WebSci and Learning to Rank for IR

Ernesto Diaz-Aviles

L3S Research Center. Hannover, Germany
diaz@L3S.de
Motivation: Information Explosion

Today???
Motivation: Information Explosion

How can an Information Retrieval System help users to meet their particular information needs and preferences?
How can an Information Retrieval System help users to meet their particular information needs and preferences?

In practice this problem can be cast as a: **Ranking Problem**

- **Goal:** define an ordering among items
- **i.e.,** Web pages, documents, news articles, Web sites, CDs, books, or movies.
- **Ranks relevant ones in higher positions of the retrieved list.**
Ranking
Learning to Rank

- Classical and Most popular IR model for document retrieval: Vector Space Model
- In recent years: Supervised learning-based methods have been proposed to automatically learn an effective ranking model based on training data and then applied to unseen test data
- This task is referred to as Learning To Rank for IR
Learning to Rank Model

Training Data

Labels: multiple-level ratings for example.

Model $f(q, d, w)$

MicrosoftResearch@Tutorial at WWW 2008
### Summary of notations

<table>
<thead>
<tr>
<th>Notations</th>
<th>Explanations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q = {q_1, \cdots, q_{</td>
<td>Q</td>
</tr>
<tr>
<td>$q_i \in Q$</td>
<td>Query</td>
</tr>
<tr>
<td>$D = {d_1, \cdots, d_{</td>
<td>D</td>
</tr>
<tr>
<td>$d_j \in D$</td>
<td>Document</td>
</tr>
<tr>
<td>$Y = {y_1, \cdots, y_{</td>
<td>Y</td>
</tr>
<tr>
<td>$y_{ij} \in Y$</td>
<td>Relevance judgement of query-document pair $(q_i, d_j) \in Q \times D$</td>
</tr>
<tr>
<td>$\phi(q_i, d_j)$</td>
<td>Feature vector w.r.t. $(q_i, d_j)$</td>
</tr>
<tr>
<td>$\phi_k(q_i, d_j)$</td>
<td>$k^{th}$ dimension of $\phi(q_i, d_j)$</td>
</tr>
<tr>
<td>$T = {(q_i, d_j), \phi(q_i, d_j), y_{ij}}$</td>
<td>Training set</td>
</tr>
</tbody>
</table>

1 \leq i \leq |Q|  
1 \leq j \leq |D|
LETOR is a package of benchmark data sets for LEarning TO Rank, released by Microsoft Research Asia.

<table>
<thead>
<tr>
<th>Feature type</th>
<th>Feature name</th>
<th>References</th>
<th>Number of Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-level Content</td>
<td>tf</td>
<td>[Baeza-Yates2008]</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>idf</td>
<td>[Baeza-Yates2008]</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>dl</td>
<td>[Baeza-Yates2008]</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>tfidf</td>
<td>[Baeza-Yates2008]</td>
<td>4</td>
</tr>
<tr>
<td>High-level Content</td>
<td>BM25</td>
<td>[BM25]</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>LMIR</td>
<td>[LMIR]</td>
<td>9</td>
</tr>
<tr>
<td>Hyperlink</td>
<td>PageRank</td>
<td>[PageRank]</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Topical PageRank</td>
<td>[topicalPageRank·HTS]</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>HITS</td>
<td>[HITS]</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Topical HITS</td>
<td>[topicalPageRank·HTS]</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>HostRank</td>
<td>[HostRank]</td>
<td>1</td>
</tr>
<tr>
<td>Hybrid</td>
<td>Hyperlink-based relevance propagation</td>
<td>[relevancePropagation]</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Sitemap-based relevance propagation</td>
<td>[siteMapPropagation]</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>total:</td>
<td></td>
<td>44</td>
</tr>
</tbody>
</table>
Fig. 1 Sample data from LETOR

feature is dependent on both the query and the document, D means that the feature only depends on the document, and Q means that the feature only depends on the query. Here we would like to point out that a linear ranking function cannot make use of the class-Q features, since these features are the same for all the documents under a query.

