Learning to Rank and Evaluation in the Online Setting - Course Introduction

Harrie Oosterhuis

August 27, 2018

University of Amsterdam

oosterhuis@uva.nl

The Online Setting

Interactive Algorithms

This course will focus on algorithms that learn from users by **directly interacting** with them, we call this learning in the **Online Setting**.

These online algorithms can learn **efficiently**: they require few user interactions; and **very responsively**: they can adapt to user behaviour almost immediately.

The Online Setting for Ranking

Search and recommendation are some of the most vital parts of many websites/products and rely heavily on **ranking**.
Learning user preferences is especially valuable in these settings.

However, most general online approaches: **Bandit or Reinforcement Learning algorithms** are not very effective in ranking settings.
As a result, there were online methods designed **specifically for ranking**.

Evaluation

Evaluation in the Online Setting

The big question of **ranker evaluation**:

- Do **users prefer** ranking model A over B?

Research and development is impossible without answering this question.

Evaluation in the Online Setting

This course will discuss:

- The **problems with offline approaches** to ranker evaluation.
- What user interactions can tell us about **their preferences**.
- **Interleaving**: learning from the user while **simultaneously helping them**.
- **Theoretical look** at online evaluation methods.
- **Multileaving**: evaluation on a **very large scale**.

Learning to Rank

Learning to Rank in the Online Setting

Most ranking models **combine hundreds of ranking signals** (features), these models are optimized with **machine learning**.

Online Learning to Rank algorithms learn ranking models while simultaneously providing a **good user experience**.
The **potential** for learning from user interactions is great, but it also brings **many difficulties**.

Learning to Rank in the Online Setting

This course will cover:

- The **limitations of offline approaches** to learning to rank.
- **Difficulties** in optimizing based on user interactions.
- **Dueling Bandit Gradient Descent** optimization using **evaluation**.
- Improvements in **gradient estimation** in the online setting.
- **Most recent** novel online approach.
- **Future directions** for online learning to rank.

Goals for this Course

Goals for this course

At the **end of this course**, you should:

- be convinced of the **importance of online methods**.
- understand the **most relevant algorithms** in online evaluation and online learning to rank.
- be capable of designing novel online methods yourself.
- apply online methods in practice.

Acknowledgments

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

Learning to Rank and Evaluation in the Online Setting - Online Evaluation

Harrie Oosterhuis

August 27, 2018

University of Amsterdam

oosterhuis@uva.nl

The slides of this presentation are available at:

`https://staff.fnwi.uva.nl/h.r.oosterhuis`

You can also click on my name in the RuSSIR program to get there.

Introduction: Ranking Systems

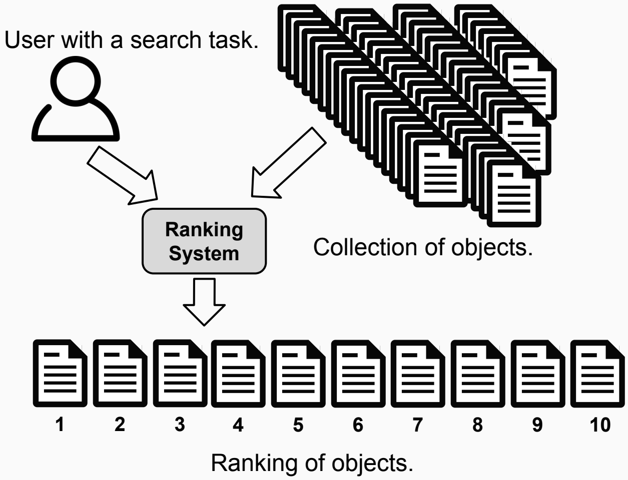
Ranking Systems

Ranking systems are vital for **making the internet accessible**.

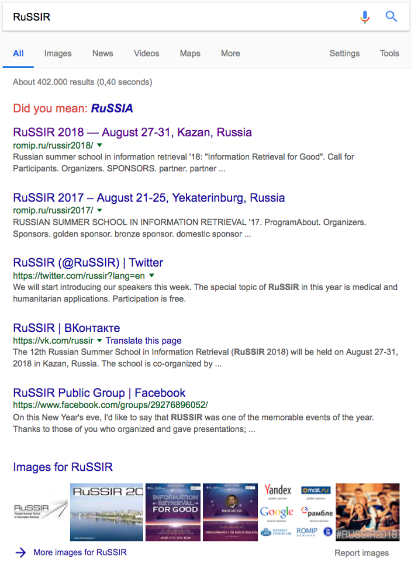
Instead of displaying **millions of unordered results**, they can present users **a small comprehensible selection**.

