# 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

- $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ :set of instances to be ordered
- Two sources of ranking information:
  - Base ranker: $G : \mathbb{R}^d \to \mathbb{R}$
  - Users' feedback:

$$\mathcal{O} = \{(\mathbf{x}_{i_k} \succ \mathbf{x}_{j_k}) | k = 1, \ldots, m\}$$

- Goal: $F : \mathbb{R}^d \to \mathbb{R}$

# 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 $g_i \equiv G(\mathbf{x}_i)$

- $\lambda$ : 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

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

$\eta \in [0, 1]$

- Related to probability of ranking $\mathbf{x}_i$ before $\mathbf{x}_j$
- 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

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

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

- Upper bounds $\widehat{err}_w \geq err_w, \ \widehat{err}_t \geq err_t$
- Helps develop boosting algorithms

# Combining Errors

- Linear Ranking Refinement (LRR)

$$\begin{aligned} L_a &= \gamma \widehat{err}_w + \widehat{err}_t \\ &= \sum_{i,j=1}^{n} (\gamma W_{i,j} + T_{i,j}) \exp(F_j - F_i) \end{aligned}$$

- Drawback: need to decide parameter $\gamma$

# Combining Errors (cont'd)

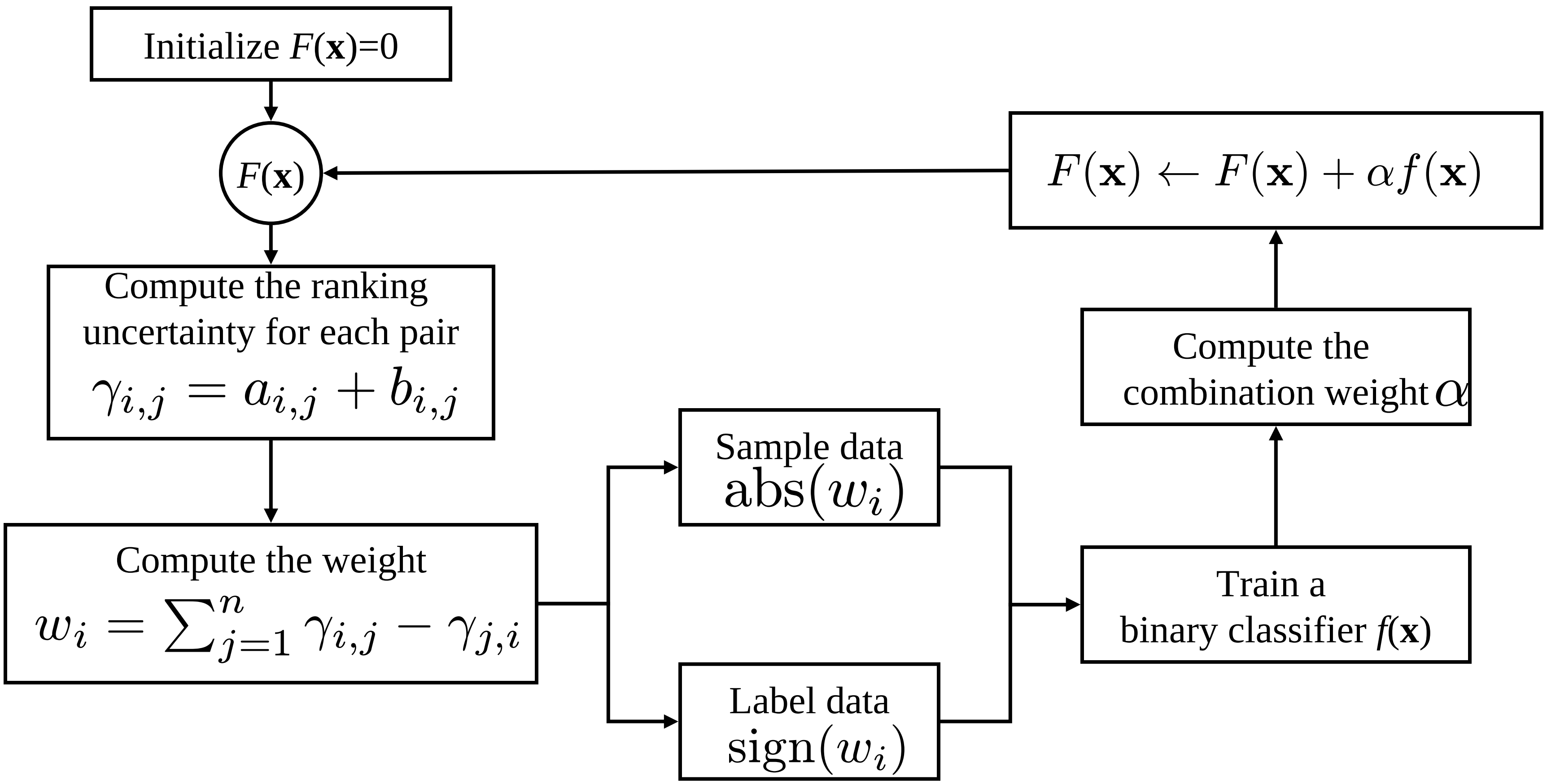
- Multiplicative Ranking Refinement (MRR)

