Counterfactual Learning to Rank from User Interactions

Harrie Oosterhuis, Rolf Jagerman

June 17, 2020

University of Amsterdam

oosterhuis@uva.nl, rolf.jagerman@uva.nl

Based on the WWW'20 tutorial: *Unbiased Learning to Rank: Counterfactual and Online Approaches* (Harrie Oosterhuis, Rolf Jagerman, and Maarten de Rijke).

Who are we?

Harrie Oosterhuis
PhD student at
U. Amsterdam

Rolf Jagerman
PhD student at
U. Amsterdam

Maarten de Rijke
Professor at
U. Amsterdam

Introduction

Learning to Rank

Learning to Rank (LTR) is:

"... the task to automatically construct a ranking model using training data, such that the model can sort new objects according to their degrees of relevance, preference, or importance."

— Liu et al. (2009)

Learning to Rank is a **core task** in informational retrieval:

- Key component for **search**, **recommendation**, and **digital assistants**.

Learning to Rank: Problem Definition

The **ranking** R of **ranker** f_θ over a document set D is:

$$R = (R_1, R_2, R_3, \ldots)$$

where documents are **ordered** by their (descending) **scores**:

$$f_\theta(R_1) \geq f_\theta(R_2) \geq f_\theta(R_3) \geq \ldots,$$

and **every document** is in the ranking:

$$d \in D \iff d \in R.$$

Learning to Rank: Problem Definition

For this tutorial, we will cast **the goal of LTR** as:

- Find the **parameters** θ for the **model** f_θ,
  where sorting documents d according to their scores $f_\theta(d)$
  results in the **most optimal rankings**.

We will later define what is *optimal* according to well-known ranking metrics.

Limitations of Annotated Datasets

Learning to Rank from Annotated Datasets

Traditionally, learning to rank is **supervised** through **annotated datasets**:

- **Relevance annotations** for query-document pairs provided by **human judges**.

However, over time **several limitations** of this approach have become apparent.

Limitations of the Annotated Datasets

Some of the most substantial limitations of **annotated datasets** are:

- **expensive** to make (Qin and Liu, 2013; Chapelle and Chang, 2011).
- **unethical** to create in **privacy-sensitive settings** (Wang et al., 2016).
- **impossible** for small scale problems, e.g., **personalization**.
- **stationary**, cannot capture **future changes in relevancy** (Lefortier et al., 2014).
- **not necessarily aligned with actual user preferences** (Sanderson, 2010),

  i.e., annotators and users often disagree.

Limitations of the Supervised Approach

Annotated datasets are **valuable** and have an **important place in research and development**.

However, the supervised approach is:

- **Unavailable** for practitioners without a **considerable budget**.
- **Impossible** for certain ranking problems.
- Often **misaligned** with *true* user preferences.

Therefore, there is a **need** for an **alternative** learning to rank approach.

Learning from User Interactions

Learning from User Interactions: Advantages

Learning from user interactions solves the problems of annotations:

- Interactions are **virtually free** if you have users.
- User **behavior** is indicative of their **preferences**.

User interactions also bring their **own difficulties**:

- Interactions give **implicit feedback**.

Learning from User Interactions: Difficulties

User interactions bring their **own difficulties**:

- **Noise:**
  - Users click for **unexpected reasons**.
  - Often clicks occur **not because** of relevancy.
  - Often clicks do not occur **despite** of relevancy.

- **Bias:** Interactions are affected by **factors other than relevancy**:
  - **Position bias:** **Higher ranked** documents get more attention.
  - **Item selection bias:** Interactions are **limited** to the **presented** documents.
  - **Presentation bias:** Results that are **presented differently** will be **treated differently**.
  - ...

The Golden Triangle

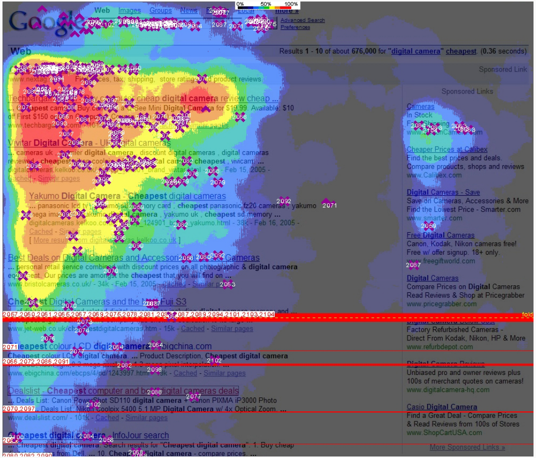


Image source: `http://www.mediative.com/`

Learning from User Interactions: Goal

Goal of unbiased learning to rank:

- Optimize a ranker w.r.t. **relevance preferences** of users from their interactions.
- **Avoid** being **biased by other factors** that influence interactions.

