Reinforcement Learning to Rank with Markov Decision Process

Jun Xu
ICT, CAS
Outline

• Background: learning to rank for IR
• Reinforcement learning to rank
• Summary
Ranking is Important for Web Search

- **Criteria**
  - Relevance
  - Diversity
  - Freshness

- **Ranking model**
  - Heuristic
    - Relevance: BM25, LMIR
    - Diversity: MMR, xQuAD
  - Learning to rank
Ranking in Information Retrieval

query \( q \)

ranking model \( f(q, d) \)

document index \( D = \{d_i\} \)

\[ d_1, f(q, d_1) \]
\[ d_2, f(q, d_2) \]
\[ \vdots \]
\[ d_N, f(q, d_N) \]
Learning to Rank for Information Retrieval

• Machine learning algorithms for relevance ranking

Point-wise: ranking as regression or classification over query-documents
Pair-wise: ranking as binary classification over preference pairs
List-wise: training/predicting ranking at query (document list) level
Independent Relevance Assumption

- Utility of a doc is independent of other docs
- Ranking as scoring & sorting
  - Each document can be scored independently
  - Scores are independent of the rank
**Beyond Independent Relevance**

- More ranking criteria, e.g., search result diversification
  - Covering as much subtopics as possible with a few documents
  - Need consider the **novelty** of a document given preceding documents
- Complex application environment, e.g., Interactive IR
  - Human interacts with the system during the ranking process
  - User feedback is helpful for improving the remaining results

*Query: Programming language*

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java</td>
<td>Java</td>
</tr>
<tr>
<td>C++</td>
<td>Java</td>
</tr>
<tr>
<td>Python</td>
<td>Java</td>
</tr>
</tbody>
</table>

Need more powerful ranking mechanism!
Outline

• Background: learning to rank for IR

• Reinforcement learning to rank
  – Ranking as Markov decision process
  – Adapting MDP for relevance and diverse ranking

• Summary
From Scoring & Sorting to Sequential Decision Making

- Advantages: beyond independent relevance
  - Modeling the dependencies between documents
  - Taking the ranking positions into consideration
Markov Decision Process (MDP)

- An MDP is composed by states, actions, rewards, policy, and transitions, and represented by a tuple $\langle S, A, T, R, \pi \rangle$

- **States** $S$: a set of states.
- **Actions** $A$: a discrete set of actions that an agent can take.
- **Transition** $T$: the state transition function $s_{t+1} = T(s_t, a_t)$
- **Reward** $r = R(s, a)$: the immediate reward, also known as reinforcement
- **Policy** $\pi(a|s)$: a probability distribution over the possible actions.
Ranking as Markov Decision Process

- **Time steps:** ranks
- **State:** query, preceding docs, candidates, …
- **Policy:** distribution over remaining candidate documents
- **Action (Decision):** selecting a doc and placing it to current pos
- **Reward**
  - Additional utility (e.g., the increase of DCG) from the selected doc
  - Calculated based on widely used evaluation measures (e.g., DCG, ERR-IA)
Learning and Online Ranking

• Learning the parameters
  – Model parameters: policy function, state initialization and transition etc.
  – Reinforcement learning: policy gradient
  – Rewards based on relevance labels as supervision

• Online ranking
  – Without rewards (rewards are based on relevance labels)
  – Fully trust the learned policy
Example 1: Learning for Search Result Diversification

Search Result Diversification

Query: jaguar

- Query: information needs are ambiguous and multi-faceted
- Search results: may contain redundant information
- Goal: covering as much subtopics as possible with a few documents
Modeling Diverse Ranking with MDP

• Key points
  – Mimic user top-down browsing behaviors
  – Model dynamic information needs with MDP state

• States $s_t = [Z_t, X_t, h_t]$
  – $Z_t$: sequence of $t$ preceding documents, $Z_0 = \phi$
  – $X_t$: set of candidate documents, $X_0 = X$
  – $h_t \in R^K$: latent vector, encodes user perceived utility from preceding documents, initialized with the information needs form the query:
    $$h_0 = \sigma(V_q q)$$
# Modeling Diverse Ranking with MDP

