Unbiased Learning to Rank with Unbiased Propensity Estimation

Qingyao Ai\textsuperscript{1}, Keping Bi\textsuperscript{1}, Cheng Luo\textsuperscript{2}, Jiafeng Guo\textsuperscript{3}, W. Bruce Croft\textsuperscript{1}

\textsuperscript{1}CICS, University of Massachusetts Amherst
\textsuperscript{2}DCST, Tsinghua University
\textsuperscript{3}ICT, Chinese Academy of Sciences

aiqy@cs.umass.edu
Learning to Rank from User Feedback

User

Result page

Interaction

Search Engine

Learning to rank models

- SVMrank [4]
- Duet [1]
- LambdaMART [3]
- DNN [5]
- DRMM [2]  ......
Learning to Rank from Click Data

- Click data are cheap but biased.

Unbiased Learning to Rank!
Problem Analysis

$P(c_i = 1) = P(o_i = 1) \cdot P(r_i = 1)$

Examination Propensity

Relevance Feedback

Bias!
Previous Studies: Click Model

- **Click Models**
  - Extract **true relevance feedback** with user behavior hypothesis [12].
Previous Studies: Click Model

- Problems
  - Offline parameter estimation with EM algorithm.
  - Require observing each query-doc pair in multiple sessions.
  - Not work for new documents.
• Unbiased Learning to Rank with Counterfactual Learning
  – No need to estimate the true relevance feedback.

Ranking loss based on clicks:

\[
\hat{l}(S, q) = \sum_{x_i \in \pi_q, c_i = 1} \Delta(x_i, c_i | \pi_q)
\]

Inverse Propensity Weighting

Ranking loss based on relevance labels:

\[
l(S, q) = \sum_{x_i \in \pi_q, r_i = 1} \Delta(x_i, r_i | \pi_q)
\]

\[\Delta(x, c)\]

\[\Delta(x, r)\]

\[x_i: \text{ranking score}\]
\[c_i: \text{user click}\]
\[r_i: \text{true relevance label}\]
\[\pi_q: \text{rank list}\]
• Inverse Propensity Weighting [6-8]:

\[ l_{IPW}(S, q) = \sum_{x_i \in \pi_q, c_i = 1} \frac{\Delta(x_i, c_i | \pi_q)}{P(o_i = 1 | \pi_q)} \]

\[ l(S, q) = \sum_{x_i \in \pi_q, r_i = 1} \Delta(x_i, r_i | \pi_q) \]

\[ \mathbb{E}_{o_q}[l_{IPW}(S, q)] = \mathbb{E}_{o_q} \left[ \sum_{x_i \in \pi_q, o_i = 1, r_i = 1} \frac{\Delta(x_i, r_i | \pi_q)}{P(o_i = 1 | \pi_q)} \right] \]

\[ = \mathbb{E}_{o_q} \left[ \sum_{x_i \in \pi_q, r_i = 1} o_i \cdot \frac{\Delta(x_i, r_i | \pi_q)}{P(o_i = 1 | \pi_q)} \right] \]

\[ = \sum_{x_i \in \pi_q, r_i = 1} \mathbb{E}_{o_q}[o_i] \cdot \frac{\Delta(x_i, r_i | \pi_q)}{P(o_i = 1 | \pi_q)} \]

\[ = \sum_{x_i \in \pi_q, r_i = 1} P(o_i = 1 | \pi_q) \cdot \frac{\Delta(x_i, r_i | \pi_q)}{P(o_i = 1 | \pi_q)} \]

\[ = \sum_{x_i \in \pi_q, r_i = 1} \Delta(x_i, r_i | \pi_q) \]

\[ c_i = 1 \Rightarrow o_i = 1, r_i = 1 \]
Previous Studies: Unbiased LTR

- Propensity Estimation \( P(o_i = 1) \)
  - Online randomization experiments [6, 7]
Previous Studies: Unbiased LTR

• Problems
  – Online randomization hurts user experience
  – No adaptive to the changes of user behavior.
Summary of Existing Solutions

**Click Models**
- Offline EM process.
- Require duplicated sessions.
- Not applicable to new documents.

**Unbiased LTR Frameworks**
- Not end-to-end.
- Not adaptive to user behavior changes.
- Online result randomization.
Summary of Existing Solutions

**Click Models**

- Offline EM process.
- Not adaptive to user behavior changes.

**Unbiased LTR Frameworks**

- Online result randomization.
- Require duplicated sessions.
- Not applicable to new documents.
- Not end-to-end.