$$L_p = \widehat{err}_w \times \widehat{err}_t$$

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

- Solution is Pareto efficient

# Boosting Algorithm



# Convergence of Boosting Alg.

- Aims to minimize

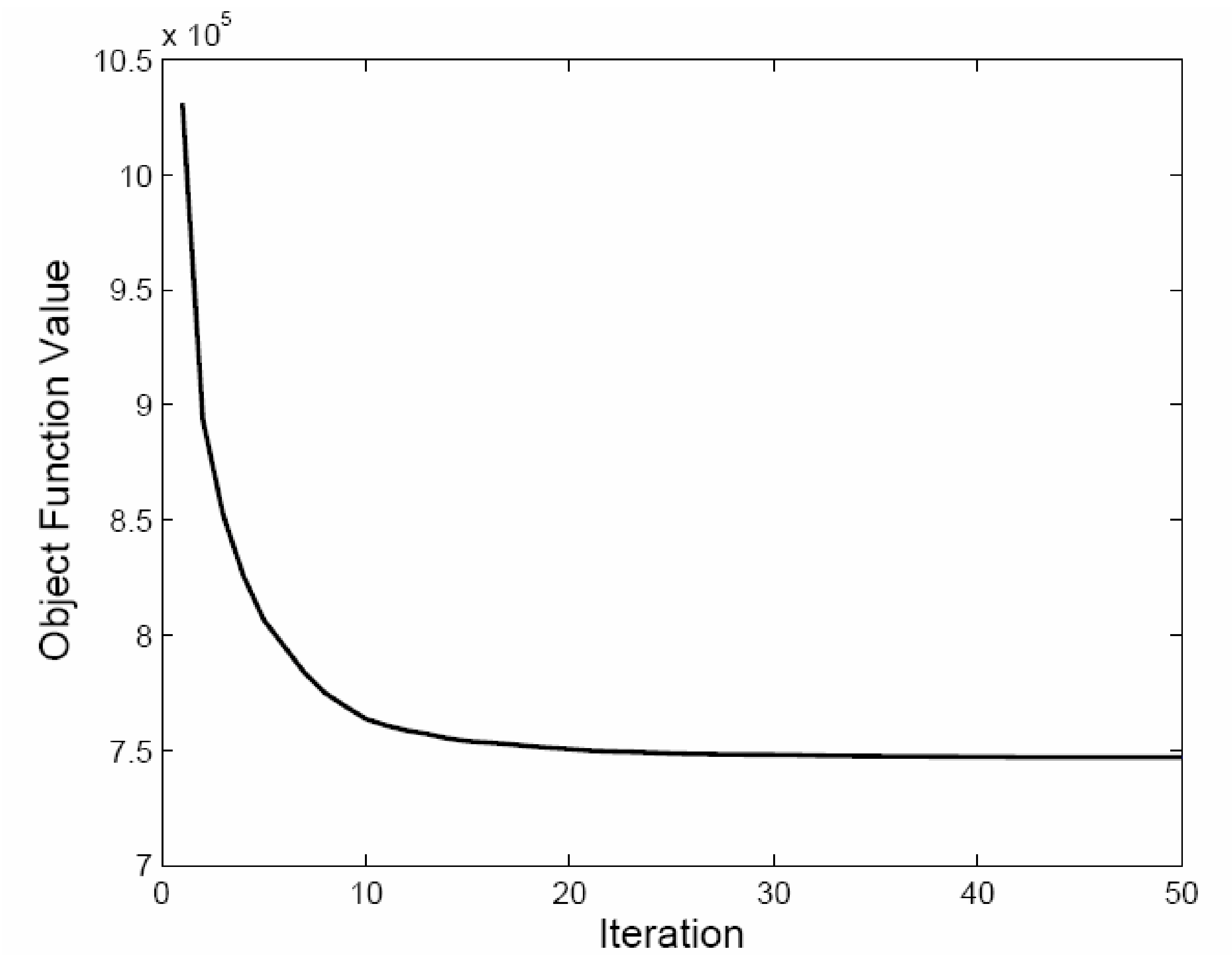
$$L_p = \left(\sum_{i,j=1}^{n} T_{i,j} \exp(F_j - F_i)\right)\left(\sum_{i,j=1}^{n} W_{i,j} \exp(F_j - F_i)\right)$$