<table>
<thead>
<tr>
<th>MDP factors</th>
<th>Corresponding diverse ranking factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time steps</td>
<td>The ranking positions</td>
</tr>
<tr>
<td>State</td>
<td>$s_t = [Z_t, X_t, h_t]$</td>
</tr>
<tr>
<td>Policy</td>
<td>$\pi(a_t</td>
</tr>
<tr>
<td>Action</td>
<td>Selecting a doc and placing it to rank $t + 1$</td>
</tr>
<tr>
<td>Reward</td>
<td>Based on evaluation measure $\alpha$DCG, SRecall etc. For example: $R = \alpha$DCG[$t + 1] - \alpha$DCG[$t$]; $R = \text{SRecall}[$t + 1$] - \text{SRecall}[$t$]$</td>
</tr>
<tr>
<td>State Transition</td>
<td>$s_{t+1} = T(s_t = [Z_t, X_t, h_t], a_t)$ $= [Z_t \oplus {x_{m(a_t)}}, X_t \setminus {x_{m(a_t)}}, \sigma(Vx_{m(a_t)} + Wh_t)]$</td>
</tr>
</tbody>
</table>

$x_{m(a_t)}$: document embedding
Ranking Process: Initialize State

\[ s_0 = [\phi, X, \sigma(V_q q)] \]
Ranking Process: Policy

Document ranking

Candidate documents

Query

Ranker

Calculate the policy

\[ \pi(a_t | s_t) = \frac{\exp\{x_m(a_t)Uh_t\}}{Z} \]
Ranking Process: Action

Document ranking → Ranker

Candidate documents → Ranker

Sample action according to policy

Query

doc at rank 1
Ranking Process: Reward

Get reward, e.g.,
\[ R = \alpha DCG[t + 1] - \alpha DCG[t] \]
Ranking Process: State Transition

Update ranked list, candidate set, and latent vector

\[ s_{t+1} = [Z_t \oplus \{x_{m(a_t)}\}, X_t \setminus \{x_{m(a_t)}\}, \sigma(V x_{m(a_t)} + W h_t)] \]
Ranking Process: Iterate

- Query
- Document ranking
- Candidate documents
- Ranker

Query leads to Document ranking, which is then fed into the Ranker. The Ranker iterates, assigning ranks to documents, starting with doc at rank 1, followed by doc at rank 2, and continuing with doc at rank 3.
Learning with Policy Gradient

- Model parameters $\Theta = \{V_q, U, V, W\}$
- Learning objective: maximizing expected return (discounted sum of rewards) of each training query

$$\max_{\Theta} v(q) = E_{\pi} G_0 = E_{\pi} \left[ \sum_{k=0}^{M-1} \gamma^k r_{k+1} \right]$$

  - Directly optimizes evaluation measure as $G_0 = \alpha \text{DCG}@M$
- Monte-Carlo stochastic gradient ascent is used to conduct the optimization (REINFORCE algorithm)

$$\nabla_{\Theta} v(q) = \gamma^t G_t \nabla_{\Theta} \log \pi(a_t|s_t; \Theta)$$
Analysis

• Optimize general diversity evaluation measures (e.g., $\alpha$-DCG, S-recall)

\[ \max_{\Theta} V(q) = \mathbb{E}_{\pi} G_0 \]

• Given an episode and time step $t$

\[
\nabla_{\Theta} V(\Theta) \overset{\text{sample}}{=} \gamma^t \sum_{a \in A(s_t)} \nabla_{\Theta} \pi(a|s_t) Q^\pi(s_t, a) \\
= \gamma^t \sum_{a \in A(s_t)} \pi(a|s_t) \cdot \left( Q^\pi(s_t, a) \frac{\nabla_{\Theta} \pi(a|s_t)}{\pi(a|s_t)} \right) \\
= \gamma^t Q^\pi(s_t, a_t) \frac{\nabla_{\Theta} \pi(a_t|s_t)}{\pi(a_t|s_t)} \\
= \gamma^t G_t \nabla_{\Theta} \log \pi(a_t|s_t)
\]

discounted sum of the rewards, starting from position 0 (return)