Table 2: Learning Features for the “Gov” Corpus

<table>
<thead>
<tr>
<th>ID</th>
<th>Feature Description</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d)$ in body</td>
<td>Q-D</td>
</tr>
<tr>
<td>2</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d)$ in anchor</td>
<td>Q-D</td>
</tr>
<tr>
<td>3</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d)$ in title</td>
<td>Q-D</td>
</tr>
<tr>
<td>4</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d)$ in URL</td>
<td>Q-D</td>
</tr>
<tr>
<td>5</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d)$ in whole document</td>
<td>Q-D</td>
</tr>
<tr>
<td>6</td>
<td>$\sum_{q_i \in q} id f(q_i)$ in body</td>
<td>Q</td>
</tr>
<tr>
<td>7</td>
<td>$\sum_{q_i \in q} id f(q_i)$ in anchor</td>
<td>Q</td>
</tr>
<tr>
<td>8</td>
<td>$\sum_{q_i \in q} id f(q_i)$ in title</td>
<td>Q</td>
</tr>
<tr>
<td>9</td>
<td>$\sum_{q_i \in q} id f(q_i)$ in URL</td>
<td>Q</td>
</tr>
<tr>
<td>10</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d) \cdot id f(q_i)$ in body</td>
<td>Q-D</td>
</tr>
<tr>
<td>11</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d) \cdot id f(q_i)$ in anchor</td>
<td>Q-D</td>
</tr>
<tr>
<td>12</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d) \cdot id f(q_i)$ in title</td>
<td>Q-D</td>
</tr>
<tr>
<td>13</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d) \cdot id f(q_i)$ in URL</td>
<td>Q-D</td>
</tr>
<tr>
<td>14</td>
<td>$\sum_{q_i \in q \cap d} (q_i, d) \cdot id f(q_i)$ in whole document</td>
<td>Q-D</td>
</tr>
<tr>
<td>15</td>
<td>$</td>
<td>d</td>
</tr>
<tr>
<td>16</td>
<td>$</td>
<td>d</td>
</tr>
<tr>
<td>17</td>
<td>$</td>
<td>d</td>
</tr>
<tr>
<td>18</td>
<td>$</td>
<td>d</td>
</tr>
<tr>
<td>19</td>
<td>$</td>
<td>d</td>
</tr>
</tbody>
</table>
Ranking Model

Ranking Model: Real valued function of features

\[ f(q, d) = \vec{w} \cdot \phi(q, d) \]  

- \( \vec{w} \) denotes a weight vector
- \( \phi(q_i, d_j) \) & Feature vector w.r.t. \((q_i, d_j)\)
- Ranking associates a score to each of the documents \(d_j\) as their degree of relevance with respect to query \(q_i\) using \(f(q_i, d_j)\)
- Sort the documents based on their scores
IR Evaluation

• Objective
  – Evaluate the effectiveness of a ranking model

• A standard test set, containing
  – A large number of queries, their associated documents, and the label (relevance judgments) of these documents.

• A measure
  – Evaluating the effectiveness of a ranking model for a particular query.
  – Averaging the measure over the entire test set to represent the expected effectiveness of the model.
Widely-used Judgments

• Binary judgment
  – Relevant vs. Irrelevant

• Multi-level ratings
  – Perfect > Excellent > Good > Fair > Bad

• Pairwise preferences
  – Document $A$ is more relevant than document $B$ with respect to query $q$
Evaluation Measures

• MAP (Mean Average Precision)
• NDCG (Normalized Discounted Cumulative Gain)
• MRR (Mean Reciprocal Rank)
  – For query $q_i$, rank position of the first relevant document: $r_i$
  – MRR: average of $1/r_i$ over all queries
• WTA (Winners Take All)
  – If top ranked document is relevant: 1; otherwise 0
  – average over all queries
• ......
MAP

- Precision at position $n$
  \[ P \@ n = \frac{\#\{\text{relevant documents in top } n \text{ results}\}}{n} \]

- Average precision
  \[ AP = \frac{\sum_{n} P \@ n \cdot I\{\text{document } n \text{ is relevant}\}}{\#\{\text{relevant documents}\}} \]

\[ AP = \frac{1}{3} \cdot \left( \frac{1}{1} + \frac{2}{3} + \frac{3}{5} \right) \approx 0.76 \]