---

**Our goal:**

An unbiased learning-to-rank algorithm that:

1. End-to-end, no separate experiment.
2. Not hurt user experience.
3. Adaptive to the changes of user behavior.
Re-Visit the Problem

- Why existing unbiased learning-to-rank methods need online result randomization?

Inverse Propensity Weighting:

$$l_{IPW}(S, q) = \sum_{x_i \in \pi_q, c_i = 1} \frac{\Delta(x_i, c_i|\pi_q)}{P(o_i = 1|\pi_q)}$$

How to estimate examination propensity without result randomization?
Re-Visit the Problem

- Step back a little bit:

\[
P(c_i = 1) = P(o_i = 1) \cdot P(r_i = 1)
\]

Inverse Propensity Weighting
Unbiased Propensity Estimation

• Goal:
  – Estimate the true propensity model $E$

**Propensity loss based on clicks:**

$$\hat{l}(E, q) = \sum_{x_i \in \pi_q, c_i = 1} \Delta(x_i, c_i | \pi_q)$$

**Propensity loss based on examination labels:**

$$l(E, q) = \sum_{x_i \in \pi_q, o_i = 1} \Delta(x_i, o_i | \pi_q)$$
Unbiased Propensity Estimation

• Inverse Relevance Weighting:

\[ l_{IRW}(E, q) = \sum_{x_i \in \pi_q, c_i=1} \frac{\Delta(x_i, c_i|\pi_q)}{P(r_i = 1|\pi_q)} \]

Propensity loss based on clicks

\[ l(E, q) = \sum_{x_i \in \pi_q, o_i=1} \Delta(x_i, o_i|\pi_q) \]

Propensity loss based on observation labels

\[ \mathbb{E}_{r_q}[l_{IRW}(E, q)] = \mathbb{E}_{r_q} \left[ \sum_{x_i \in \pi_q, o_i=1, r_i=1} \frac{\Delta(x_i, o_i|\pi_q)}{P(r_i = 1|\pi_q)} \right] \]

\[ = \mathbb{E}_{r_q} \left[ \sum_{x_i \in \pi_q, o_i=1} r_i \cdot \frac{\Delta(x_i, o_i|\pi_q)}{P(r_i = 1|\pi_q)} \right] \]

\[ = \sum_{x_i \in \pi_q, o_i=1} \mathbb{E}_{r_q}[r_i] \cdot \frac{\Delta(x_i, o_i|\pi_q)}{P(r_i = 1|\pi_q)} \]

\[ = \sum_{x_i \in \pi_q, o_i=1} P(r_i = 1|\pi_q) \cdot \frac{\Delta(x_i, o_i|\pi_q)}{P(r_i = 1|\pi_q)} \]

\[ = \sum_{x_i \in \pi_q, o_i=1} \Delta(x_i, o_i|\pi_q) \]

\[ c_i=1 \Rightarrow o_i=1, r_i=1 \]
Dual Learning Algorithm (DLA)

- Jointly learn the propensity model and the ranking model:

\[
l_{IRW}(E, q) = \sum_{x_i \in \pi_q, c_i=1} \Delta(x_i, c_i|\pi_q) \frac{P(r_i = 1|\pi_q)}{P(o_i = 1|\pi_q)}
\]

Propensity Model

\[
l_{IPW}(S, q) = \sum_{x_i \in \pi_q, c_i=1} \Delta(x_i, c_i|\pi_q) \frac{P(o_i = 1|\pi_q)}{P(r_i = 1|\pi_q)}
\]

Ranking Model
Dual Learning Algorithm

- Examination/relevance probability:
  \[ P_E(o_i = 1| \pi_q) = \frac{e^{g_i(\phi)}}{\sum_{z \in \pi_q} e^{g_z(\phi)}} \]
  \[ P_S(r_i = 1| \pi_q) = \frac{e^{f_i(\theta)}}{\sum_{z \in \pi_q} e^{f_z(\theta)}} \]

