents with 1890 features
  - 800 queries for training, 400 queries for testing

<table>
<thead>
<tr>
<th>Model</th>
<th>MAP</th>
<th>P@1</th>
<th>ERR</th>
<th>MRR</th>
<th>NDCG@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ListNET</td>
<td>0.2863</td>
<td>0.2074</td>
<td>0.1661</td>
<td>0.3714</td>
<td>0.2949</td>
</tr>
<tr>
<td>LambdaMART</td>
<td>0.4644</td>
<td>0.4630</td>
<td>0.2654</td>
<td>0.6105</td>
<td>0.5236</td>
</tr>
<tr>
<td>RankNET</td>
<td>0.3005</td>
<td>0.2222</td>
<td>0.1873</td>
<td>0.3816</td>
<td>0.3386</td>
</tr>
<tr>
<td>RankBoost</td>
<td>0.4548</td>
<td>0.4370</td>
<td>0.2463</td>
<td>0.5829</td>
<td>0.4866</td>
</tr>
<tr>
<td>RankSVM</td>
<td>0.3507</td>
<td>0.2370</td>
<td>0.1895</td>
<td>0.4154</td>
<td>0.3585</td>
</tr>
<tr>
<td>AdaRank</td>
<td>0.4321</td>
<td>0.4111</td>
<td>0.2307</td>
<td>0.5482</td>
<td>0.4421</td>
</tr>
<tr>
<td>pLogistic</td>
<td>0.4519</td>
<td>0.3926</td>
<td>0.2489</td>
<td>0.5535</td>
<td>0.4945</td>
</tr>
<tr>
<td>Logistic</td>
<td>0.4348</td>
<td>0.3778</td>
<td>0.2410</td>
<td>0.5526</td>
<td>0.4762</td>
</tr>
</tbody>
</table>
Connection with Traditional IR

• People have foreseen this topic long time ago
  – Recall: probabilistic ranking principle
Conditional models for $P(R=1 \mid Q,D)$

- Basic idea: relevance depends on how well a query matches a document
  - $P(R=1 \mid Q,D)=g(\text{Rep}(Q,D) \mid \theta)$ ← a functional form
  - $\text{Rep}(Q,D)$: feature representation of query-doc pair
    - E.g., #matched terms, highest IDF of a matched term, docLen
  - Using training data (with known relevance judgments) to estimate parameter $\theta$
  - Apply the model to rank new documents

- Special case: logistic regression
Broader Notion of Relevance
Future

- Tighter bounds
- Faster solution
- Larger scale
- Wider application scenario
Resources

• Books

• Helpful pages

• Packages

• Data sets
References


References


Ranking with Large Margin Principles
A. Shashua and A. Levin, NIPS 2002

Maximizing the sum of margins

\[
\min_{w, a_j, b_j} \sum_{j=1}^{k-1} (a_j - b_j) + C \sum_i \sum_j \left( \varepsilon_i^j + \varepsilon_i^{j+1} \right)
\]
subject to
\[
\begin{align*}
  a_j &\leq b_j, \\
  b_j &\leq a_{j+1}, \quad j = 1, \ldots, k - 2
\end{align*}
\]
\[
\begin{align*}
  w \cdot x_i^j &\leq a_j + \varepsilon_i^j, \quad b_j - \varepsilon_i^{j+1} \leq w \cdot x_i^{j+1}
\end{align*}
\]
\[
\begin{align*}
  w \cdot w &\leq 1, \quad \varepsilon_i^j \geq 0, \varepsilon_i^{j+1} \geq 0
\end{align*}
\]
AdaRank: a boosting algorithm for information retrieval
Jun Xu & Hang Li, SIGIR’07

• Loss defined by IR metrics
  \[
  \sum_{q \in Q} Pr(q) \exp[-O(q)]
  \]

  – Optimizing by boosting

  Target metrics: MAP, NDCG, MRR

BM25
\[\{w_n^{(1)}\}\]

PageRank
\[\{w_n^{(2)}\}\]

Cosine
\[\{w_n^{(M)}\}\]

Updating \(Pr(q)\)

\[y_1(x)\] \[y_2(x)\] \[\ldots\] \[y_M(x)\]

Credibility of each committee member (ranking feature)

\[
Y_M(x) = \text{sig} \left( \sum_{m=1}^{M} f_m^M(x) \right)
\]

from Pattern Recognition and Machine Learning, P658
Analysis of the Approaches

• What are they really optimizing?
  – Relation with IR metrics
Pointwise Approaches

- **Regression based**

\[
1 - \text{NDCG}(f) \leq \frac{1}{Z_m} \left( \frac{2 \sum_{j=1}^{m} \eta_j^e}{\sum_{j=1}^{m} (f(x_j) - y_j)^\beta} \right)^{1/\alpha}
\]

Discount coefficients in DCG

Regression loss

- **Classification based**

\[
1 - \text{NDCG}(f) \leq \frac{15}{Z_m} \sqrt{2 \left( \sum_{j=1}^{m} \eta_j^2 - m \prod_{j=1}^{m} \eta_j^2 \right)} \cdot \sum_{j=1}^{m} I_{\{y_j \neq f(x_j)\}}
\]

Discount coefficients in DCG

Classification loss
Pointwise Approach

- Although it seems the loss functions can bound \((1-\text{NDCG})\), the constants before the losses seem too large.

\[
\begin{align*}
Z_m & \approx 21.4 \\
\text{DCG}(f) & \approx 21.4 \\
|1 - \text{NDCG}(f)| & = 0 \\
\frac{15}{Z_m} \sqrt{2 \left( \sum_{j=1}^{m} \left( \frac{1}{\log(j+1)} \right)^2 - m \sum_{j=1}^{m} \left( \frac{1}{\log(j+1)} \right) \right) \cdot \sum_{j=1}^{m} I_{\{y_j \neq f(x_j)\}} \approx 1.15 > 1}
\end{align*}
\]

From Tie-Yan Liu @ WWW 2009 Tutorial on Learning to Rank
Pairwise Approach  

• Unified loss vs. (1-NDCG) Discount coefficients in DCG  
  – When $\beta_t = \frac{G(t)\eta(t)}{Z_m}$, $L(f)$ is a tight bound of (1-NDCG).  

• Surrogate function of Unified loss  
  – After introducing weights $\beta_t$, loss functions in Ranking SVM, RankBoost, RankNet are Cost-sensitive Pairwise Comparison surrogate functions, and thus are consistent with and are upper bounds of the unified loss.  
  – Consequently, they also upper bound (1-NDCG).
Listwise Approaches

• No general analysis
  – Method dependent
  – Directness and consistency