Counterfactual Learning to Rank

The remainder of this talk will cover the following topics:

- **Counterfactual Evaluation**
    - Evaluating unbiasedly from historical interactions.
- **Propensity-weighted LTR**
    - Learning unbiasedly from historical interactions.
- **Estimating Position Bias**
- **Practical Considerations**

Counterfactual Evaluation

Counterfactual Evaluation: Introduction

Evaluation is incredibly **important before deploying** a ranking system.

However, with the **limitations of annotated datasets**,
can we **evaluate** a ranker **without deploying** it or **annotated data**?

Counterfactual Evaluation:
Evaluate a new ranking function f_θ using **historical interaction data** (e.g., clicks) collected from a previously deployed ranking function f_{deploy}.

Counterfactual Evaluation: Full Information

If we **know** the **true relevance labels** ($y(d_i)$ for all i), we can compute any additive linearly decomposable IR metric as:

$$\Delta(f_\theta, D, y) = \sum_{d_i \in D} \lambda(\mathit{rank}(d_i \mid f_\theta, D)) \cdot y(d_i),$$

where λ is a rank weighting function, e.g.,

$$\begin{aligned}
\text{Average Relevant Position} \quad & \mathit{ARP} : \lambda(r) = r, \\
\text{Discounted Cumulative Gain} \quad & \mathit{DCG} : \lambda(r) = \frac{1}{\log_2(1 + r)}, \\
\text{Precision at } k \quad & \mathit{Prec@}k : \lambda(r) = \frac{\mathbf{1}[r \leq k]}{k}.
\end{aligned}$$

Counterfactual Evaluation: Full Information

$y(d_1) = 1$	Document d_1
$y(d_2) = 0$	Document d_2
$y(d_3) = 0$	Document d_3
$y(d_4) = 1$	Document d_4
$y(d_5) = 0$	Document d_5

Counterfactual Evaluation: Partial Information

We often do not know the true relevance labels $y(d_i)$, but can only observe implicit feedback in the form of, e.g., clicks:

- A click c_i on document d_i is a **biased and noisy indicator** that d_i is relevant
- A missing click does **not** necessarily indicate non-relevance.

Counterfactual Evaluation: Clicks

$y(d_1) = 1$	Document d_1	$c_1 = 1$
$y(d_2) = 0$	Document d_2	$c_2 = 0$
$y(d_3) = 0$	Document d_3	$c_3 = 1$
$y(d_4) = 1$	Document d_4	$c_4 = 0$
$y(d_5) = 0$	Document d_5	$c_5 = 0$

Counterfactual Evaluation: Clicks

Remember that there are many reasons why a click on a document may **not** occur:

- **Relevance**: the document may not be relevant.
- **Observance**: the user may not have examined the document.
- **Miscellaneous**: various random reasons why a user may not click.

Some of these reasons are considered to be:

- **Noise**: averaging over many clicks will remove their effect.
- **Bias**: averaging will **not** remove their effect.

Counterfactual Evaluation: Examination User Model

If we **only** consider **examination** and **relevance**, a user click can be modelled by:

- The probability of document d_i **being examined** $(o_i = 1)$ in a ranking R:

$$P(o_i = 1 \mid R, d_i).$$

- The probability of a **click** $c_i = 1$ on d_i given its **relevance** $y(d_i))$ and whether it was **examined** o_i:

$$P(c_i = 1 \mid o_i, y(d_i)).$$

- **Clicks only occur on examined documents**, thus the probability of a click in ranking R is:

$$P(c_i = 1 \wedge o_i = 1 \mid y(d_i), R) = P(c_i = 1 \mid o_i = 1, y(d_i)) \cdot P(o_i = 1 \mid R, d_i).$$

Counterfactual Evaluation: Naive Estimator

A **naive way** to estimate is to assume clicks are a unbiased relevance signal:

$$\hat{\Delta}_{\mathit{NAIVE}}(f_\theta, D, c) = \sum_{d_i \in D} \lambda(\mathit{rank}(d_i \mid f_\theta, D)) \cdot c_i.$$

Even if **no click noise** is present: $P(c_i = 1 \mid o_i = 1, y(d_i)) = y(d_i)$, this estimator is **biased** by the observation probabilities:

$$\begin{aligned}
\mathbb{E}_o[\hat{\Delta}_{\mathit{NAIVE}}(f_\theta, D, c)] &= \mathbb{E}_o\left[\sum_{d_i : o_i = 1 \wedge y(d_i) = 1} \lambda(\mathit{rank}(d_i \mid f_\theta, D))\right] \\
&= \sum_{d_i : y(d_i) = 1} P(o_i = 1 \mid R, d_i) \cdot \lambda(\mathit{rank}(d_i \mid f_\theta, D)).
\end{aligned}$$

Counterfactual Evaluation: Naive Estimator Bias

The biased estimator **weights documents** according to their **observation probabilities** in the ranking R displayed during **logging**:

$$\mathbb{E}_o[\hat{\Delta}_{\mathit{NAIVE}}(f_\theta, D, c)] = \sum_{d_i : y(d_i)=1} P(o_i = 1 \mid R, d_i) \cdot \lambda(\mathit{rank}(d_i \mid f_\theta, D)).$$

In rankings, **documents at higher ranks** are more likely to be examined: **position bias**.