- MAP: averaged over all queries in the test set
NDCG

• NDCG at position $n$:

$$N(n) = Z_n \sum_{j=1}^{n} \frac{(2^{r(j)} - 1)}{\log(1 + j)}$$

• NDCG averaged for all queries in test set
## G: Gain

<table>
<thead>
<tr>
<th>Relevance Rating</th>
<th>Value (Gain)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect</td>
<td>$31=2^5-1$</td>
</tr>
<tr>
<td>Excellent</td>
<td>$15=2^4-1$</td>
</tr>
<tr>
<td>Good</td>
<td>$7=2^3-1$</td>
</tr>
<tr>
<td>Fair</td>
<td>$3=2^2-1$</td>
</tr>
<tr>
<td>Bad</td>
<td>$0=2^0-1$</td>
</tr>
</tbody>
</table>
**CG: Cumulative G**

*If each rank position is treated equal*

<table>
<thead>
<tr>
<th>URL</th>
<th>Gain</th>
<th>Cumulative Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1 <a href="http://abc.go.com/">http://abc.go.com/</a></td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td>#2 <a href="http://www.abcteach.com/">http://www.abcteach.com/</a></td>
<td>3</td>
<td>34 = 31 + 3</td>
</tr>
<tr>
<td>#3 <a href="http://abcnews.go.com/sections/scitech/">http://abcnews.go.com/sections/scitech/</a></td>
<td>15</td>
<td>49 = 31 + 3 + 15</td>
</tr>
<tr>
<td>#4 <a href="http://www.abc.net.au/">http://www.abc.net.au/</a></td>
<td>15</td>
<td>64 = 31 + 3 + 15 + 15</td>
</tr>
<tr>
<td>#5 <a href="http://abcnews.go.com/">http://abcnews.go.com/</a></td>
<td>15</td>
<td>79 = 31 + 3 + 15 + 15 + 15</td>
</tr>
<tr>
<td>#6 ...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
DCG: Discounted CG

- Discounting factor: $\log(2) / (\log(1+\text{rank}))$

<table>
<thead>
<tr>
<th>URL</th>
<th>Gain</th>
<th>Discounted Cumulative Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1 <a href="http://abc.go.com/">http://abc.go.com/</a></td>
<td>31</td>
<td>$31 = 31 \times 1$</td>
</tr>
<tr>
<td>#2 <a href="http://www.abcteach.com/">http://www.abcteach.com/</a></td>
<td>3</td>
<td>$32.9 = 31 + 3 \times 0.63$</td>
</tr>
<tr>
<td>#3 <a href="http://abcnews.go.com/sections/scitech/">http://abcnews.go.com/sections/scitech/</a></td>
<td>15</td>
<td>$40.4 = 32.9 + 15 \times 0.50$</td>
</tr>
<tr>
<td>#4 <a href="http://www.abc.net.au/">http://www.abc.net.au/</a></td>
<td>15</td>
<td>$46.9 = 40.4 + 15 \times 0.43$</td>
</tr>
<tr>
<td>#5 <a href="http://abcnews.go.com/">http://abcnews.go.com/</a></td>
<td>15</td>
<td>$52.7 = 46.9 + 15 \times 0.39$</td>
</tr>
<tr>
<td>#6 ...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Ideal Ordering: Max DCG

• Sort the documents according to their labels

<table>
<thead>
<tr>
<th>URL</th>
<th>Gain</th>
<th>Max DCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1 <a href="http://abc.go.com/">http://abc.go.com/</a></td>
<td>31</td>
<td>31 = 31x1</td>
</tr>
<tr>
<td>#2 <a href="http://abcnews.go.com/sections/scitech/">http://abcnews.go.com/sections/scitech/</a></td>
<td>15</td>
<td>40.5 = 31 + 15x0.63</td>
</tr>
<tr>
<td>#3 <a href="http://www.abc.net.au/">http://www.abc.net.au/</a></td>
<td>15</td>
<td>48.0 = 40.5 + 15x0.50</td>
</tr>
<tr>
<td>#4 <a href="http://abcnews.go.com/">http://abcnews.go.com/</a></td>
<td>15</td>
<td>54.5 = 48.0 + 15x0.43</td>
</tr>
<tr>
<td>#5 <a href="http://www.abc.org/">http://www.abc.org/</a></td>
<td>15</td>
<td>60.4 = 54.5 + 15x0.39</td>
</tr>
<tr>
<td>#6 ...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
NDCG: Normalized DCG