- Propensity/ranking loss:
  \[ l_{IRW}(E, q) = \sum_{x_i \in \pi_q, c_i = 1} \frac{P_S(r_1 = 1| \pi_q)}{P_S(r_i = 1| \pi_q)} \cdot \Delta(x_i, o_i| \pi_q) = -\sum_{x_i \in \pi_q, c_i = 1} \frac{P_S(r_1 = 1| \pi_q)}{P_S(r_i = 1| \pi_q)} \cdot \log \frac{e^{g_i(\phi)}}{\sum_{z \in \pi_q} e^{g_z(\phi)}} \]
  \[ l_{IPW}(S, q) = \sum_{x_i \in \pi_q, c_i = 1} \frac{P_E(o_1 = 1| \pi_q)}{P_E(o_i = 1| \pi_q)} \cdot \Delta(x_i, c_i| \pi_q) = -\sum_{x_i \in \pi_q, c_i = 1} \frac{P_E(o_1 = 1| \pi_q)}{P_E(o_i = 1| \pi_q)} \cdot \log \frac{e^{f_i(\theta)}}{\sum_{z \in \pi_q} e^{f_z(\theta)}} \]

- Model implementation:
Experiments

• Research questions:

RQ1. Can DLA effectively estimate the true presentation bias and produce an unbiased ranker at the same time?

RQ2. Compared to the methodology that debias click data and trains learning-to-rank models separately, are there any benefits from the joint learning of rankers and examination propensity models?
Simulation Experiments

RQ1. Can DLA effectively estimate the true presentation bias and produce an unbiased ranker at the same time?

- Dataset
  - Yahoo! LETOR (29,921 queries, 710k docs)
  - Train models with clicks, but test models with human annotations.
  - Simulate clicks based on presentation bias [7, 9]:

\[
P(c_i = 1|\pi_q) = P(o_i = 1) \cdot P(r_i = 1|\pi_q)
\]
Simulation Experiments

RQ1. Can DLA effectively estimate the true presentation bias and produce an unbiased ranker at the same time?

- Baselines
  - NoCorrect:
    - Train LTR models with clicks directly.
  - RandList [7, 8]:
    - Estimate examination propensity with randomization experiments.
    - Train LTR models with clicks and inverse propensity weighting.
  - Initial Ranker
    - The model that generates the initial ranked lists for click simulation.
  - Oracle DNN
    - The model trained with human annotations (the best possible model).
## Simulation Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN with DLA</td>
<td></td>
<td>0.658</td>
<td>0.338</td>
<td>0.683</td>
<td>0.433</td>
<td>0.729</td>
<td>0.447</td>
</tr>
<tr>
<td>SVMrank</td>
<td>NoCorrect</td>
<td>0.628</td>
<td>0.316</td>
<td>0.661</td>
<td>0.416</td>
<td>0.711</td>
<td>0.432</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.642</td>
<td>0.330</td>
<td>0.675</td>
<td>0.428</td>
<td>0.721</td>
<td>0.442</td>
</tr>
<tr>
<td>DNN</td>
<td>NoCorrect</td>
<td>0.622</td>
<td>0.317</td>
<td>0.653</td>
<td>0.416</td>
<td>0.704</td>
<td>0.431</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td><strong>0.658</strong></td>
<td><strong>0.338</strong></td>
<td>0.679</td>
<td><strong>0.433</strong></td>
<td>0.725</td>
<td><strong>0.447</strong></td>
</tr>
<tr>
<td>Initial Ranker</td>
<td></td>
<td>0.559</td>
<td>0.271</td>
<td>0.617</td>
<td>0.381</td>
<td>0.675</td>
<td>0.397</td>
</tr>
<tr>
<td>Oracle DNN</td>
<td></td>
<td>0.667</td>
<td>0.339</td>
<td>0.695</td>
<td>0.435</td>
<td>0.740</td>
<td>0.449</td>
</tr>
</tbody>
</table>

**Static mode:**
- Click bias doesn’t change over time.
- In this case, RandList is an optimal propensity estimator in theory.
Simulation Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN with DLA</td>
<td></td>
<td>0.658</td>
<td>0.338</td>
<td>0.683</td>
<td>0.433</td>
<td>0.729</td>
<td>0.447</td>
</tr>
<tr>
<td>SVMrank</td>
<td>NoCorrect</td>
<td>0.628</td>
<td>-</td>
<td>0.661</td>
<td>-</td>
<td>0.711</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.642</td>
<td>0.330</td>
<td>0.675</td>
<td>0.428</td>
<td>0.721</td>
<td>0.442</td>
</tr>
<tr>
<td>DNN</td>
<td>NoCorrect</td>
<td>0.622</td>
<td>0.317</td>
<td>0.653</td>
<td>0.416</td>
<td>0.704</td>
<td>0.431</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.658</td>
<td>0.338</td>
<td>0.679</td>
<td>0.433</td>
<td>0.725</td>
<td>0.447</td>
</tr>
<tr>
<td>Initial Ranker</td>
<td></td>
<td>0.559</td>
<td>0.271</td>
<td>0.617</td>
<td>0.381</td>
<td>0.675</td>
<td>0.397</td>
</tr>
<tr>
<td>Oracle DNN</td>
<td></td>
<td>0.667</td>
<td>0.339</td>
<td>0.695</td>
<td>0.435</td>
<td>0.740</td>
<td>0.449</td>
</tr>
</tbody>
</table>