- $L_p$ is reduced exponentially

$$L_p^T \leq \left(\sum_{i,j=1}^{n} T_{i,j}\right)\left(\sum_{i,j=1}^{n} W_{i,j}\right) \exp\left(-\sum_{k=1}^{T} \theta_k\right)$$

where $L_p^T$ is $L_p$ at th Tth iteration

# Convergence (cont'd)



# 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

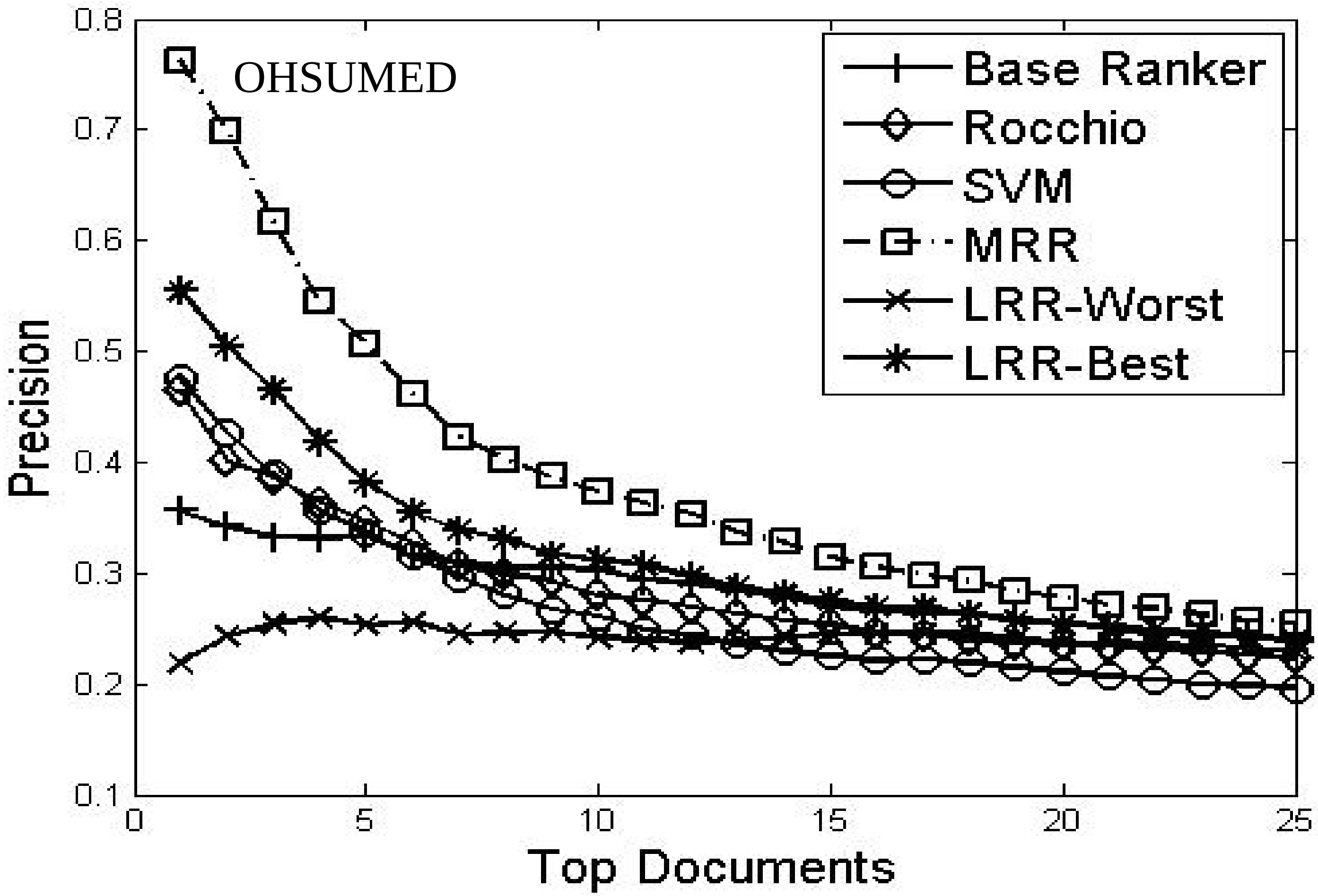
- Precision $P_R@k = \sum_{i=1}^{k} r_{R_i}/k$