- Normalized by the Max DCG

<table>
<thead>
<tr>
<th>#</th>
<th>URL</th>
<th>Gain</th>
<th>DCG</th>
<th>Max DCG</th>
<th>NDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td><a href="http://abc.go.com/">http://abc.go.com/</a></td>
<td>31</td>
<td>31</td>
<td>31</td>
<td>1 = 31/31</td>
</tr>
<tr>
<td>#2</td>
<td><a href="http://www.abcteach.com/">http://www.abcteach.com/</a></td>
<td>3</td>
<td>32.9</td>
<td>40.5</td>
<td>0.81 = 32.9/40.5</td>
</tr>
<tr>
<td>#3</td>
<td><a href="http://abcnews.go.com/sections/scitech/">http://abcnews.go.com/sections/scitech/</a></td>
<td>15</td>
<td>40.4</td>
<td>48.0</td>
<td>0.84 = 40.4/48.0</td>
</tr>
<tr>
<td>#4</td>
<td><a href="http://www.abc.net.au/">http://www.abc.net.au/</a></td>
<td>15</td>
<td>46.9</td>
<td>54.5</td>
<td>0.86 = 46.9/54.5</td>
</tr>
<tr>
<td>#5</td>
<td><a href="http://abcnews.go.com/">http://abcnews.go.com/</a></td>
<td>15</td>
<td>52.7</td>
<td>60.4</td>
<td>0.87 = 52.7/60.4</td>
</tr>
<tr>
<td>#6</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
More about Evaluation Measures

• Query-level
  – Averaged over all testing queries.
  – The measure is bounded, and every query contributes equally to the averaged measure.

• Position
  – Position discount is used here and there.

• Non-smooth
  – Computed based on ranks of documents.
  – With small changes to parameters of the ranking model, the scores will change smoothly, but the ranks will not change until one document’s score passes another.
## Conventional Ranking Models

<table>
<thead>
<tr>
<th>Category</th>
<th>Example Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Similarity-based models</td>
<td>Boolean model</td>
</tr>
<tr>
<td></td>
<td>Vector space model</td>
</tr>
<tr>
<td></td>
<td>Latent Semantic Indexing</td>
</tr>
<tr>
<td></td>
<td>......</td>
</tr>
<tr>
<td>Probabilistic models</td>
<td>BM25 model</td>
</tr>
<tr>
<td></td>
<td>Language model for IR</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Hyperlink-based models</td>
<td>HITS</td>
</tr>
<tr>
<td></td>
<td>PageRank</td>
</tr>
<tr>
<td></td>
<td>......</td>
</tr>
</tbody>
</table>
Discussions on Conventional Ranking Models

• For a particular model
  – Parameter tuning is usually difficult, especially when there are many parameters to tune.

• For comparison between two models
  – Given a test set, it is difficult to compare two models, one is over-tuned (over-fitting) while the other is not.

• For a collection of models
  – There are hundreds of models proposed in the literature.
  – It is non-trivial to combine them effectively.
Machine Learning Can Help

• Machine learning is an effective tool
  – To automatically tune parameters
  – To combine multiple evidences
  – To avoid over-fitting (regularization framework, structure risk minimization, etc.)
Learning to Rank for IR

Overview

Pointwise Approach
Pairwise Approach
Listwise Approach
Framework of Learning to Rank for IR

Labels: multiple-level ratings for example.

Learning System

Model $f(q,d,w)$

Ranking System

Training Data

Test data

4/20/2008

Tie-Yan Liu @ Tutorial at WWW 2008
Learning to Rank Algorithms

Query refinement (WWW 2008)

LambdaRank (NIPS 2006) Frank (SIGIR 2007)    MPRank (ICML 2007)
Discriminative model for IR (SIGIR 2004) SVM Structure (JMLR 2005)
GPRank (LR4IR 2007)    QBRank (NIPS 2007)    GBRank (SIGIR 2007)
Relational ranking (WWW 2008) Supervised Rank Aggregation (WWW 2007)
Categorization: Relation to Conventional Machine Learning

• With certain assumptions, learning to rank can be reduced to conventional learning problems
  – Regression
    • Assume labels to be scores
  – Classification
    • Assume labels to have no orders
  – Pairwise classification
    • Assume document pairs to be independent of each other
Categorization: Relation to Conventional Machine Learning

