Ranking Refinement and its Application to Information Retrieval

Rong Jin, Hamed Valizadegan

Computer Science & Engineering
Michigan State University

Hang Li

Microsoft Research Asia
Ranking Refinement

- How to combine two sources of ranking information:
  - Source 1 (S1): existing ranking function
  - Source 2 (S2): users’ feedbacks

- The challenge is
  - The existing ranking function is imperfect
  - The users’ feedbacks are noisy
Applications

- **Relevance Feedback**
  - S1: retrieval algorithm (e.g. Language Models)
  - S2: relevance judgments by users

- **Recommendation System**
  - S1: ranking by collaborative filtering
  - S2: items rated by the target user

- **Online ranking**
  - S1: existing ranking algorithm
  - S2: additional training examples
Problem Definition

- $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$: set of instances to be ordered
- Two sources of ranking information:
  - Base ranker: $G : \mathbb{R}^d \rightarrow \mathbb{R}$
  - Users’ feedback:
    $$\mathcal{O} = \{(x_{i_k} \succ x_{j_k}) | k = 1, \ldots, m\}$$
- Goal: $F : \mathbb{R}^d \rightarrow \mathbb{R}$
Encode Base Ranker

\[ W = [W_{i,j}]_{n \times n}, \quad W_{i,j} = \frac{\exp(\lambda g_i)}{\exp(\lambda g_i) + \exp(\lambda g_j)} \]

where \( g_i \equiv G(x_i) \)

- \( \lambda \): confidence of ranking function
  - \( \lambda = 0 \rightarrow W_{i,j} = 0.5 \)
  - \( \lambda = \infty \rightarrow W_{i,j} = \begin{cases} 
1 & g_i > g_j \\
0.5 & g_i = g_j \\
0 & g_i < g_j 
\end{cases} \)
Encode User Feedback

$$T = [T_{i,j}]_{n \times n}, \quad T_{i,j} = \begin{cases} 
1 - \eta/2 & (x_i \succ x_j) \in \mathcal{O} \\
\eta/2 & \text{otherwise}
\end{cases}$$

$$\eta \in [0, 1]$$

- Related to probability of ranking $x_i$ before $x_j$
- Similar to regularization in SVM.
Ranking Errors

- Given a ranking function $F$, there are two types of errors:

  $$err_w = \sum_{i,j=1}^{n} W_{i,j} I(F_j \geq F_i)$$

  $$err_t = \sum_{i,j=1}^{n} T_{i,j} I(F_j \geq F_i)$$

- $err_w$: ranking error of $F$ relative to $W_{i,j}$
- $err_t$: ranking error of $F$ relative to $T_{i,j}$
Ranking Error (cont’d)

- Relaxation by exponential functions

\[
\hat{err}_w = \sum_{i,j=1}^{n} W_{i,j} \exp(F_j - F_i)
\]

\[
\hat{err}_t = \sum_{i,j=1}^{n} T_{i,j} \exp(F_j - F_i)
\]

- Upper bounds \( \hat{err}_w \geq err_w, \hat{err}_t \geq err_t \)
- Helps develop boosting algorithms
Combining Errors

- Linear Ranking Refinement (LRR)

\[ L_a = \gamma \hat{err}_w + \hat{err}_t \]

\[ = \sum_{i,j=1}^{n} (\gamma W_{i,j} + T_{i,j}) \exp(F_j - F_i) \]

- Drawback: need to decide parameter \( \gamma \)
Combining Errors (cont’d)

- Multiplicative Ranking Refinement (MRR)

\[
L_p = \hat{err}_w \times \hat{err}_t \\
= \left( \sum_{i,j=1}^{n} T_{i,j} \exp(F_j - F_i) \right) \left( \sum_{i,j=1}^{n} W_{i,j} \exp(F_j - F_i) \right)
\]

- Solution is Pareto efficient
Boosting Algorithm

Initialize $F(x) = 0$

$F(x)$

Compute the ranking uncertainty for each pair
$\gamma_{i,j} = a_{i,j} + b_{i,j}$

Compute the weight
$w_i = \sum_{j=1}^{n} \gamma_{i,j} - \gamma_{j,i}$

Sample data
$\text{abs}(w_i)$

Label data
$\text{sign}(w_i)$

$F(x) \leftarrow F(x) + \alpha f(x)$

Compute the combination weight $\alpha$

Train a binary classifier $f(x)$
Convergence of Boosting Alg.

- Aims to minimize

\[
L_p = \left( \sum_{i,j=1}^{n} T_{i,j} \exp(F_j - F_i) \right) \left( \sum_{i,j=1}^{n} W_{i,j} \exp(F_j - F_i) \right)
\]

- \(L_p\) is reduced exponentially

\[
L^T_p \leq \left( \sum_{i,j=1}^{n} T_{i,j} \right) \left( \sum_{i,j=1}^{n} W_{i,j} \right) \exp \left( - \sum_{k=1}^{T} \theta_k \right)
\]

where \(L^T_p\) is \(L_p\) at th Tth iteration
Convergence (cont’d)

![Convergence Diagram](image)
Experiments-data sets

- Letor test bed
  - OHSUMED dataset: 106 queries, 16140 query-document relevance judgment
  - TREC dataset: 1000 queries, 49171 query-document relevance judgment

- Movie recommendation
  - 943 users, 1682 movies
  - 51 binary features for each movie
Experimental Setup

- Following algorithms are compared:
  - Base ranker
  - Rocchio
  - SVM
  - Multiplicative Ranking Refinement (MRR)
  - Linear Ranking Refinement (LRR)
    - LRR-Worst: worst $\gamma$
    - LRR-Best: best $\gamma$
Evaluation Metrics

- Precision
  \[ P_R@k = \sum_{i=1}^{k} r_{R_i} / k \]

- Normalized Discounted Cumulative Gain (NDCG)
  \[ NDCG_R@k = \frac{DCG_R@k}{DCG_T@k} \]
  \[ DCG_X@k = \begin{cases} 
  r_{X_1} & \text{if } k = 1 \\
  r_{X_1} + \sum_{i=2}^{k} \frac{r_{X_i}}{\log_2 i} & \text{if } k > 1 
\end{cases} \]
Relevance Feedback: Precision

![Graph showing precision vs. top documents for different methods including Base Ranker, Rocchio, SVM, MRR, LRR-Worst, and LRR-Best with OHSUMED as a reference.]
Relevance Feedback: NDCG

![Graph showing NDCG values for different methods over top documents](image)
Relevance Feedback: Effect of Base Rankers

![Graph showing the effect of base rankers on MRR with OHSUMED data.]
Relevance Feedback: Number of Feedback Docs
Movie Recommendation: Precision

![Graph showing precision vs. top documents for different methods including Base Ranker, Rocchio, SVM, MRR, LRR-Worst, and LRR-Best.](image-url)
Movie Recommendation: Number of Rated Movies

![Graph showing precision vs. top documents for different ranking methods.](image)
Conclusion

- Present the problem of ranking refinement
- Proposed Boosting frameworks for ranking refinement
- Extensive studies with the proposed boosting framework for ranking refinement
Thank you!