Maximizing the return starting from position $t$
The Learning Algorithm

Algorithm 1 MDP-DIV learning

Input: Labeled training set $D = \{(q^{(n)}, X^{(n)}, r^{(n)})\}_{n=1}^{N}$, learning rate $\eta$, discount factor $\gamma$, and reward function $r$

Output: $\Theta = \{V_q, U, V, W\}$

1. Initialize $\Theta = \{V_q, U, V, W\} \leftarrow$ random

2. repeat

3. for all $(q, X, J) \in D$ do

4. $(s_0, a_0, r_1, \cdots, s_{M-1}, a_{M-1}, r_M) \leftarrow$ Sample a SampleEpisode

5. for $t = 0$ to $M - 1$ do

6. $G_t \leftarrow \sum_{k=0}^{M-1-t} \gamma^k r_{t+k+1}$ \{Equation (2), and $M = |X|$\}

7. $\Theta \leftarrow \Theta + \eta \gamma^t G_t \nabla \Theta \log \pi(a_t|s_t; \Theta)$

8. end for

9. end for

10. until converge

11. return $\Theta$

Algorithm 2 SampleEpisode

Input: Parameters $\Theta = \{V_q, U, V, W\}$, $q, X, J$, and $R$

Output: An episode

1. Initialize $s \leftarrow [\emptyset, X, \sigma(Vqq)]$ \{Equation (1)\}

2. $M \leftarrow |X|$ \{Algorithm (2), and $M = |X|$\}

3. $E = ()$ \{empty episode\}

4. for $t = 0$ to $M - 1$ do

5. $A \leftarrow A(s)$ \{Possible actions according to $X$ in state $s$\}

6. for all $a \in A$ do

7. $P_a \leftarrow \pi(a|s; \Theta)$

8. end for

9. Sample an action $\hat{a} \in A$, according to $P$

10. $r \leftarrow R(s, \hat{a})$ \{Calculation on the basis of $J$\}

11. Append $(s, \hat{a}, r)$ to the tail of $E$

12. $[Z, X, h] \leftarrow s$

13. $s \leftarrow [Z \oplus \{x_{m(\hat{a})}\}, X \setminus \{x_{m(\hat{a})}\}, \sigma(Vx_{m(\hat{a})} + Wh)]$

14. end for

15. return $E = (s_0, a_0, r_1, \cdots, s_{M-1}, a_{M-1}, r_M)$
Online Ranking Algorithm

• Fully trust the policy

```
Algorithm 3 MDP-DIV online ranking

Input: Parameters Θ = {V_q, U, V, W}, query q, documents X
Output: Permutation of documents τ

1: Initialize s ← [∅, X, σ(V_q q)]{Equation (1)}
2: M ← |X|
3: for t = 0 to M − 1 do
4:    A ← A(s) {Possible actions according to X in state s}
5:    ˆa ← arg max_{a ∈ A} π(a|s; Θ) {Choosing most possible action}
6:    τ[t + 1] ← m(ˆa) {Document x_{m(ˆa)} is ranked at t + 1}
7:    [Z, X, h] ← s
8:    s ← [Z ⊕ {x_{m(ˆa)}}], X \ {x_{m(ˆa)}}, σ(V x_{m(ˆa)} + Wh)]
9: end for
10: return τ
```