Applications for ranking systems are very wide, search and recommendation are **practically everywhere**.

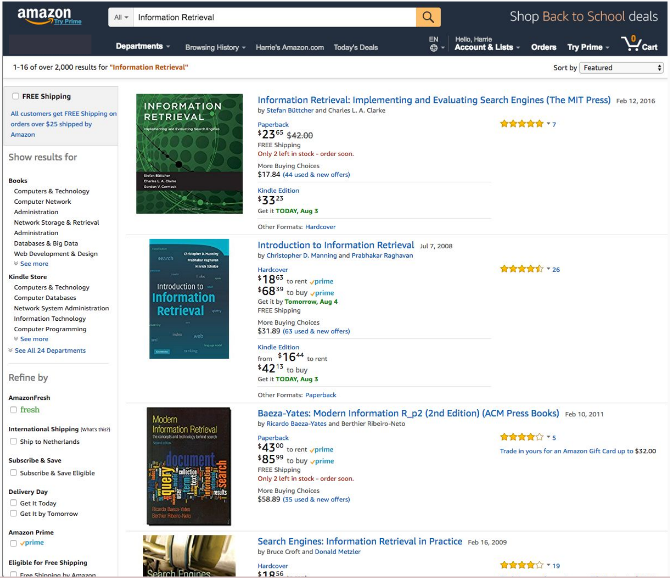
Ranking Systems: Schematic Example



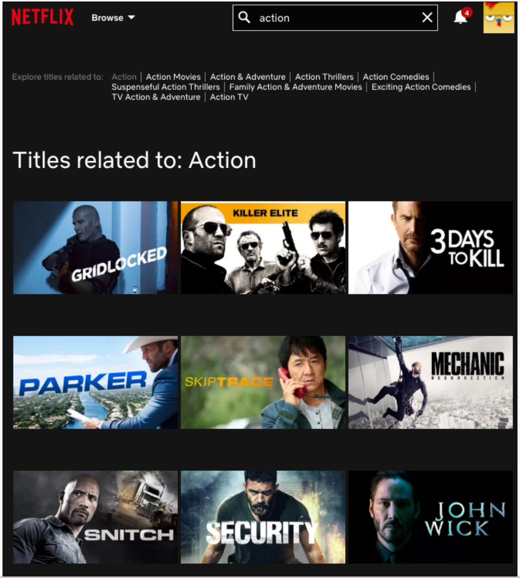
Ranking Systems: Examples



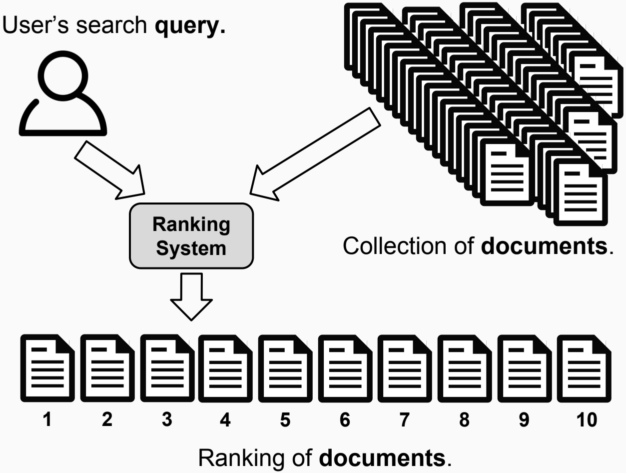
Ranking Systems: Examples



Ranking Systems: Examples



Ranking Systems: Schematic Example Naming



Importance of Evaluation

Evaluation for Ranking Systems

As ranking systems are very important, so is **their evaluation**.

Consider the following:

- You **updated the ranking system** of your product with an amazing new feature.
- How can you find out **whether you improved the system?**

Evaluation Example

You have a **great idea** for a **new ranking model!**

Evaluation Example

You have a **great idea** for a **new ranking model!**

With permission of your boss, you **implement** the model and put it in **production**.

Evaluation Example

You have a **great idea** for a **new ranking model!**

With permission of your boss, you **implement** the model and put it in **production**.