- Normalized Discounted Cumulative Gain (NDCG)

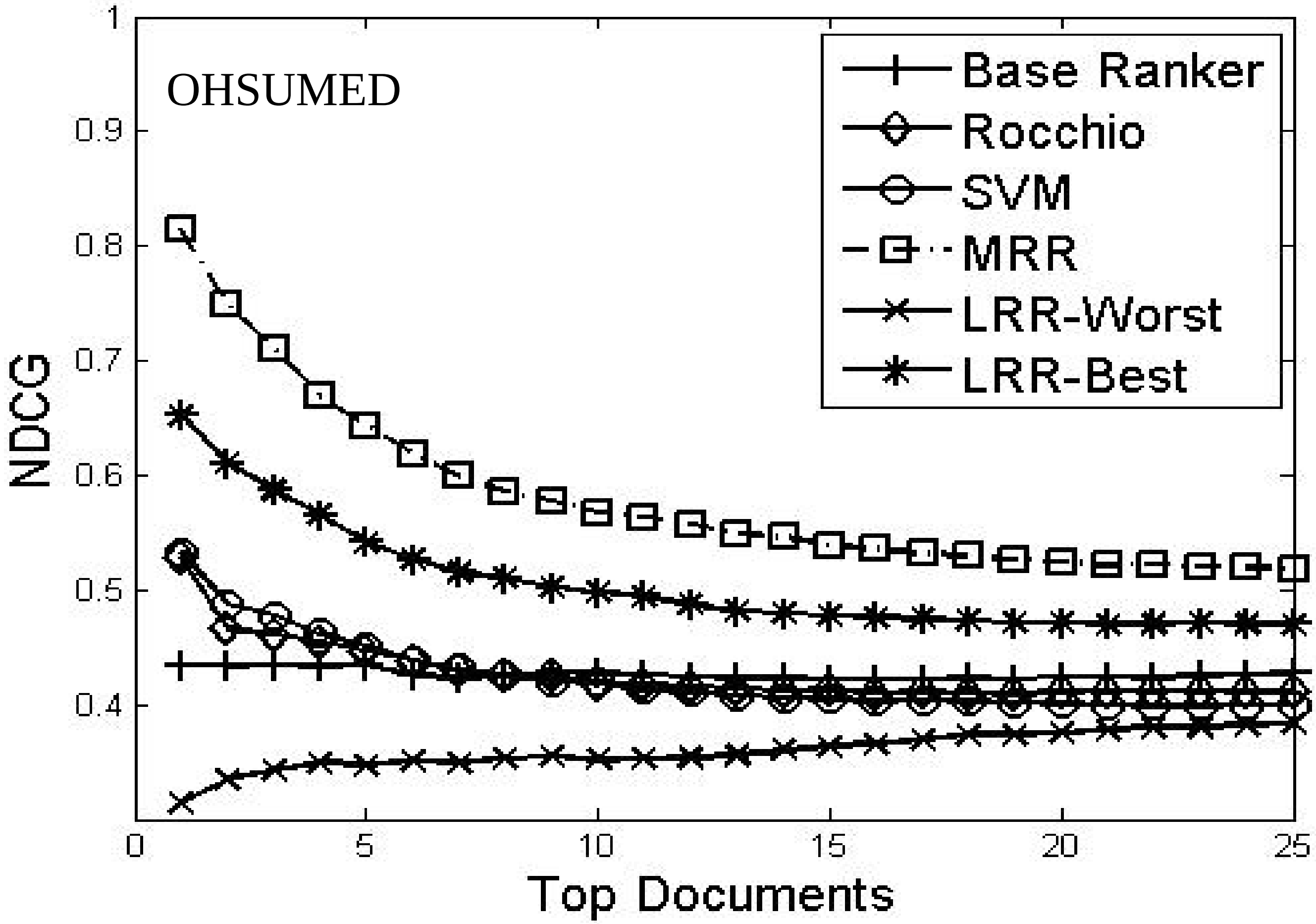
$$NDCG_R@k = \frac{DCG_R@k}{DCG_T@k}$$

$$DCG_X@k = \begin{cases} r_{X_1} & if\ k = 1 \\ r_{X_1} + \sum_{i=2}^{k} \frac{r_{X_i}}{\log_2 i} & if\ k > 1 \end{cases}$$