Position bias causes **logging-policy-confirming** behavior:

- Documents displayed at **higher ranks during logging** are incorrectly considered as **more relevant**.

Inverse Propensity Scoring

Counterfactual Evaluation: Inverse Propensity Scoring

Inverse Propensity Scoring (IPS) estimators can remove bias:

- First introduced by Wang et al. (2016) and Joachims et al. (2017).
- **Main idea**: weight clicks depending on their *observation probability*
- Clicks near the **top** of the ranked list:
    - Have **high** observation probability $\Leftrightarrow$ Get assigned **small** weight
- Clicks near the **bottom** of the ranked list:
    - Have **low** observation probability $\Leftrightarrow$ Get assigned **large** weight

Counterfactual Evaluation: Inverse Propensity Scoring

Counterfactual evaluation accounts for bias using **Inverse Propensity Scoring (IPS)**:

$$\hat{\Delta}_{\mathit{IPS}}(f_\theta, D, c) = \sum_{d_i \in D} \frac{\lambda(\mathit{rank}(d_i \mid f_\theta, D))}{P(o_i = 1 \mid R, d_i)} \cdot c_i,$$

where

- $\lambda(\mathit{rank}(d_i \mid f_\theta, D))$: (weighted) rank of document d_i by ranker f_θ,
- c_i: observed click on the document in the log,
- $P(o_i = 1 \mid R, d_i)$: observation probability of d_i in ranking R displayed during logging.

This is an **unbiased estimate** of any additive linearly decomposable IR metric.

Counterfactual Evaluation: Proof of Unbiasedness

If no click noise is present, this provides an **unbiased estimate**:

$$
\begin{aligned}
\mathbb{E}_o[\hat{\Delta}_{\mathit{IPS}}(f_\theta, D, c)] &= \mathbb{E}_o\left[\sum_{d_i \in D} \frac{\lambda(\mathit{rank}(d_i \mid f_\theta, D))}{P(o_i = 1 \mid R, d_i)} \cdot c_i\right] \\
&= \mathbb{E}_o\left[\sum_{d_i : o_i = 1 \wedge y(d_i) = 1} \frac{\lambda(\mathit{rank}(d_i \mid f_\theta, D))}{P(o_i = 1 \mid R, d_i)}\right] \\
&= \sum_{d_i : y(d_i) = 1} \frac{P(o_i = 1 \mid R, d_i) \cdot \lambda(\mathit{rank}(d_i \mid f_\theta, D))}{P(o_i = 1 \mid R, d_i)} \\
&= \sum_{d_i \in D} \lambda(\mathit{rank}(d_i \mid f_\theta, D)) \cdot y(d_i) \\
&= \Delta(f_\theta, D, y).
\end{aligned}
$$

Counterfactual Evaluation: Robustness of Noise

So far we have **no click noise**: $P(c_i = 1 \mid o_i = 1, y(d_i)) = y(d_i)$.

However, the IPS approach still works without these assumptions, as long as:

$$y(d_i) > y(d_j) \Leftrightarrow P(c_i = 1 \mid o_i = 1, y(d_i)) > P(c_j = 1 \mid o_j = 1, y(d_j)).$$

Since we can prove **relative differences** are inferred unbiasedly:

$$\mathbb{E}_{o,c}[\hat{\Delta}_{\mathit{IPS}}(f_\theta, D, c)] > \mathbb{E}_{o,c}[\hat{\Delta}_{\mathit{IPS}}(f_{\theta'}, D, c)] \Leftrightarrow \Delta(f_\theta, D) > \Delta(f_{\theta'}, D).$$

Propensity-weighted Learning to Rank

Propensity-weighted Learning to Rank (LTR)

The inverse-propensity-scored estimator can unbiasedly estimate performance:

$$\hat{\Delta}_{IPS}(f_\theta, D, c) = \sum_{d_i \in D} \frac{\lambda(\mathit{rank}(d_i \mid f_\theta, D))}{P(o_i = 1 \mid R, d_i)} \cdot c_i.$$

How do we **optimize** for this **unbiased performance estimate**?

- It is **not differentiable**.
- **Common problem for all ranking metrics**.