After one week in production you see **increases in ad-revenue**!
Your boss is very excited.

Evaluation Example

You have a **great idea** for a **new ranking model!**

With permission of your boss, you **implement** the model and put it in **production**.

After one week in production you see **increases in ad-revenue**!
Your boss is very excited.

Did your ranking model improve the previous model?

Evaluation Example

You have a **great idea** for a **new ranking model!**

With permission of your boss, you **implement** the model and put it in **production**.

After one week in production you see **increases in ad-revenue**!
Your boss is very excited.

Did your ranking model improve the previous model?

Possible explanations:

- Ranking quality is **much better** and people **stay longer** on your website

Evaluation Example

You have a **great idea** for a **new ranking model!**

With permission of your boss, you **implement** the model and put it in **production**.

After one week in production you see **increases in ad-revenue**!
Your boss is very excited.

Did your ranking model improve the previous model?

Possible explanations:

- Ranking quality is **much better** and people **stay longer** on your website
- Ranking quality is so **bad** that **advertisements are more relevant than results**.

Evaluation Example

You have a **great idea** for a **new ranking model!**

With permission of your boss, you **implement** the model and put it in **production**.

After one week in production you see **increases in ad-revenue**!
Your boss is very excited.

Did your ranking model improve the previous model?

Possible explanations:

- Ranking quality is **much better** and people **stay longer** on your website
- Ranking quality is so **bad** that **advertisements are more relevant than results**.
- **External factors** cause people to be more active that particular week.
  (Kohavi et al., 2013)

Evaluation for Ranking Systems

In order to improve a ranking system i.e. research and development, an evaluation method is needed that can recognize improvements:

- Is system **A** better than system **B**?

Without reliable evaluation changes to systems may have **unintentional and unexpected consequences**.

Traditional Evaluation

Evaluation Setting

Given **two rankers** A and B, for a **query** q_i, a set of **documents** D, they each produce different **rankings**:

$$A(q_i, D) = R_A^i = [d_1, d_2, \ldots d_n] \quad (1)$$

$$B(q_i, D) = R_B^i = [d_{n+1}, d_{n+2}, \ldots d_{n+m}] \quad (2)$$

According to **what metric we choose** A or B can be better, or equally good, depending on **where they place relevant documents**.

In general, rankers that place **more relevant documents higher** are considered better, (Sakai, 2007).

Evaluation Metrics: Precision

Precision: How likely is a **retrieved document to be relevant**?

$$precision@K = \frac{\textit{number of relevant results in top } K}{K} \quad (3)$$

Evaluation Metrics: Recall

Recall: How likely is a **relevant document to be retrieved**?

$$recall@K = \frac{\textit{number of relevant results in top } K}{\textit{total number of relevant documents}} \tag{4}$$

Evaluation Metrics: Normalized Discumulative Cumulative Gain

NDCG: Relevant documents at **lower ranks should weigh less**, i.e. discounted more.

Discounted Cumulative Gain:

$$DCG@K(\mathbf{R}) = \sum_{i=1}^{K} \frac{2^{\textit{relevance label of document}(\mathbf{R}[i])} - 1}{\log_2(i+1)} \tag{5}$$

Normalized Discounted Cumulative Gain:

$$NDCG@K(\mathbf{R}) = \frac{DCG@K(\mathbf{R})}{\max_{\mathbf{R}'} DCG@K(\mathbf{R}')}. \tag{6}$$

Traditional Evaluation Setting

We have two ranking systems: **A** and **B**.

Given the previous metrics, **what else do we need** to compare them?

Traditional Evaluation Requirements

In order to compute IR-metrics the **following are required**:

- The **queries** users will ask.

Traditional Evaluation Requirements

In order to compute IR-metrics the **following are required**:

- The **queries** users will ask.
- The **documents** the systems will rank.

Traditional Evaluation Requirements

In order to compute IR-metrics the **following are required**:

- The **queries** users will ask.
- The **documents** the systems will rank.
  (A large pre-selection is made for each query, to avoid ranking entire world-wide-web.)

Traditional Evaluation Requirements

In order to compute IR-metrics the **following are required**:

- The **queries** users will ask.
- The **documents** the systems will rank.
  (A large pre-selection is made for each query, to avoid ranking entire world-wide-web.)
- Which documents are **relevant** for which queries.