using max instead of sampling
Experimental Results

<table>
<thead>
<tr>
<th>Method</th>
<th>$\alpha$-NDCG@5</th>
<th>$\alpha$-NDCG@10</th>
<th>S-recall@5</th>
<th>S-recall@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>MMR</td>
<td>0.2753</td>
<td>0.2979</td>
<td>0.4388</td>
<td>0.5151</td>
</tr>
<tr>
<td>xQuAD</td>
<td>0.3165</td>
<td>0.3941</td>
<td>0.4933</td>
<td>0.6043</td>
</tr>
<tr>
<td>PM-2</td>
<td>0.3047</td>
<td>0.3730</td>
<td>0.4910</td>
<td>0.6012</td>
</tr>
<tr>
<td>SVM-DIV</td>
<td>0.3030</td>
<td>0.3699</td>
<td>0.5122</td>
<td>0.6230</td>
</tr>
<tr>
<td>R-LTR</td>
<td>0.3498</td>
<td>0.4132</td>
<td>0.5397</td>
<td>0.6511</td>
</tr>
<tr>
<td>PAMM($\alpha$-NDCG)</td>
<td>0.3712</td>
<td>0.4327</td>
<td>0.5561</td>
<td>0.6612</td>
</tr>
<tr>
<td>NTN-DIV($\alpha$-NDCG)</td>
<td>0.3962</td>
<td>0.4577</td>
<td>0.5817</td>
<td>0.6872</td>
</tr>
<tr>
<td>MDP-DIV(S-recall)</td>
<td>0.4156</td>
<td>0.4734</td>
<td><strong>0.6123</strong></td>
<td><strong>0.7155</strong></td>
</tr>
<tr>
<td>MDP-DIV($\alpha$-DCG)</td>
<td><strong>0.4189</strong></td>
<td><strong>0.4762</strong></td>
<td>0.6102</td>
<td>0.7117</td>
</tr>
</tbody>
</table>

- Based on combination of TREC 2009 ~ 2012 Web Track
- Directly optimize a predefined measure via defining the rewards based on the measure
How it works?
Using Query 93 as Example

$q$
raffles

[1]: “Raffles Hotel in Singapore”
[2]: “Sir Stamford Raffles”
[3]: “organizing a raffle”
[4]: “the Raffles hotel in Dubai”
[5]: “car raffles”

$d_1$: “Stamford Raffles – Wikipedia, the free encyclopedia” [2]
$d_2$: “Fundraiser Raffle Ideas” [3, 5]
$d_3$: “Luxury Hotel Guide | Raffles Hotels” [1, 4]
$d_5$: “Raffles Hotels and Resorts” [1, 4]
How it works?
Using Query 93 as Example

[1]: “Raffles Hotel in Singapore”
[2]: “Sir Stamford Raffles”
[3]: “organizing a raffle”
[4]: “the Raffles hotel in Dubai”
[5]: “car raffles”

\[ d_1 : “Stamford Raffles – Wikipedia, the free encyclopedia” \]
\[ d_2 : “Fundraiser Raffle Ideas” \]
\[ d_3 : “Luxury Hotel Guide | Raffles Hotels” \]
\[ d_4 : “National Corvette Museum – Corvette Raffles” \]
\[ d_5 : “Raffles Hotels and Resorts” \]
How it works?
Using Query 93 as Example

[1]: “Raffles Hotel in Singapore”
[2]: “Sir Stamford Raffles”
[3]: “organizing a raffle”
[4]: “the Raffles hotel in Dubai”
[5]: “car raffles”

d1: “Stamford Raffles – Wikipedia, the free encyclopedia” [2]
d2: “Fundraiser Raffle Ideas” [3, 5]
d3: “Luxury Hotel Guide | Raffles Hotels” [1, 4]
d5: “Raffles Hotels and Resorts” [1, 4]
How it works?
Using Query 93 as Example

[1] : “Raffles Hotel in Singapore”

$\textbf{d}_1 : \text{“Stamford Raffles – Wikipedia, the free encyclopedia”} \ [2]$
$\textbf{d}_2 : \text{“Fundraiser Raffle Ideas”} [3, 5]$
$\textbf{d}_3 : \text{“Luxury Hotel Guide | Raffles Hotels”} [1, 4]$
$\textbf{d}_4 : \text{“National Corvette Museum – Corvette Raffles”} [5]$
$\textbf{d}_5 : \text{“Raffles Hotels and Resorts”} [1, 4]$
How it works?
Using Query 93 as Example