Upper Bound on Rank

Rank-SVM (Joachims, 2002) optimizes the following **differentiable upper bound**:

$$
\begin{aligned}
\mathit{rank}(d \mid f_\theta, D) &= \sum_{d' \in R} \mathbb{1}[f_\theta(d) \leq f_\theta(d')] \\
&\leq \sum_{d' \in R} \max(1 - (f_\theta(d) - f_\theta(d')), 0) = \overline{\mathit{rank}}(d \mid f_\theta, D).
\end{aligned}
$$

Alternative choices are possible, i.e., a **sigmoid-like bound** (with parameter σ):

$$
\mathit{rank}(d \mid f_\theta, D) \leq \sum_{d' \in R} \log_2(1 + \exp^{-\sigma(f_\theta(d) - f_\theta(d'))}).
$$

Commonly used for pairwise learning, LambdaMart (Burges, 2010), and Lambdaloss (Wang et al., 2018b).

Propensity-weighted LTR: Average Relevance Position

Then for the Average Relevance Position metric:

$$\Delta_{\mathit{ARP}}(f_\theta, D, y) = \sum_{d_i \in D} \mathit{rank}(d_i \mid f_\theta, D) \cdot y(d_i).$$

This gives us an **unbiased estimator** and **upper bound**:

$$\begin{aligned}\hat{\Delta}_{\mathit{ARP\text{-}IPS}}(f_\theta, D, c) &= \sum_{d_i \in D} \frac{\mathit{rank}(d_i \mid f_\theta, D)}{P(o_i = 1 \mid R, d_i)} \cdot c_i \\ &\leq \sum_{d_i \in D} \frac{\overline{\mathit{rank}}(d_i \mid f_\theta, D)}{P(o_i = 1 \mid R, d_i)} \cdot c_i,\end{aligned}$$

This upper bound is **differentiable** and **optimizable** by stochastic gradient descent or Quadratic Programming, i.e., Rank-SVM (Joachims, 2006).

Propensity-weighted LTR: Additive Metrics

A similar approach can be applied to **additive metrics** (Agarwal et al., 2019a).

If λ is a **monotonically decreasing** function:

$$x \leq y \Rightarrow \lambda(x) \geq \lambda(y),$$

then:

$$\mathit{rank}(d \mid \cdot) \leq \overline{\mathit{rank}}(d \mid \cdot) \Rightarrow \lambda(\mathit{rank}(d \mid \cdot)) \geq \lambda(\overline{\mathit{rank}}(d \mid \cdot)).$$

This provides a **lower bound**, for instance for Discounted Cumulative Gain (DCG):

$$\frac{1}{\log_2(1 + \mathit{rank}(d \mid \cdot))} \geq \frac{1}{\log_2(1 + \overline{\mathit{rank}}(d \mid \cdot))}.$$

Propensity-weighted LTR: Discounted Cumulative Gain

Then for the Discounted Cumulative Gain metric:

$$\Delta_{\mathit{DCG}}(f_\theta, D, y) = \sum_{d_i \in D} \log_2(1 + \mathit{rank}(d_i \mid f_\theta, D))^{-1} \cdot y(d_i).$$

This gives us an **unbiased estimator** and **lower bound**:

$$\begin{aligned}\hat{\Delta}_{\mathit{DCG\text{-}IPS}}(f_\theta, D, c) &= \sum_{d_i \in D} \frac{\log_2(1 + \mathit{rank}(d_i \mid f_\theta, D)^{-1}}{P(o_i = 1 \mid R, d_i)} \cdot c_i \\ &\geq \sum_{d_i \in D} \frac{\log_2(1 + \overline{\mathit{rank}}(d_i \mid f_\theta, D)^{-1}}{P(o_i = 1 \mid R, d_i)} \cdot c_i.\end{aligned}$$

This lower bound is **differentiable** and **optimizable** by stochastic gradient descent or the Convex-Concave Procedure (Agarwal et al., 2019a).

Propensity-weighted LTR: Walkthrough

Overview of the approach:

- Obtain a **model of position bias**.
- Acquire a **large click-log**.
- Then for every click in the log:
  - Compute the **propensity of the click**:

$$P(o_i = 1 \mid R, d_i).$$

  - Calculate the **gradient** of the **bound** on the **unbiased estimator**:

$$\nabla_\theta \left[\frac{\overline{\mathit{rank}}(d_i \mid f_\theta, D)}{P(o_i = 1 \mid R, d_i)} \right].$$

  - **Update the model** f_θ by adding/subtracting the gradient.