Traditional Evaluation Requirements

Where to get these requirements?:

- The most common **queries** can be sampled from user logs.
- A **pre-selection of documents** can be **retrieved** using the systems.
- **Human judges can annotate query-document pairs** for relevance.

Traditional Evaluation

```
How to kill a mockingbird
```

ENCYCLOPÆDIA BRITANNICA

SPOTLIGHT · DEMYSTIFIED · QUIZZES · GALLERIES · LISTS · ON THIS DAY · BIOGRAPHIES

To Kill a Mockingbird

NOVEL BY LEE

WRITTEN BY: Anna Foca

See Article History

This contribution has not yet been formally edited by Britannica. Learn more.

To Kill a Mockingbird, novel by Harper Lee, published in 1960. An enormously popular novel, it was translated into some 40 languages and sold more than 30 million copies worldwide, and it won a Pulitzer Prize in 1961. The novel has been widely praised for its sensitive treatment of a child's awakening to racism and prejudice in the American South.

To Kill a Mockingbird

This book cover is one of many given to Harper Lee's classic work *To Kill a Mockingbird* (1960). The novel won a Pulitzer Prize in 1961 and the

RELATED TOPICS

- Harper Lee
- To Kill a Mockingbird
- Novel
- American literature
- Pulitzer Prize

SIMILAR TOPICS

- War and Peace
- Moby Dick
- Pride and Prejudice
- Don Quixote
- Les Misérables
- The Hobbit
- The Picture of Dorian Gray

How relevant is this page to the query?

1. Not relevant
2. A little relevant
3. Relevant
4. Very relevant
5. Perfectly relevant

Problems with Offline Evaluation

Unfortunately offline evaluation has substantial limitations, **annotated datasets** are:

Problems with Offline Evaluation

Unfortunately offline evaluation has substantial limitations, **annotated datasets** are:

- **time consuming and expensive** to make (Qin and Liu, 2013; Chapelle and Chang, 2011).

Problems with Offline Evaluation

Unfortunately offline evaluation has substantial limitations, **annotated datasets** are:

- **time consuming and expensive** to make (Qin and Liu, 2013; Chapelle and Chang, 2011).
- **unethical** to create in **privacy-sensitive settings** (Wang et al., 2016).

Problems with Offline Evaluation

Unfortunately offline evaluation has substantial limitations, **annotated datasets** are:

- **time consuming and 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**.

Problems with Offline Evaluation

Unfortunately offline evaluation has substantial limitations, **annotated datasets** are:

- **time consuming and 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 account for **future changes in relevancy** (Lefortier et al., 2014).

Problems with Offline Evaluation

Unfortunately offline evaluation has substantial limitations, **annotated datasets** are:

- **time consuming and 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 account for **future changes in relevancy** (Lefortier et al., 2014).
- **not necessarily aligned with actual user preferences** (Sanderson, 2010),

i.e. annotators and users often disagree.

User Interactions

Learning from users

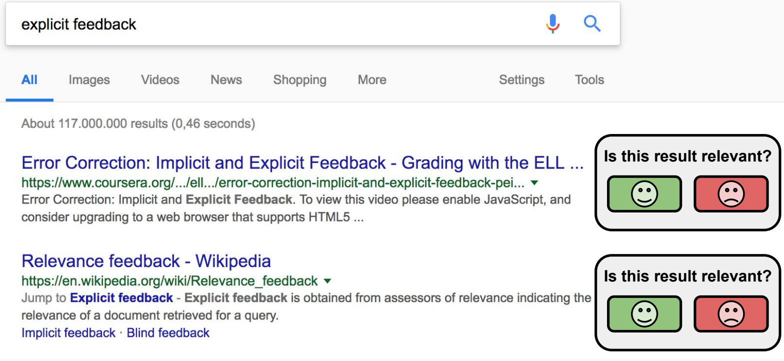
A solution to the problems of traditional evaluation is to **learn from users directly**.