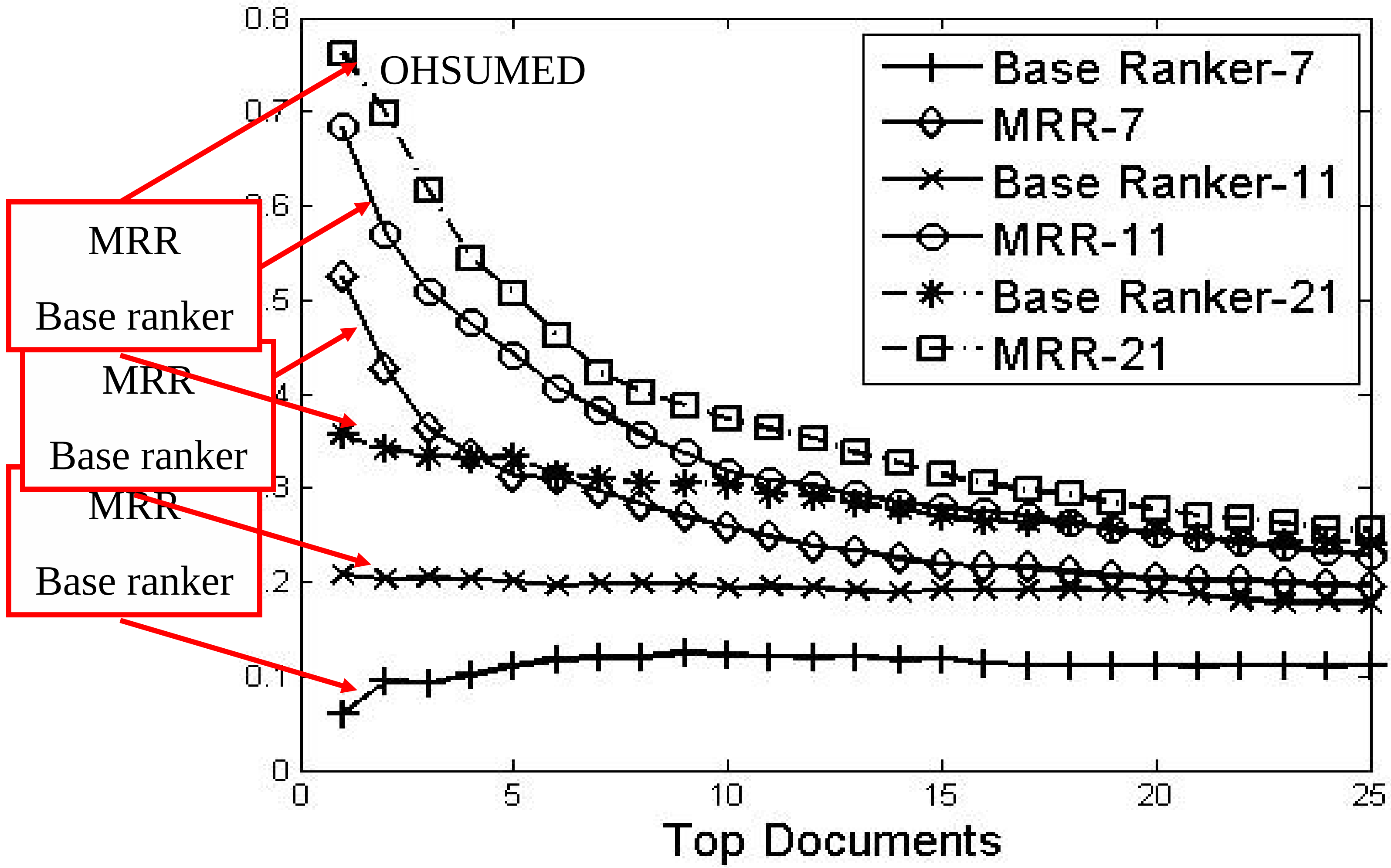
# Relevance Feedback: Precision