[ranking: ]

<table>
<thead>
<tr>
<th></th>
<th>doc:subtopics</th>
<th>ranking score</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_1$</td>
<td>[2]</td>
<td>0.51</td>
</tr>
<tr>
<td>$d_2$</td>
<td>[3, 5]</td>
<td>1.19</td>
</tr>
<tr>
<td>$d_3$</td>
<td>[1, 4]</td>
<td>1.05</td>
</tr>
<tr>
<td>$d_4$</td>
<td>[5]</td>
<td>0.46</td>
</tr>
<tr>
<td>$d_5$</td>
<td>[1, 4]</td>
<td>1.01</td>
</tr>
</tbody>
</table>

[1] : “Raffles Hotel in Singapore”

d_1 : “Stamford Raffles – Wikipedia, the free encyclopedia” [2]
d_2 : “Fundraiser Raffle Ideas” [3, 5]
d_3 : “Luxury Hotel Guide | Raffles Hotels” [1, 4]
d_5 : “Raffles Hotels and Resorts” [1, 4]
How it works?
Using Query 93 as Example

ranking:  

[1]  "Raffles Hotel in Singapore"  
[2]  "Sir Stamford Raffles"  
[3]  "organizing a raffle"  
[4]  "the Raffles hotel in Dubai"  
[5]  "car raffles"  

[2]  "Fundraiser Raffle Ideas"  
[3]  "Luxury Hotel Guide | Raffles Hotels"  
[5]  "Raffles Hotels and Resorts"  

[1]  "Stamford Raffles — Wikipedia, the free encyclopedia"  
[2]  "Fundraiser Raffle Ideas"  
[3]  "Luxury Hotel Guide | Raffles Hotels"  
[5]  "Raffles Hotels and Resorts"
How it works?
Using Query 93 as Example

ranking: \{ d_2 : [3, 5], d_5 : [1, 4] \}

[1] : “Raffles Hotel in Singapore”
How it works?
Using Query 93 as Example


[1] : “Raffles Hotel in Singapore”

d1 : “Stamford Raffles – Wikipedia, the free encyclopedia” [2]
d2 : “Fundraiser Raffle Ideas” [3, 5]
d3 : “Luxury Hotel Guide | Raffles Hotels” [1, 4]
d5 : “Raffles Hotels and Resorts” [1, 4]
Using Immediate Rewards in Training

Train with $\alpha$-DCG@k (k=1, ..., M)

Train with $\alpha$-DCG@M

Figure 4: The performance curves on the test data for MDP-DIV($\alpha$-DCG), and the modified MDP-DIV($\alpha$-DCG) in which the training only involves the long-term returns. The performances of other baselines are shown as horizontal lines.
Convergence and Online Ranking Criterion

$$\hat{a} \leftarrow \arg \max_{a \in A} \pi(a|s; \Theta)$$

**for all** $a \in A$ **do**

$$P(a) \leftarrow \pi(a|s; \Theta)$$

**end for**

Sample an action $\hat{a} \in A$, according to $P$

**Figure 5**: The performance curves in terms of $\alpha$-DCG on the training data ("train(arg max)") and the test data ("test(arg max)"). The average performances of the sampled rankings over all training queries are also shown ("train(sample)").
Advantages

• Unified criterion (additional utility user can perceive) for selecting documents at each iteration

• End-to-end learning of the diverse ranking model
  – No need of handcrafted features

• Utilizes both the immediate rewards and the long-term returns as the supervision information during training
Example 2: Relevance Ranking as an MDP