• Observation:
  – Click signals are useful for learning to rank.
## Simulation Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN with DLA</td>
<td>NoCorrect</td>
<td>0.658</td>
<td>0.338</td>
<td>0.683</td>
<td>0.433</td>
<td>0.729</td>
<td>0.447</td>
</tr>
<tr>
<td>DNN</td>
<td>NoCorrect</td>
<td>0.628</td>
<td>0.316</td>
<td>0.661</td>
<td>0.416</td>
<td>0.711</td>
<td>0.432</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.642</td>
<td>0.330</td>
<td>0.675</td>
<td>0.428</td>
<td>0.721</td>
<td>0.442</td>
</tr>
<tr>
<td>DNN</td>
<td>NoCorrect</td>
<td>0.622</td>
<td>0.317</td>
<td>0.653</td>
<td>0.416</td>
<td>0.704</td>
<td>0.431</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.658</td>
<td>0.338</td>
<td>0.679</td>
<td>0.433</td>
<td>0.725</td>
<td>0.447</td>
</tr>
<tr>
<td>Initial Ranker</td>
<td></td>
<td>0.559</td>
<td>0.271</td>
<td>0.617</td>
<td>0.381</td>
<td>0.675</td>
<td>0.397</td>
</tr>
<tr>
<td>Oracle DNN</td>
<td></td>
<td>0.667</td>
<td>0.339</td>
<td>0.695</td>
<td>0.435</td>
<td>0.740</td>
<td>0.449</td>
</tr>
</tbody>
</table>

- **Observation:**
  - Click signals are useful for learning to rank.
  - Inverse propensity weighting with RandList is beneficial.
## Simulation Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN with DLA</td>
<td></td>
<td>0.658</td>
<td>0.338</td>
<td>0.683</td>
<td>0.433</td>
<td>0.729</td>
<td>0.447</td>
</tr>
<tr>
<td>SVMrank</td>
<td>NoCorrect</td>
<td>0.628</td>
<td>0.316</td>
<td>0.661</td>
<td>0.416</td>
<td>0.711</td>
<td>0.432</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.642</td>
<td>0.330</td>
<td>0.675</td>
<td>0.428</td>
<td>0.721</td>
<td>0.442</td>
</tr>
<tr>
<td>DNN</td>
<td>NoCorrect</td>
<td>0.622</td>
<td>0.317</td>
<td>0.653</td>
<td>0.416</td>
<td>0.704</td>
<td>0.431</td>
</tr>
<tr>
<td></td>
<td>RandList</td>
<td>0.658</td>
<td>0.338</td>
<td>0.679</td>
<td>0.433</td>
<td>0.725</td>
<td>0.447</td>
</tr>
<tr>
<td>Initial Ranker</td>
<td></td>
<td>0.559</td>
<td>0.271</td>
<td>0.617</td>
<td>0.381</td>
<td>0.675</td>
<td>0.397</td>
</tr>
<tr>
<td>Oracle DNN</td>
<td></td>
<td>0.667</td>
<td>0.339</td>
<td>0.695</td>
<td>0.435</td>
<td>0.740</td>
<td>0.449</td>
</tr>
</tbody>
</table>

- **Observation:**
  - Click signals are useful for learning to rank.
  - Inverse propensity weighting with RandList is beneficial
  - DLA is as effective as RandList.
Simulation Experiments

Figure 1: Ranking performance

- Dynamic mode:
  - Click bias changes over time.
  - The presentation bias in the randomization experiment ($\eta=1$) could be different from those in model training ($\eta \neq 1$).
Simulation Experiments

Figure 1: Ranking performance

Figure 2: Propensity estimation performance

• Observation:
  – The change of click bias severely hurts the performance of RandList.
  – DLA is much more robust and adaptive.

No result randomization needed!
RQ2. Compared to the methodology that **debias click data and trains learning-to-rank models separately**, are there any benefits from **the joint learning of rankers and propensity models**?