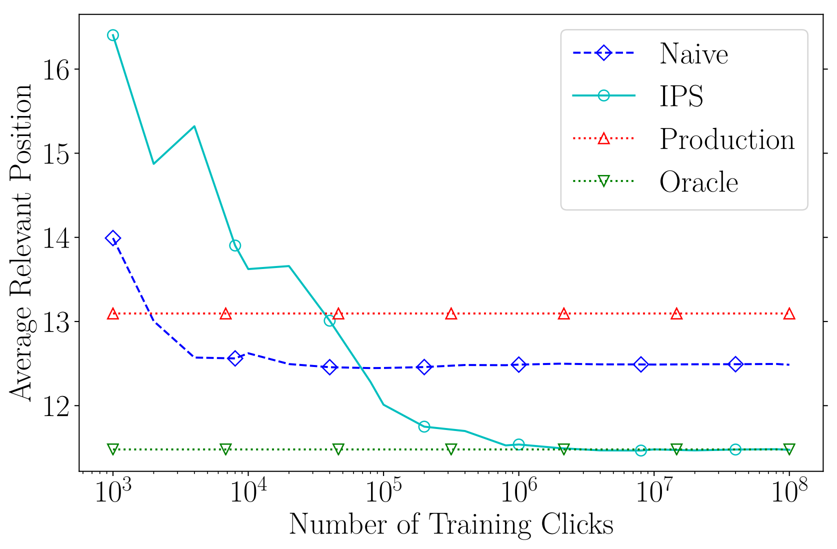
Propensity-weighted LTR: Semi-synthetic Experiments

Unbiased LTR methods are commonly **evaluated** through **semi-synthetic experiments** (Joachims, 2002; Agarwal et al., 2019a; Jagerman et al., 2019).

The experimental setup:

- Traditional LTR dataset, e.g., Yahoo! Webscope (Chapelle and Chang, 2011).
- Simulate queries by uniform sampling from the dataset.
- Create a ranking according to a baseline ranker.
- Simulate clicks by modelling:
    - **Click Noise**, e.g., 10% chance of clicking on a non-relevant document.
    - **Position Bias**, e.g., $P(o_i = 1 \mid R, d_i) = \frac{1}{\mathit{rank}(d|R)}$.
- Hyper-parameter tuning by unbiased evaluation methods.

Propensity-weighted LTR: Results



Estimating Position Bias

Estimating Position Bias

So far we have seen how to:

- Perform **Counterfactual Evaluation** with **unbiased estimators**.
- Perform **Counterfactual LTR** by optimizing **unbiased estimators**.

At the core of these methods is the propensity score: $P(o_i = 1 \mid R, d_i)$, which helps to remove bias from user interactions.

In this section, we will show how this **propensity score** can be **estimated** for a specific kind of bias: **position bias**.

Estimating Position Bias

Recall that position bias is a form of bias where higher positioned results are more likely to be observed and therefore clicked.

Assumption: The **observation probability** only depends on the rank of a document:

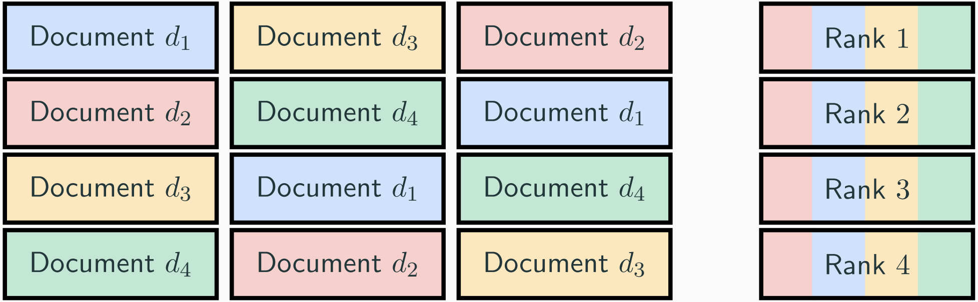
$$P(o_i = 1 \mid i).$$

The objective is now to **estimate**, for each rank i, the propensity $P(o_i = 1 \mid i)$.

This user model was first formalized by Craswell et al. (2008).

Estimating Position Bias

RandTop-n Algorithm:



Estimating Position Bias

RandTop-n Algorithm:

1. Repeat:
   - Randomly shuffle the top n items
   - Record clicks
2. Aggregate clicks per rank
3. Normalize to obtain propensities $p_i \propto P(o_i \mid i)$

Note: we only need propensities proportional to the true observation probability for learning.

Estimating Position Bias

Uniformly **randomizing** the top n results may negatively impacts users during data logging.