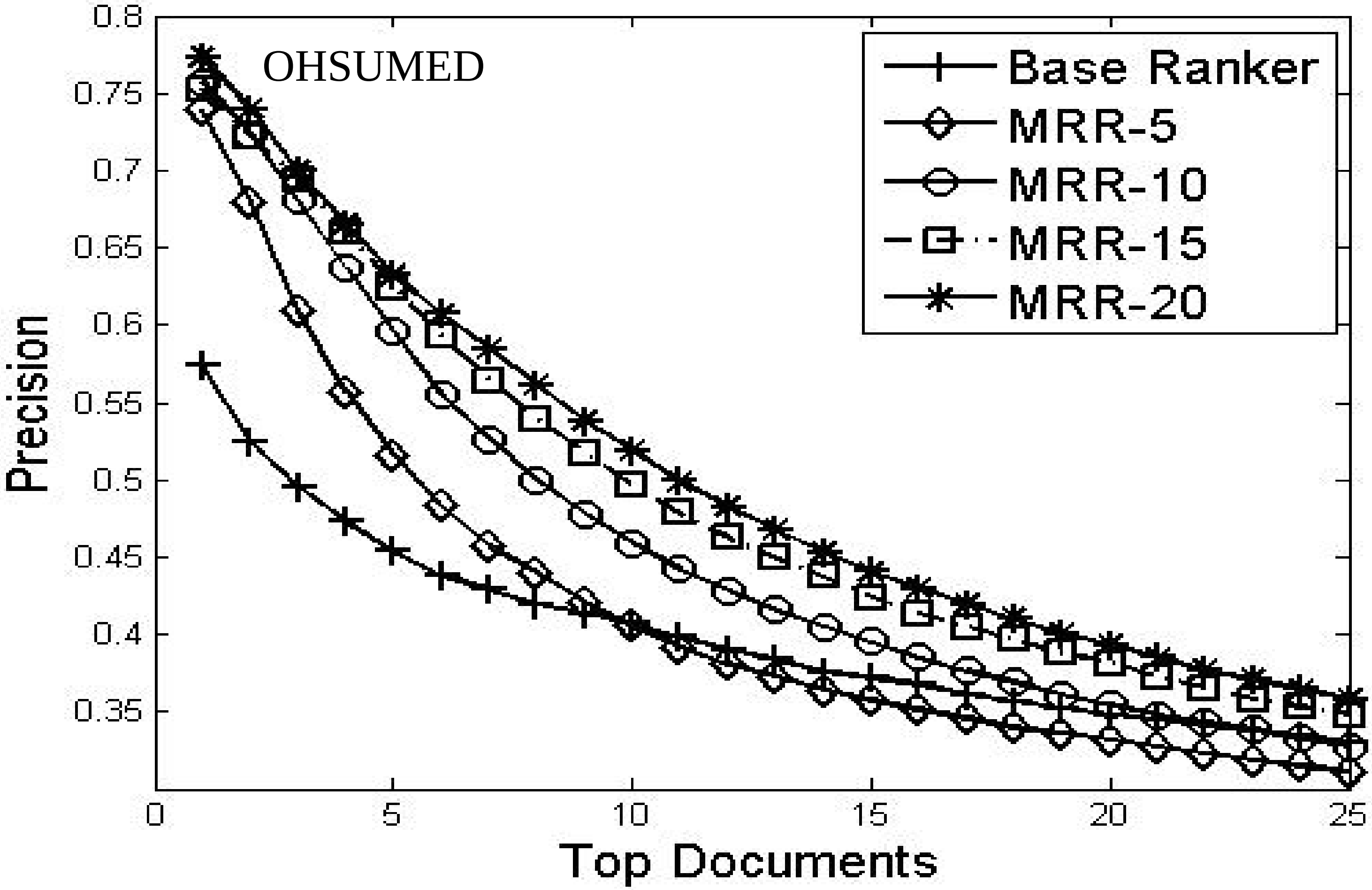
# Relevance Feedback: NDCG