Learning from users

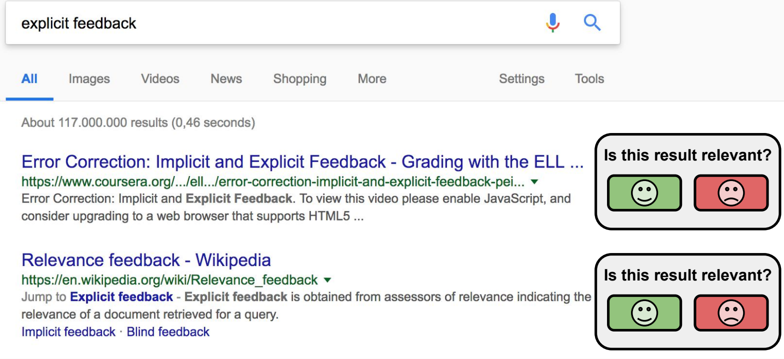
A solution to the problems of traditional evaluation is to **learn from users directly**.

Instead of having **annotators** guess, **why don't we ask users** if they are happy?

Direct User Feedback

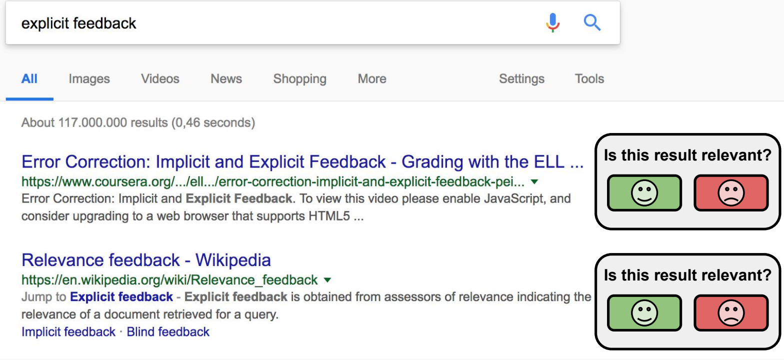


Direct User Feedback



- **Users hate giving feedback** like this.
- The process is too **invasive** and considered annoying.

Direct User Feedback



- **Users hate giving feedback** like this.
- The process is too **invasive** and considered annoying.
- Also vulnerable to **abuse**.

User Behaviour

We can expect:

- People to act in their own interest.
- Users to behave according to what they want.

User Behaviour

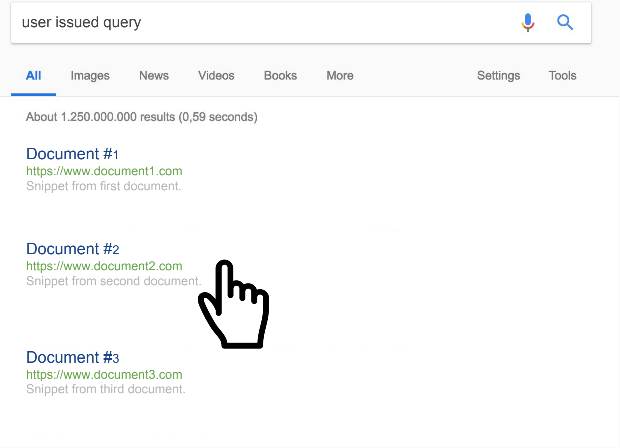
We can expect:

- People to act in their own interest.
- Users to behave according to what they want.

Thus, user behaviour to **indirectly indicate** user satisfaction.

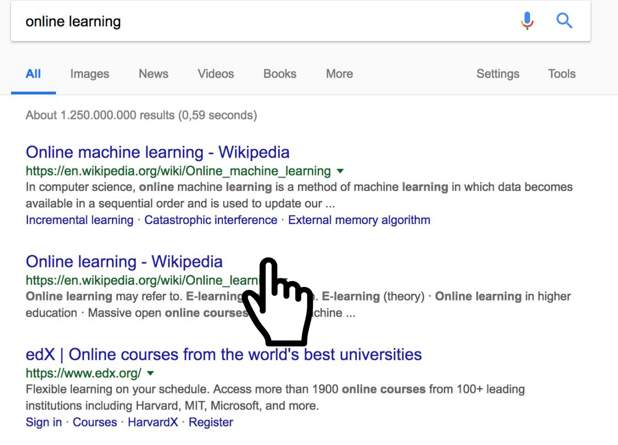
We have to **infer preferences** from user behaviour,
this means it provides **implicit feedback** (Joachims et al., 2017a).