## Modeling Relevance Ranking with MDP

<table>
<thead>
<tr>
<th>MDP factors</th>
<th>Corresponding relevance ranking factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time steps</td>
<td>The ranking positions</td>
</tr>
<tr>
<td>State</td>
<td>( s_t = [t, X_t] )</td>
</tr>
<tr>
<td>Policy</td>
<td>( \pi(a_t</td>
</tr>
<tr>
<td>Action</td>
<td>Selecting a doc and placing it to current position</td>
</tr>
<tr>
<td>Reward</td>
<td>Based on evaluation measure DCG: ( R = \begin{cases} 2^{y(a_t)} - 1 &amp; t = 0 \ \frac{2^{y(a_t)} - 1}{\log_2(t+1)} &amp; t &gt; 0 \end{cases} )</td>
</tr>
<tr>
<td>State Transition</td>
<td>( s_{t+1} = T(s_t = [t, X_t], a_t) = [t + 1, X_t \setminus {\mathbf{x}_{m(a_t)}},] )</td>
</tr>
</tbody>
</table>

\( \mathbf{x}_{m(a_t)} \): query-doc relevance features
The Ranking Process

State: \( s_t = [t, X_t] \)

1. the ranking position
2. the documents in candidate set

Reward: evaluate the action

\[
 r_t = R_{\text{DCG}}(s_t, a_t) = \left\{ \begin{array}{ll}
 2^{y_{m(a_t)}} - 1 & t = 0 \\
 \frac{2^{y_{m(a_t)}} - 1}{\log_2(t+1)} & t > 0
 \end{array} \right.
\]

Action: select a document

\[
 a_t \sim \pi(a_t|s_t; w) = \frac{\exp \{ w^T x_{m(a_t)} \}}{\sum_{a \in A(s_t)} \exp \{ w^T x_{m(a)} \}}
\]

Measure metric as reward

Move the selected document from the candidate set to the ranking list.
Learning with Policy Gradient

Algorithm 1 MDPRank learning

Input: Labeled training set \( D = \{(q^{(n)}, X^{(n)}, Y^{(n)})\}_{n=1}^{N} \), learning rate \( \eta \), discount factor \( \gamma \), and reward function \( R \)

Output: \( w \)

1. Initialize \( w \leftarrow \) random values
2. repeat
3. \( \Delta w = 0 \)
4. for all \( (q, X, Y) \in D \) do
5. \( (s_0, a_0, r_1, \ldots, s_{M-1}, a_{M-1}) \) \{Algorithm (2), and \( M = \gamma \}
6. for \( t = 0 \) to \( M - 1 \) do
7. \( G_t = \sum_{k=1}^{M-t} \gamma^{k-1} r_t \)
8. \( \Delta w \leftarrow \Delta w + \gamma^t G_t \nabla_w \}
9. end for
10. end for
11. \( w \leftarrow w + \eta \Delta w \)
12. until converge
13. return \( w \)

Algorithm 2 SampleAnEpisode

Input: Parameters \( w, q, X, Y, \) and \( \mathcal{R} \)

Output: An episode

1. Initialize \( s_0 \leftarrow [0, X], M \leftarrow |X|, \) and episode \( E \leftarrow \emptyset \)
2. for \( t = 0 \) to \( M - 1 \) do
3. Sample an action \( a_t \in A(s_t) \sim \pi(a_t | s_t; w) \) \{Equation (2)\}
4. \( r_{t+1} \leftarrow \mathcal{R}(s_t, a_t) \{Equation (1), calculation on the basis of \( Y \}\}
5. Append \( (s_t, a_t, r_{t+1}) \) at the end of \( E \)
6. State transition \( s_{t+1} \leftarrow [t+1, X \setminus \{x_{m(a_t)}\}] \)
7. end for
8. return \( E = (s_0, a_0, r_1, \ldots, s_{M-1}, a_{M-1}, r_M) \)
Experimental Results