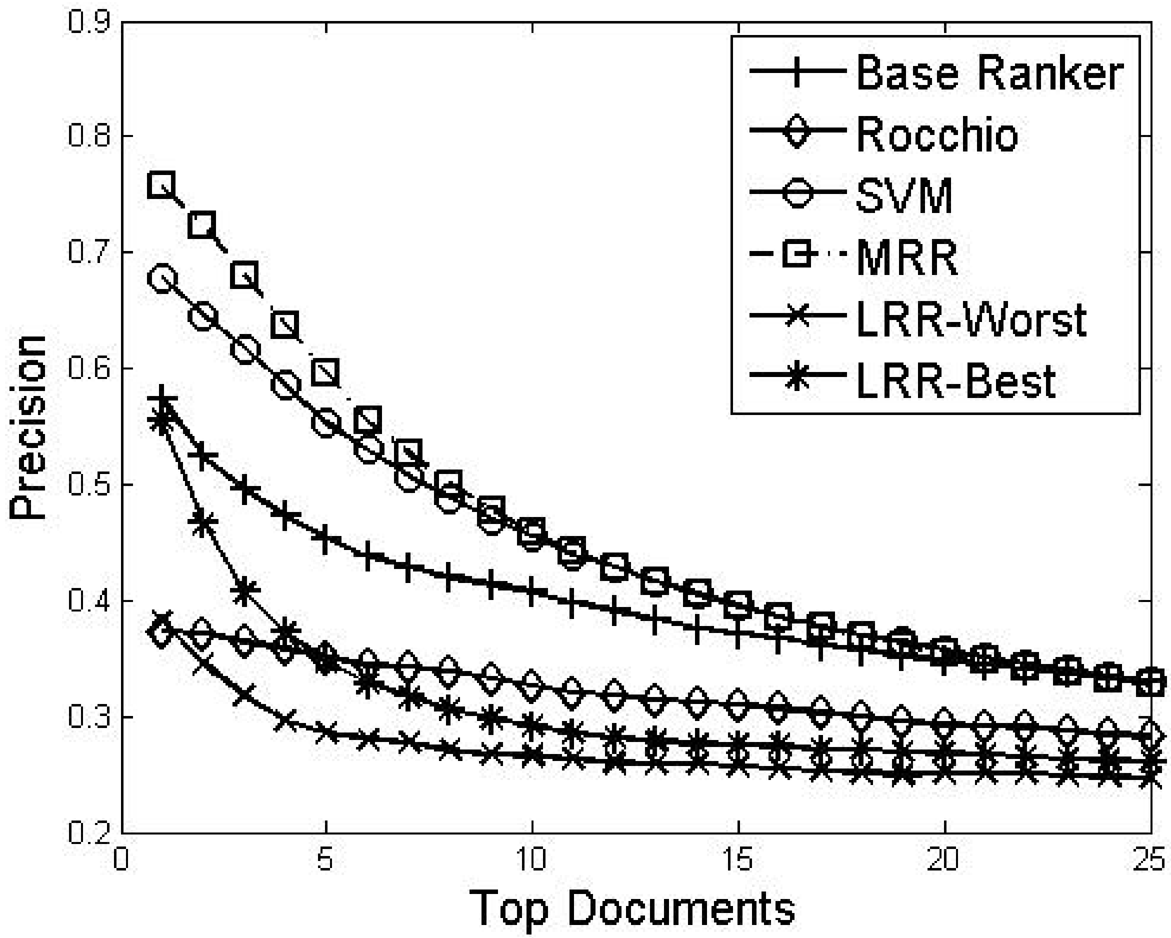
# Relevance Feedback: Effect of Base Rankers