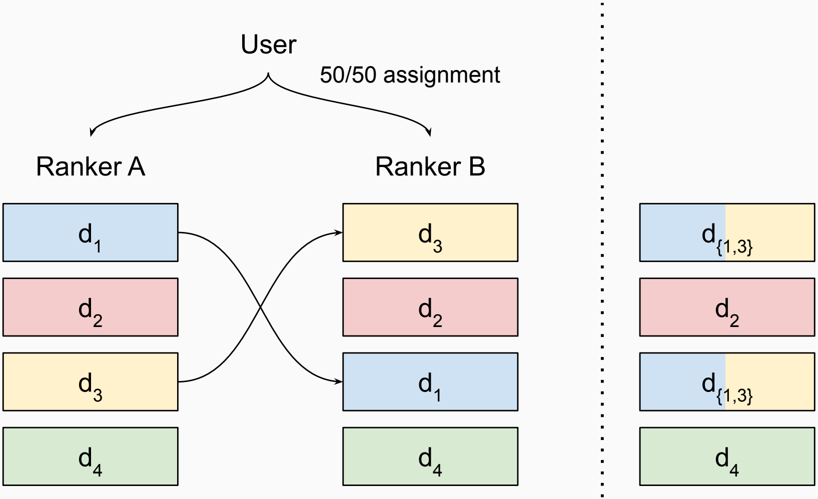
There are various methods that minimize the impact to the user:

- **RandPair:** Choose a pivot rank k and only swap a random other document with the document at this pivot rank (Joachims et al., 2017).
- **Interventional Sets:** Exploit inherent "randomness" in data coming from multiple rankers (e.g., A/B tests in production logs) (Agarwal et al., 2017).

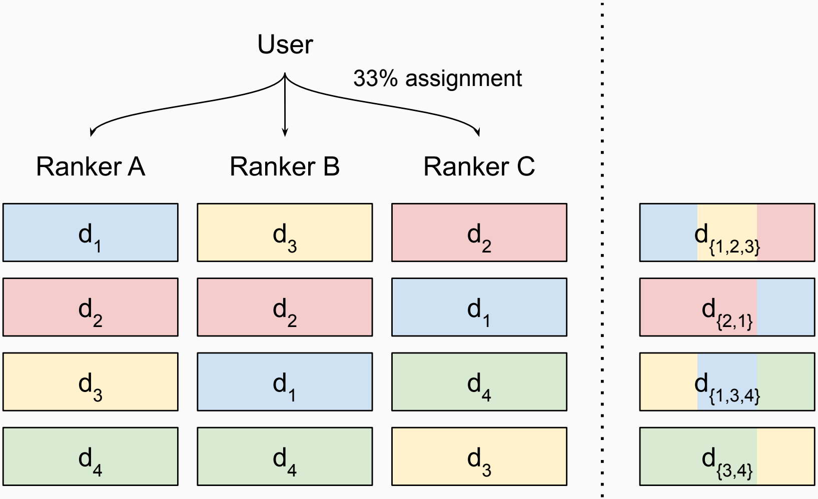
Intervention Harvesting

- As we have seen, to measure position bias, the most straightforward approach is to perform randomization.
- Naturally, we want to avoid randomizing because this negatively affects the end-user experience.
- **Main idea**: In real-world production systems many (randomized) interventions take place due to *A/B tests*. Can we use these interventions instead?
- This approach is called *intervention harvesting* (Agarwal et al. (2017); Fang et al. (2019); Agarwal et al. (2019c))

Intervention Harvesting



Intervention Harvesting



Jointly Learning and Estimating

Jointly Learning and Estimating

In the previous sections we have seen:

- Counterfactual ranker evaluation with unbiased estimators.
- Counterfactual LTR by optimizing unbiased estimators.
- Estimating propensity scores through randomization.

Instead of treating **propensity estimation** and **unbiased learning to rank** as two separate tasks, recent work has explored **jointly learning rankings and estimating propensities**.

Jointly Learning and Estimating

Recall that the probability of a click can be decomposed as:

$$\underbrace{P(c_i = 1 \wedge o_i = 1 \mid y(d_i), R)}_{\text{click probability}} = \underbrace{P(c_i = 1 \mid o_i = 1, y(d_i))}_{\text{relevance probability}} \cdot \underbrace{P(o_i \mid R, d_i)}_{\text{observation probability}}.$$

In the previous sections we have seen that, if the **observation probability** is known, we can find an unbiased estimate of relevance via IPS.

Jointly Learning and Estimating

It is possible to **jointly learn and estimate** by iterating two steps:

1. Learn an optimal ranker given a correct propensity model:

$$\underbrace{P(c_i = 1 \mid o_i = 1, y(d_i))}_{\text{relevance probability}} = \frac{P(c_i = 1 \wedge o_i = 1 \mid y(d_i), R)}{P(o_i \mid R, d_i)}.$$

2. Learn an optimal propensity model given a correct ranker:

$$\underbrace{P(o_i \mid R, d_i)}_{\text{observation probability}} = \frac{P(c_i = 1 \wedge o_i = 1 \mid y(d_i), R)}{P(c_i = 1 \mid o_i = 1, y(d_i))}.$$

Jointly Learning and Estimating

- Given an accurate **model of relevance**, it is possible to find an accurate **propensity model**, and vice versa.
- This approach requires **no randomization**.
- Recent work has solved this via either an **Expectation-Maximization approach** (Wang et al. (2018a)) or a **Dual Learning Objective** (Ai et al. (2018)).