Result on MQ2007 Dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>NDCG@1</th>
<th>NDCG@3</th>
<th>NDCG@5</th>
<th>NDCG@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>RankSVM</td>
<td>0.4045</td>
<td>0.4019</td>
<td>0.4072</td>
<td>0.4383</td>
</tr>
<tr>
<td>ListNet</td>
<td>0.4002</td>
<td>0.4091</td>
<td>0.4170</td>
<td>0.4440</td>
</tr>
<tr>
<td>AdaRank-MAP</td>
<td>0.3821</td>
<td>0.3984</td>
<td>0.4071</td>
<td>0.4335</td>
</tr>
<tr>
<td>AdaRank-NDCG</td>
<td>0.3876</td>
<td>0.4044</td>
<td>0.4102</td>
<td>0.4369</td>
</tr>
<tr>
<td>SVMMAP</td>
<td>0.3853</td>
<td>0.3899</td>
<td>0.3983</td>
<td>0.4187</td>
</tr>
<tr>
<td>MDPRank</td>
<td>0.4061</td>
<td>0.4101</td>
<td>0.4171</td>
<td>0.4416</td>
</tr>
<tr>
<td>MDPRank(return only)</td>
<td>0.4033</td>
<td>0.4059</td>
<td>0.4113</td>
<td>0.4350</td>
</tr>
</tbody>
</table>

Result on OHSUMED Dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>NDCG@1</th>
<th>NDCG@3</th>
<th>NDCG@5</th>
<th>NDCG@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>RankSVM</td>
<td>0.4958</td>
<td>0.4207</td>
<td>0.4164</td>
<td>0.4140</td>
</tr>
<tr>
<td>ListNet</td>
<td>0.5326</td>
<td>0.4732</td>
<td>0.4432</td>
<td>0.4410</td>
</tr>
<tr>
<td>AdaRank-MAP</td>
<td>0.5388</td>
<td>0.4682</td>
<td>0.4613</td>
<td>0.4429</td>
</tr>
<tr>
<td>AdaRank-NDCG</td>
<td>0.5330</td>
<td>0.4790</td>
<td>0.4673</td>
<td>0.4496</td>
</tr>
<tr>
<td>SVMMAP</td>
<td>0.5229</td>
<td>0.4663</td>
<td>0.4516</td>
<td>0.4319</td>
</tr>
<tr>
<td>MDPRank</td>
<td>0.5925</td>
<td>0.4992</td>
<td>0.4909</td>
<td>0.4587</td>
</tr>
<tr>
<td>MDPRank(return only)</td>
<td>0.5363</td>
<td>0.4885</td>
<td>0.4694</td>
<td>0.4591</td>
</tr>
</tbody>
</table>

• MDPRank is better because
  – Utilize the IR measures calculated at all the ranking positions as supervision information for training
  – Directly optimizes the IR measure on the training data without any approximation or upper bounding
Outline

• Background: learning to rank for IR
• Reinforcement learning to rank
• Summary
Summary

• Reinforcement learning to rank
  – Ranking as sequential decision making
  – Adapting MDP for the task
  – Learning with policy gradient

• Two examples
  – Diverse ranking
  – Relevance ranking
Easy Machine Learning Project
Design of Easy Machine Learning

Data Storage and Management
- Large scale data management: HDFS
- Structured data management: MySQL

Distributed Computing
- Map-Reduce
- Spark
- TensorFlow

Scheduling: Oozie
- Execute command lines
- Program status

Interactive GUI (GWT)
- Designer
- Monitor

Workflow DAG
- Submit Oozie job

Node: program / data
Edge: dataflow


Node: program / start / end / fork / join
Edge: dependency
Deploy as Web Service

http://159.226.40.104:18080/dev

- **Advantages**
  - **Sharing**: share data/programs/tasks among users
  - **Collaborating**: working together for one task
  - **Mobility**: accessing with web browsers anywhere
  - **Open**: ETL for data import/export; can run third-party programs
Source Shared at Github

https://github.com/ICT-BDA/EasyML

- Top 1 Java project at Github trending for one week
- 1400 + stars and ~300 forks
- CIKM 2016 best demo candidate
  [Guo et al., CIKM ’16]
Thanks!

junxu@ict.ac.cn
http://www.bigdatalab.ac.cn/~junxu