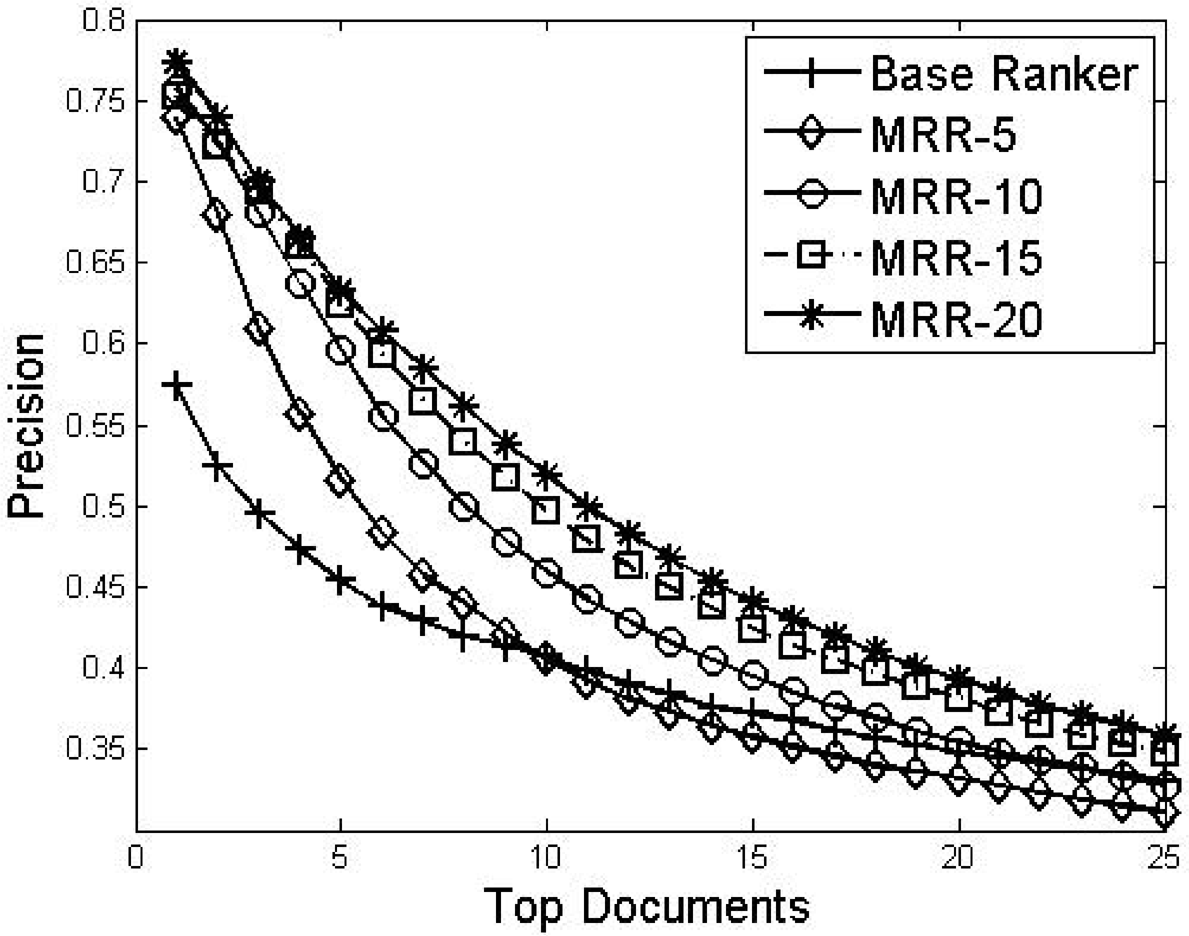
# Relevance Feedback: Number of Feedback Docs



# Movie Recommendation: Precision



# Movie Recommendation: Number of Rated Movies



# 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!