- **Dataset**
  - Search logs from a commercial search engine (3,449 queries).
  - Real user clicks (> 3M).
  - LETOR style features [10].
  - A separate test collection with 100 queries and 10k docs annotated by human [11].
RQ2. Compared to the methodology that debias click data and trains learning-to-rank models separately, are there any benefits from the joint learning of rankers and propensity models?

- **Baselines:**
  - **NoCorrect:**
    - Train LTR models with click directly.
  - **UBM:**
    - Train LTR models with the relevance signals extracted by UBM [12].
  - **DBN:**
    - Train LTR models with the relevance signals extracted by DBN [13].
Real-world Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>DLA</td>
<td>0.433</td>
<td>0.406</td>
<td>0.422</td>
<td>0.571</td>
<td>0.421</td>
<td>0.582</td>
</tr>
<tr>
<td></td>
<td>DBN</td>
<td>0.363</td>
<td>0.340</td>
<td>-</td>
<td>0.390</td>
<td>0.504</td>
<td>0.419</td>
</tr>
<tr>
<td></td>
<td>UBM</td>
<td>0.359</td>
<td>0.336</td>
<td>-</td>
<td>0.352</td>
<td>0.502</td>
<td>0.365</td>
</tr>
<tr>
<td></td>
<td>NoCorrect</td>
<td>0.357</td>
<td>0.334</td>
<td>-</td>
<td>0.349</td>
<td>0.484</td>
<td>0.358</td>
</tr>
</tbody>
</table>

- Observations:
## Real-world Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>DLA</td>
<td>0.433</td>
<td>0.406</td>
<td>0.422</td>
<td>0.571</td>
<td>0.421</td>
<td>0.582</td>
</tr>
<tr>
<td>DNN</td>
<td>DBN</td>
<td>0.363</td>
<td>0.340</td>
<td>0.390</td>
<td>0.504</td>
<td>0.419</td>
<td>0.521</td>
</tr>
<tr>
<td>DNN</td>
<td>UBM</td>
<td>0.359</td>
<td>0.336</td>
<td>0.352</td>
<td>0.502</td>
<td>0.365</td>
<td>0.519</td>
</tr>
<tr>
<td>DNN</td>
<td>NoCorrect</td>
<td>0.357</td>
<td>0.334</td>
<td>0.349</td>
<td>0.484</td>
<td>0.358</td>
<td>0.500</td>
</tr>
</tbody>
</table>

- **Observations:**
  - Click models $>$ NoCorrect.
## Real-world Experiments

<table>
<thead>
<tr>
<th>Ranking Model</th>
<th>Correction Method</th>
<th>NDCG@1</th>
<th>ERR@1</th>
<th>NDCG@5</th>
<th>ERR@5</th>
<th>NDCG@10</th>
<th>ERR@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>DLA</td>
<td>0.433</td>
<td>0.406</td>
<td>0.422</td>
<td>0.571</td>
<td>0.421</td>
<td>0.582</td>
</tr>
<tr>
<td></td>
<td>DBN</td>
<td>0.363</td>
<td>0.340</td>
<td>-</td>
<td>0.390</td>
<td>-</td>
<td>0.504</td>
</tr>
<tr>
<td></td>
<td>UBM</td>
<td>0.359</td>
<td>0.336</td>
<td>-</td>
<td>0.352</td>
<td>-</td>
<td>0.502</td>
</tr>
<tr>
<td></td>
<td>NoCorrect</td>
<td>0.357</td>
<td>0.334</td>
<td>-</td>
<td>0.349</td>
<td>-</td>
<td>0.484</td>
</tr>
</tbody>
</table>

- Observations:
  - Click models > NoCorrect.
  - Joint learning > separate learning.
Summary

• We formulate a problem of *Unbiased Propensity Estimation* and discuss its relationship with unbiased learning to rank.

• We propose a *Dual Learning Algorithm* for unbiased learning to rank.
  – No *offline parameter estimation* or *online result randomization*.
  – End-to-end and *adaptive to user behavior changes*.

• We demonstrate the effectiveness of the *joint learning paradigm* for unbiased propensity estimation and learning to rank, both theoretically and empirically.
Thanks!

Q&A

aiqy@cs.umass.edu
http://www.cs.umass.edu/~aiqy/
https://github.com/QingyaoAi/Dual-Learning-Algorithm-for-Unbiased-Learning-to-Rank
Reference