- Ranking in IR has unique properties
  - Query determines the logical structure of the ranking problem.
    - Loss function should be defined on ranked lists w.r.t. a query, since IR measures are computed at query level.
  - Relative order is important
    - No need to predict category, or value of f(x).
  - Position sensitive
    - Top-ranked objects are more important.
  - Rank based evaluation
    - Learning objective is non-smooth and non-differentiable.
Categorization: Basic Unit of Learning

• Pointwise
  – Input: single documents
  – Output: scores or class labels
• Pairwise
  – Input: document pairs
  – Output: partial order preference
• Listwise
  – Input: document collections
  – Output: ranked document List
## Categorization of Learning to Rank Algorithms

<table>
<thead>
<tr>
<th></th>
<th>Pointwise Approach</th>
<th>Pairwise Approach</th>
<th>Listwise Approach</th>
</tr>
</thead>
</table>
Learning to Rank for IR

Overview

Pointwise Approach

Pairwise Approach

Listwise Approach
Overview of Pointwise Approach

• Reduce ranking to regression or classification on single documents.

• Examples:
  – Discriminative model for IR.
  – McRank: learning to rank using multi-class classification and gradient boosting.
Discriminative Model for IR  
(R. Nallapati, SIGIR 2004)

- Features extracted for each document

<table>
<thead>
<tr>
<th>Feature</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\sum_{q_i \in Q \cap D} \log(c(q_i, D))$</td>
</tr>
<tr>
<td>2</td>
<td>$\sum_{i=1}^{n} \log(1 + \frac{c(q_i, D)}{</td>
</tr>
<tr>
<td>3</td>
<td>$\sum_{q_i \in Q \cap D} \log(idf(q_i))$</td>
</tr>
<tr>
<td>4</td>
<td>$\sum_{q_i \in Q \cap D} (\log(\frac{</td>
</tr>
<tr>
<td>5</td>
<td>$\sum_{i=1}^{n} \log(1 + \frac{c(q_i, D)}{</td>
</tr>
<tr>
<td>6</td>
<td>$\sum_{i=1}^{n} \log(1 + \frac{c(q_i, D)}{</td>
</tr>
</tbody>
</table>

- Treat relevant documents as positive examples, while irrelevant documents as negative examples.
Discriminative Model for IR

• Learning algorithms
  – Max Entropy
    \[ P(R|D, Q) = \frac{1}{Z(Q, D)} \exp(\sum_{i=1}^{n} \lambda_{i,R} f_i(D, Q)) \]
    \[ \log \frac{P(R|D, Q)}{P(\bar{R}|D, Q)} = \sum_{i=1}^{n} (\lambda_{i,R} - \lambda_{i,\bar{R}}) f_i(D, Q) \]
  – Support Vector Machines
    \[ g(R|D, Q) = w \cdot \phi(f(D, Q)) + b \]

• Experimental results
  – Comparable with LM
McRank
(P. Li, et al. NIPS 2007)

• Loss in terms of DCG is upper bounded by multi-class classification error

\[
DCG_g - DCG_\pi \leq 15\sqrt{2} \left( \sum_{i=1}^{n} c_{[i]}^2 - n \prod_{i=1}^{n} c_{[i]}^{2/n} \right)^{1/2} \left( \sum_{i=1}^{n} 1_{y_i \neq \hat{y}_i} \right)^{1/2}
\]

• Multi-class classification:  \( P(y_i = k) \)
• Multiple ordinal classification:  \( P(y_i \leq k) \)
• Regression:  \( \min |f(x_i) - 2^{y_i - 1}| \)
Different Reductions

• From classification to ranking
  – Learn a classifier for each category
  – Classifier outputs converted to probabilities using logistic function
  – Convert classification to ranking:

  \[ S_i = \sum_{k=0}^{K-1} \hat{p}_{i,k} k \]

• Experimental results
  – Multiple ordinal classification > Multi-class classification > Regression
Discussions

• Assumption
  – Relevance is absolute and query-independent.
  – E.g. documents associated with different queries are put into the same category, if their labels are all “perfect”.

