A Survey on Learning to Rank

Kuang-Yu Chang
kuangyu@iis.sinica.edu.tw

Oct. 28, 2010
Outline

1. Introduction
2. Learning to Rank in IR
3. Categorization of Ranking Approaches
4. Pointwise approach
5. Pairwise approach
6. Listwise approach
7. Summary
Classification, regression and ranking.

Application examples:
- Recommendation Systems
- Information Retrieval
- Drug Discovery
- ...
Rank: natural representation of human preferences
1. Introduction
2. Learning to Rank in IR
3. Categorization of Ranking Approaches
4. Pointwise approach
5. Pairwise approach
6. Listwise approach
7. Summary
Learning to Rank in IR

query1
Learning to Rank in IR

query1

documents of
query1
Learning to Rank in IR

query1

documents of query1

query 2
Learning to Rank in IR

query1

documents of query1

query 2

documents of query 2
Learning to Rank in IR

query1
documents of query1

query 2
documents of query 2

new query
Learning to Rank in IR

Labels, refer to the judgments in IR evaluation.

How to sample the most appropriate training set?
How to extract the most useful features?
Rank-Based Evaluation Measures in IR

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

- Precision at position $n$ for query $q$:
  $$P_{@n} = \frac{\#\{\text{relevant documents in top } n \text{ results}\}}{n}$$

- Average precision for query $q$:
  $$AP = \frac{\sum_n P_{@n} \cdot \{\text{document } n \text{ is relevant}\}}{\#\{\text{relevant documents}\}}$$

  e.g. \[\begin{array}{cccccc}
  \text{GREEN} & \text{RED} & \text{GREEN} & \text{RED} & \text{GREEN} \\
  \end{array}\]  \Rightarrow  AP = \frac{1}{3} \cdot \left(\frac{1}{1} + \frac{2}{3} + \frac{3}{5}\right)

- MAP: averaged over all queries.
Normalized Discounted Cumulative Gain (NDCG)

- NDCG at position \( n \) for query \( q \):

\[
NDCG@n = \frac{\sum_{j=1}^{n} (2^{c(j)} - 1) / \log(1+j)}{Z_n}
\]

- Normalization
- Cumulating
- Gain
- Position discount

- Averaged over all queries.
Outline

1. Introduction
2. Learning to Rank in IR
   - Categorization of Ranking Approaches
     - Pointwise approach
     - Pairwise approach
     - Listwise approach
3. Summary
<table>
<thead>
<tr>
<th>Category</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pointwise Approach</td>
<td>Classification: McRank</td>
</tr>
<tr>
<td></td>
<td>Ordinal regression: Pranking</td>
</tr>
<tr>
<td>Pairwise Approach</td>
<td>RankSVM, RankBoost, RankNet, IRSVM, Multi-Hyperplane Ranker, Ordinal Regression by Extended Binary Classification</td>
</tr>
<tr>
<td>Listwise Approach</td>
<td>Listwise loss minimization: ListNet</td>
</tr>
<tr>
<td></td>
<td>Direct optimization of IR measure: LambdaRank</td>
</tr>
</tbody>
</table>
1. Introduction
2. Learning to Rank in IR
3. Categorization of Ranking Approaches
4. Pointwise approach
5. Pairwise approach
6. Listwise approach
7. Summary
Pointwise Approach

- Score each instance by some $f(x)$
- Quantize score to $r(x) = \arg\min\{f(x) < \theta_k\}$

\[
\begin{align*}
q &\leftrightarrow \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} \\
\{ (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \}
\end{align*}
\]
Multi-class classification is used to learn the ranking function.

Ranking is produced by combining the outputs of soft classification algorithm.

\[ f(x_j) = \sum_{k=0}^{K-1} \hat{P}_{j,k}, \text{ where } \hat{P}_{j,k} = P(y_j = k). \]

\[ \text{McRank}^1 \]

---

Pranking with Ranking

Project $x \rightarrow w \cdot x$

---

Pranking with Ranking

\[ \mathbf{w} \cdot \mathbf{x} \]
Pranking with Ranking

\{2, 3\}
Pranking with Ranking

Threshold adjustment
Pranking with Ranking

Perceptron update

\[ w^{t+1} \leftarrow w^t + x^t \]
Outline

1. Introduction
2. Learning to Rank in IR
3. Categorization of Ranking Approaches
4. Pointwise approach
5. Pairwise approach
6. Listwise approach
7. Summary
Pairwise approach

- Score each instance by some $f(x)$ such that $f$ matches the order of the ranks.

$$x_i > x_j \iff f(x_i) > f(x_j)$$
Assume $f$ is a linear ranking function $f(x) = \langle w, x \rangle$ such that

$$f(x_i) > f(x_j) \iff x_i > x_j$$
$$\langle w, x_i - x_j \rangle > 0 \iff x_i > x_j$$

We take any instance pair and their relation to create new training data set $S'$.

$$S' = \{ x_k^{(i)} - x_l^{(i)}, z_{k,l}^{(i)} \}, \text{ where } z_{k,l}^{(i)} = \begin{cases} +1, & \text{if } y_k^{(i)} > y_l^{(i)} \\ -1, & \text{if } y_l^{(i)} > y_k^{(i)} \end{cases}$$
Use SVM to perform pairwise classification. Hinge loss as the loss function:
\[ l(f; x_k, x_l, y_k, l) = (1 - y_k, l(f(x_k) - f(x_l)))^+ \]

\[
\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \sum_{k,l} \xi^{(i)}_{k,l}
\]

subject to \[ z_{k,l}^{(i)} \langle w, x_k^{(i)} - x_l^{(i)} \rangle \geq 1 - \xi_{k,l}^{(i)}, \]
\[ \xi_{k,l}^{(i)} \geq 0 \]