Practical Considerations

Practical Considerations

Practitioners of counterfactual LTR systems will run into the problem of **high variance**.

High variance can be due to many factors:

- Not enough training data
- Extreme position bias and very small propensity
- Large amounts of noisy clicks on documents with small propensity

The usual suspect is one or a few data points with extremely small propensity that overpower the rest of the data set.

Practical Considerations

A typical solution to **high variance** is to apply **propensity clipping**.

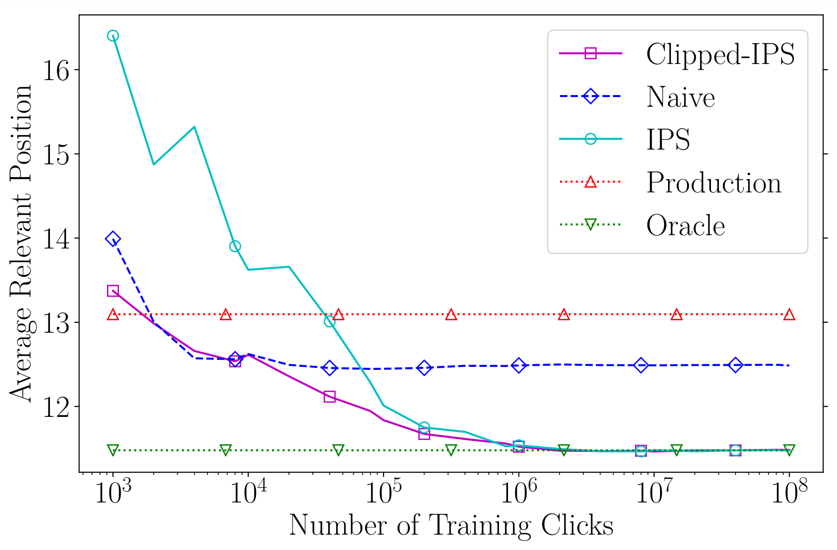
Propensity clipping: Bound the *propensity*, to prevent any single sample from overpowering the rest of the data set:

$$\hat{\Delta}_{\textit{Clipped-IPS}}(f_\theta, D, c) = \sum_{d_i \in D} \frac{\lambda(\textit{rank}(d_i \mid f_\theta, D))}{\max\{\tau, P(o_i = 1 \mid R, d_i)\}} \cdot c_i.$$

This solution trades off bias for variance: it will introduce some amount of bias but can substantially reduce variance.

Note that when $\tau = 1$, we obtain the biased naive estimator.

Practical Considerations



Comparison to Supervised LTR

Comparison to Supervised LTR

Supervised LTR:

- Uses **manually annotated labels**:
  - expensive to create,
  - impossible in many settings,
  - often misaligned with actual user preferences.
- Optimization is widely studied and very effective w.r.t. evaluation on annotated labels.
- Often unavailable for practitioners.

Counterfactual LTR:

- Uses **click logs**:
  - available in abundant quantities,
  - effectively no cost,
  - contains **noise** and **biases**.
- **Noise**: amortized over large numbers of clicks.
- **Biases**:
  - position bias mitigated with inverse propensity scoring.
  - other biases are an active area of research.

Conclusion

Conclusion

Today we discussed:

- **User interactions** with rankings are **very biased**.
- **Counterfactual Learning to Rank:**
  - Correct for position bias with inverse propensity scoring.
  - Requires an explicit user model.
- Unbiased learning from **historical** interaction logs.

Small Selection of Recent Work in the Field

The unbiased learning to rank field is very active:

- *Addressing Trust Bias for Unbiased Learning-to-Rank* (Agarwal et al., 2019b).
- *Fair Learning-to-Rank from Implicit Feedback* (Yadav et al., 2019).
- *Correcting for Selection Bias in Learning-to-rank Systems* (Ovaisi et al., 2020).
- *Policy-Aware Unbiased Learning to Rank for Top-k Rankings* (Oosterhuis and de Rijke, 2020).
- *Accelerated Convergence for Counterfactual Learning to Rank* by (Jagerman and de Rijke, 2020).

End

Thank you for your attention!