• However, in practice, relevance is query-dependent
  – An irrelevant document for a popular query might have higher term frequency than a relevant document for a rare query.
  – If simply put documents associated with different queries together, the training process may be hurt.
Learning to Rank for IR

Overview
Pointwise Approach
Pairwise Approach
Listwise Approach
Overview of Pairwise Approach

• No longer assume absolute relevance.
• Reduce ranking to classification on document pairs w.r.t. the same query.
• Examples
  – RankNet, FRank, RankBoost, Ranking SVM, etc.

\[
\begin{pmatrix}
q^{(i)} \\
(d_1^{(i)}, 5) \\
(d_2^{(i)}, 3) \\
\vdots \\
(d_n^{(i)}, 2)
\end{pmatrix}
\]

Transform

\[
\{ (d_1^{(i)}, d_2^{(i)}), (d_1^{(i)}, d_n^{(i)}), \ldots, (d_2^{(i)}, d_n^{(i)}) \}
\]

5  >  3  5  >  2  3  >  2

4/20/2008

Tie-Yan Liu @ Tutorial at WWW 2008
RankNet
(C. Burges, et al. ICML 2005)

• Target probability:
  – $\bar{P}_{ij}$ (1 means $i \succ j$; 0 mean $i \prec j$).

• Modeled probability:
  – $P_{ij} \equiv P(x_i \succ x_j) = \frac{\exp(o_{ij})}{1 + \exp(o_{ij})}$
  – Where $o_{ij} \equiv f(x_i) - f(x_j)$

• Cross entropy as the loss function
  – $C_{ij} \equiv C(o_{ij}) = -\bar{P}_{ij} \log P_{ij} - (1 - \bar{P}_{ij}) \log(1 - P_{ij})$
RankNet (2)

• Use Neural Network as model, and gradient descent as algorithm, to optimize the cross-entropy loss.

• Evaluate on single documents: output a relevance score for each document w.r.t. a new query.
Ranking SVM
(R. Herbrich, et al. ICANN 1999)

\[
\begin{align*}
\min_{w,\xi} & \quad \frac{1}{2} \| w \|^2 + C \xi_i \\
\text{s.t.} & \quad z_i \langle w, x_i^{(1)} - x_i^{(2)} \rangle \geq 1 - \xi_i \\
\xi_i & \geq 0
\end{align*}
\]

\[f(x; \hat{w}) = \langle \hat{w}, x \rangle\]

Use SVM to perform pairwise classification

Use SVM to perform binary classification on these instances, to learn model parameter \(w\)

\(x^{(1)} - x^{(2)}\) as positive instance of learning

Use \(w\) for testing

Use SVM to perform pairwise classification
Testing
Other Work in this Category

• Linear discriminant model for information retrieval (LDM), (J. Gao, H. Qi, et al. SIGIR 2005)
• Magnitude-preserving Ranking Algorithms (MPRank), (C. Cortes, M. Mohri, et al. ICML 2007)
Discussions

• Progress made as compared to pointwise approach
  – No longer assume absolute relevance

• However,
  – Unique properties of ranking in IR have not been fully modeled.
Problems with Pairwise Approach (1)

- Number of instance pairs varies according to query
  - Two queries in total
  - Same error in terms of pairwise classification $780/790 = 98.73\%$.
  - Different errors in terms of query level evaluation $99\%$ vs. $50\%$.

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document pairs of $q_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>correctly ranked</td>
<td>770</td>
<td>780</td>
</tr>
<tr>
<td>wrongly ranked</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>Accuracy</td>
<td>98.72%</td>
<td>100%</td>
</tr>
<tr>
<td>Document pairs of $q_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>correctly ranked</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>wrongly ranked</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>Accuracy</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>overall accuracy</td>
<td>Document level</td>
<td>98.73%</td>
</tr>
<tr>
<td>query level</td>
<td>99.36%</td>
<td>50%</td>
</tr>
</tbody>
</table>
Problems with Pairwise Approach (2)

• Negative effects of making errors on top positions
  – Same error in terms of pairwise classification
  – Different errors in terms of position-discounted evaluation.

p: \textit{perfect}, g: \textit{good}, b: \textit{bad}

Ideal: \quad p \ g \ g \ b \ b \ b \ b

\begin{align*}
\text{ranking 1: } & g \ p \ g \ b \ b \ b \ b \quad \text{one wrong pair} \quad \text{Worse} \\
\text{ranking 2: } & p \ g \ b \ g \ b \ b \ b \quad \text{one wrong pair} \quad \text{Better}
\end{align*}
As a Result, ...

- Users only see ranked list of documents, but not document pairs.
- It is not clear how pairwise loss correlates with IR evaluation measures, which are defined on query level.