We utilize \( w^* \) to form a ranking function for ranking instance \( f(x) = \langle w^*, x \rangle \)
Use AdaBoost to perform pairwise classification. Exponential loss as the loss function:
\[ l(f; x_k, x_l, y_k, l) = \exp(-y_k, l(f(x_k) - f(x_l))) \]

Algorithm RankBoost
Given: initial distribution \( D \) over \( X \times X \).
Initialize: \( D_1 = D \).
For \( t = 1, \ldots, T \):
- Train weak learner using distribution \( D_t \).
- Get weak ranking \( h_t : X \rightarrow \mathbb{R} \).
- Choose \( \alpha_t \in \mathbb{R} \).
- Update: \( D_{t+1}(x_0, x_1) = \frac{D_t(x_0, x_1) \exp(\alpha_t(h_t(x_0) - h_t(x_1)))}{Z_t} \)
  where \( Z_t \) is a normalization factor (chosen so that \( D_{t+1} \) will be a distribution).

Output the final ranking: \( H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \)

---

The author convert the problem of intensity estimation to a ranking problem, which is modeled by the RankBoost\textsuperscript{5}.

Target probability:

\[ P_{k,l} = 1, \text{ if } y_{k,l} = 1 (i.e. x_k > x_l); \quad P_{k,l} = 0, \text{ otherwise}. \]

Modeled probability:

\[ P_{k,l}(f) = \frac{\exp(f(x_k) - f(x_l))}{1 + \exp(f(x_k) - f(x_l))} \]

Cross entropy as the loss function

\[ l(f; x_k, x_l, y_{k,l}) = -P_{k,l} \log P_{k,l}(f) - (1 - P_{k,l}) \log (1 - P_{k,l}(f)) \]

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

---

Introduce query-level normalization to the pairwise loss function for IR.

\[
\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \mu^{(i)} \sum_{k,l} \xi_{k,l}^{(i)}
\]

\(\mu^{(i)}\) is a query parameter to penalize the error on the queries with fewer instance pairs.

\[
\mu^{(i)} = \frac{\max \#\{\text{instance pairs associated with } q(j)\}}{\#\{\text{instance pairs associated with } q(i)\}}
\]

---

Multi-Hyperplane Ranker

- If we have $K$ categories, then will have $K(K - 1)/2$ pairwise preferences between two labels.
  - Learn a model $f_{k,l}$ for the pair of categories $k$ and $l$, using any previous method (for example, RankSVM).

- Rank Aggregation

$$f(x) \sum_{k,l} = \alpha_{k,l} f_{k,l}(x)$$

---

All the binary classification problems are solved jointly to obtain a single binary classifier $f$. $f$ is to use a thresholded model, $f(x, k) = g(x) - \theta_k$. As long as the threshold vector $\theta$ is ordered, the function $f$ is rank-monotonic.

---

The binary classifier $f(x, k) = \langle w, \phi(x) \rangle - \theta_k - b$ can be obtained by solving

$$\min_{w, b, \theta_k, \xi_i^{(k)}} \langle w, w \rangle + \frac{1}{\gamma^2} \langle \theta, \theta \rangle + \kappa \sum_{i=1}^{N} \sum_{k=1}^{K-1} W_i^{(k)} \xi_i^{(k)},$$

subject to

$$Y_i^{(k)}(f(x, k)) \geq 1 - \xi_i^{(k)},$$

$$\xi_i^{(k)} \geq 0, \text{ for } i = 1, \ldots, N \text{ and } k = 1, \ldots, K - 1.$$ 

Then, a ranking rule $r$ constructed from $f$

$$r(x) \equiv 1 + \sum_{k=1}^{K-1} \mathbb{1}[f(x, k) > 0].$$
Outline

1. Introduction
2. Learning to Rank in IR
3. Categorization of Ranking Approaches
4. Pointwise approach
5. Pairwise approach
6. Listwise approach
7. Summary
Listwise approach

- Try to match the list order within each query with respect to the criteria of interest.

- Two Major Branches of listwise ranking:
  1. *Listwise loss minimization*
     - Minimize a loss function defined on permutations, which is designed by considering the properties of ranking for IR.
  2. *Direct optimization of IR measures*
     - Try to optimize IR evaluation measures, or at least something correlated to the measures.
ListNet is similar to RankNet. The former uses document lists as instances while the latter uses document pairs as instances; the former utilizes a listwise loss function while the latter utilizes a pairwise loss function.

Loss function = KL-divergence between two permutation probability distributions

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

ListNet

\[ f: \ f(A) = 3, f(B) = 0, f(C) = 1; \]
Ranking by \( f \): ABC

\[ g: \ g(A) = 6, g(B) = 4, g(C) = 3; \]
Ranking by \( g \): ABC

\[ h: \ h(A) = 4, h(B) = 6, h(C) = 3; \]
Ranking by \( h \): ACB

Using KL-divergence to measure difference between distributions

\[ \text{dis}(f, g) = 0.46 \]

\[ \text{dis}(g, h) = 2.56 \]
The total number of pairwise error of left and right are 13 and 11. However for IR measures like NDCG and ERR that emphasize the top few results, this is not what we want. The black arrows on the left denote the RankNet gradients, whereas what we really like are the red arrows on the right.

Instead of using a smooth approximation to the cost, and taking derivatives, write down the derivatives directly.

---

Outline

1. Introduction
2. Learning to Rank in IR
3. Categorization of Ranking Approaches
4. Pointwise approach
5. Pairwise approach
6. Listwise approach
7. Summary
Summary

- In learning to rank for IR: listwise approach is better.
- Good example of applying machine learning to solve real problems.
- Find the application.