Notation

Notation Used in the Slides i

Definition	Notation	Example
Query	q	–
Candidate documents	D	–
Document	$d \in D$	–
Ranking	R	$(R_1, R_2, \ldots, R_n)$
Document at rank i	R_i	$R_i = d$
Relevance	$y : D \to \mathbb{N}$	$y(d) = 2$
Ranker model with weights θ	$f_\theta : D \to \mathbb{R}$	$f_\theta(d) = 0.75$
Click	$c_i \in \{0, 1\}$	–
Observation	$o_i \in \{0, 1\}$	–
Rank of d when f_θ ranks D	$\mathit{rank}(d \mid f_\theta, D)$	$\mathit{rank}(d \mid f_\theta, D) = 4$

Differentiable upper bound on $\textit{rank}(d, \mid f_\theta, D)$	$\overline{\textit{rank}}(d, \mid f_\theta, D)$	–
Average Relevant Position metric	*ARP*	–
Discounted Cumulative Gain metric	*DCG*	–
Precision at k metric	$\textit{Prec}@k$	–
A performance measure or estimator	Δ	–

Resources i

- Tensorflow Learning to Rank, allows for inverse propensity scoring:
  `https://github.com/tensorflow/ranking`
- Inverse Propensity Scored Rank-SVM:
  `https://www.cs.cornell.edu/people/tj/svm_light/svm_proprank.html`
- Pairwise Differentiable Gradient Descent and Multileave Gradient Descent:
  `https://github.com/HarrieO/OnlineLearningToRank`
- Data and code for comparing counterfactual and online learning to rank
  `http://github.com/rjagerman/sigir2019-user-interactions`
- An older online learning to rank framework: Lerot
  `https://bitbucket.org/ilps/lerot/`

References i

A. Agarwal, S. Basu, T. Schnabel, and T. Joachims. Effective evaluation using logged bandit feedback from multiple loggers. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 687–696. ACM, 2017.

A. Agarwal, K. Takatsu, I. Zaitsev, and T. Joachims. A general framework for counterfactual learning-to-rank. In *42nd International ACM SIGIR Conference on Research & Development in Information Retrieval*, page (to appear). ACM, 2019a.

A. Agarwal, X. Wang, C. Li, M. Bendersky, and M. Najork. Addressing trust bias for unbiased learning-to-rank. In *The World Wide Web Conference*, pages 4–14. ACM, 2019b.

A. Agarwal, I. Zaitsev, X. Wang, C. Li, M. Najork, and T. Joachims. Estimating position bias without intrusive interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 474–482, 2019c.

Q. Ai, K. Bi, C. Luo, J. Guo, and W. B. Croft. Unbiased learning to rank with unbiased propensity estimation. In *Proceedings of the 41st International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 385–394. ACM, 2018.

C. J. Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81, 2010.

O. Chapelle and Y. Chang. Yahoo! Learning to Rank Challenge Overview. *Journal of Machine Learning Research*, 14:1–24, 2011.

N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*, pages 87–94, 2008.

Z. Fang, A. Agarwal, and T. Joachims. Intervention harvesting for context-dependent examination-bias estimation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 825–834, 2019.

R. Jagerman and M. de Rijke. Accelerated convergence for counterfactual learning to rank. In *SIGIR*. ACM, 2020.

R. Jagerman, H. Oosterhuis, and M. de Rijke. To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions. In *42nd International ACM SIGIR Conference on Research & Development in Information Retrieval*, page (to appear). ACM, 2019.

T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

T. Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226. ACM, 2006.

T. Joachims, A. Swaminathan, and T. Schnabel. Unbiased learning-to-rank with biased feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 781–789. ACM, 2017.

D. Lefortier, P. Serdyukov, and M. de Rijke. Online exploration for detecting shifts in fresh intent. In *CIKM 2014: 23rd ACM Conference on Information and Knowledge Management*. ACM, November 2014.

T.-Y. Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

H. Oosterhuis and M. de Rijke. Policy-aware unbiased learning to rank for top-k rankings. In *SIGIR*. ACM, 2020.

Z. Ovaisi, R. Ahsan, Y. Zhang, K. Vasilaky, and E. Zheleva. Correcting for selection bias in learning-to-rank systems. *arXiv preprint arXiv:2001.11358*, 2020.

T. Qin and T.-Y. Liu. Introducing letor 4.0 datasets. *arXiv preprint arXiv:1306.2597*, 2013.

M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4(4):247–375, 2010.

X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 115–124. ACM, 2016.

X. Wang, N. Golbandi, M. Bendersky, D. Metzler, and M. Najork. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 610–618. ACM, 2018a.

X. Wang, C. Li, N. Golbandi, M. Bendersky, and M. Najork. The lambdaloss framework for ranking metric optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1313–1322. ACM, 2018b.

H. Yadav, Z. Du, and T. Joachims. Fair learning-to-rank from implicit feedback. *arXiv preprint arXiv:1911.08054*, 2019.

Acknowledgments

All content represents the opinion of the author(s), which is not necessarily shared or endorsed by their employers and/or sponsors.