Pairwise loss vs. (1-NDCG@5)
TREC Dataset
LambdaRank
(C. Burges, et al. NIPS 2006)

• Basic idea
  – IR evaluation measures are non-smooth and non-differentiable.
  – It is usually much easier to specify rules determining how to change rank order of documents than to construct a smooth loss function.

• Example
  – Two relevant documents $D_1$ and $D_2$;
  – WTA as the evaluation measure.

To achieve WTA=1

Want: $\left| \frac{\partial C}{\partial s_1} \right| \gg \left| \frac{\partial C}{\partial s_2} \right|$
Lambda Function, Instead of Loss Function

- Instead of explicitly defining the loss function, directly define the gradient:
  \[ \frac{\partial C}{\partial s_j} = -\lambda_j(s_1, l_1, \ldots, s_n, l_n) \]
  \( S: \) score; \( l: \) label.

- A lambda which was shown to effectively optimize NDCG.
  \[ \lambda = N \left( \frac{1}{1 + e^{s_i - s_j}} \right) (2^{l(i)} - 2^{l(j)}) \left( \log \left( \frac{1}{1+i} \right) - \log \left( \frac{1}{1+j} \right) \right) \]
Experimental Results

![Graph showing NDCG vs Truncation Level]

- LambdaRankTwoLayer
- RankNetTwoLayer

Web Data
Discussions

• Progress has been made as compared to pointwise and pairwise approaches

• Unsolved problems
  – Is the upper bound in SVM-MAP tight enough?
  – Is SoftNDCG really correlated with NDCG?
  – Is there any theoretical justification on the correlation between the implicit loss function used in LambdaRank and NDCG?

  – Multiple measures are not necessarily consistent with each other: which to optimize?
Learning to Rank for IR

Overview
Pointwise Approach
Pairwise Approach
Listwise Approach
Define Listwise Loss Function

• Indirectly optimizing IR evaluation measures
  – Consider properties of ranking for IR.

• Examples
  – RankCosine: use cosine similarity between score vectors as loss function
  – ListNet: use KL divergence between permutation probability distributions as loss function
  – ListMLE: use negative log likelihood of the ground truth permutation as loss function

• Theoretical analysis on listwise loss functions
Experimental Results

Pairwise (RankNet)  Listwise (ListNet)

Training Performance on TD2003 Dataset

More correlated!
Experimental Results
Experimental Results

TD2004 Dataset
Continuity, Differentiability, Convexity, Efficiency

<table>
<thead>
<tr>
<th>Loss</th>
<th>Continuity</th>
<th>Differentiability</th>
<th>Convexity</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cosine Loss (RankCosine)</td>
<td>√</td>
<td>√</td>
<td>x</td>
<td>O(n)</td>
</tr>
<tr>
<td>Cross-entropy loss (ListNet)</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>O(n·n!)</td>
</tr>
<tr>
<td>Likelihood loss (ListMLE)</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>O(n)</td>
</tr>
</tbody>
</table>
Discussion

• Progress has been made as compared to pointwise and pairwise approaches

• Likelihood loss seems to be one of the best listwise loss functions, according to the above discussions.

• In real ranking problems, the true loss should be cost-sensitive.
  – Need further investigations.
Stochastic Pairwise Descent (SPD)

Explain algorithm in whiteboard.
Summary

- Learning to Rank: Supervised Machine Learning Approach for Ranking

Huge impact: Web Search, E-Business, Government, etc.

Summary

- Learning to Rank: Supervised Machine Learning Approach for Ranking
- Approaches: Pointwise
Learning to Rank: Supervised Machine Learning Approach for Ranking

Approaches: Pointwise, Pairwise
Summary

- **Learning to Rank**: Supervised Machine Learning Approach for Ranking
- **Approaches**: Pointwise, Pairwise, and Listwise Approaches.
Summary

- Learning to Rank: Supervised Machine Learning Approach for Ranking
- Approaches: Pointwise, Pairwise, and Listwise Approaches.
- Huge impact: Web Search, E-Business, Government, etc.
Summary

- Learning to Rank: Supervised Machine Learning Approach for Ranking
- Approaches: Pointwise, Pairwise, and Liswise Approaches.
- Huge impact: Web Search, E-Business, Government, etc.
Thank you!

Ernesto Diaz-Aviles

diaz@L3S.de
More Info