Implicit User Feedback



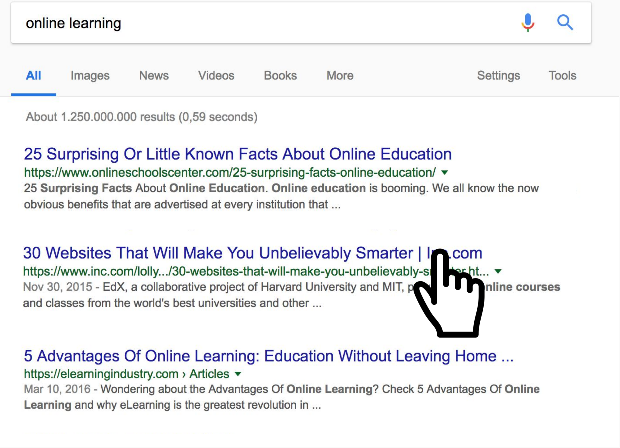
What can we learn from this interaction?

Implicit User Feedback: Example 1



What can we learn from this interaction?

Implicit User Feedback: Example 2



What can we learn from this interaction?

Implicit User Feedback: Lesson

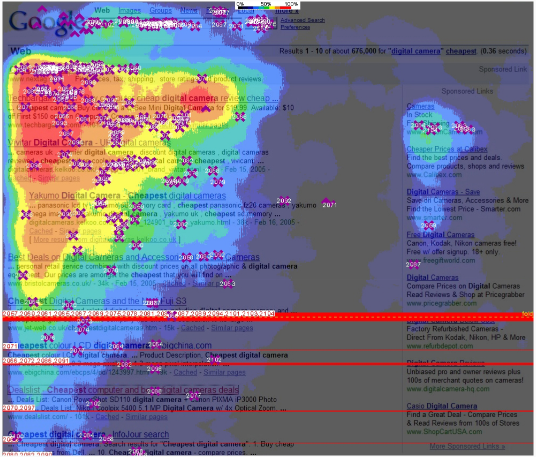
Be careful with what you infer from a user interaction.

Two types of trouble:

- **Noise**: users often click for unexpected reasons.
- **Bias**: some documents are more likely to be clicked for other reasons.

Eye-tracking studies

How do users look at results?



Source: http://www.mediative.com/

Bias in interactions

Unavoidable biases in search:

- **Position bias**:
  - documents **placed higher are more likely** to be considered.
- **Selection bias**:
  - users will **only click on documents you present** them.

Implicit Feedback

Methods that work with user interactions must:

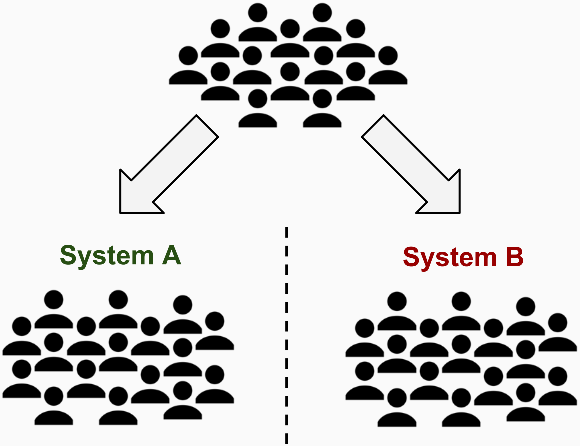
- be **robust to interaction noise**.
- be able to handle **position- and selection-bias**.

Related Approaches

A/B testing

Split the users in two groups, one group is given system **A**, the other system **B**.

The **differences in behaviour** allows for a comparison of the systems.



A/B testing: Advantages/Disadvantages

Split the users in two groups, one group is given system **A**, the other system **B**.

Advantages:

- **Straightforward** and common method, also outside of IR.
- Can test **many aspects** of user behaviour.

Disadvantages:

- **Inefficient**, requires a lot of user data.
- Tests have to run for a **long time**.
- Need to recognize individual users.

Evaluation based on click logs

Use **historical click logs to estimate performance** of a ranker.
Based around **removing the effect of bias** in the collected data.

Advantages:

- Can be performed on historical data,
  thus **no new experiments** have to be ran for a new system.

Disadvantages:

- Requires **good estimates of position bias**, this is not trivial.
- Does not work in **cold-start cases**.

Still a **very active area of research**: (Joachims et al., 2017b; Wang et al., 2018).

Online Evaluation: The Idea

Online Evaluation

Online Evaluation methods have **control over what to display** to the user.

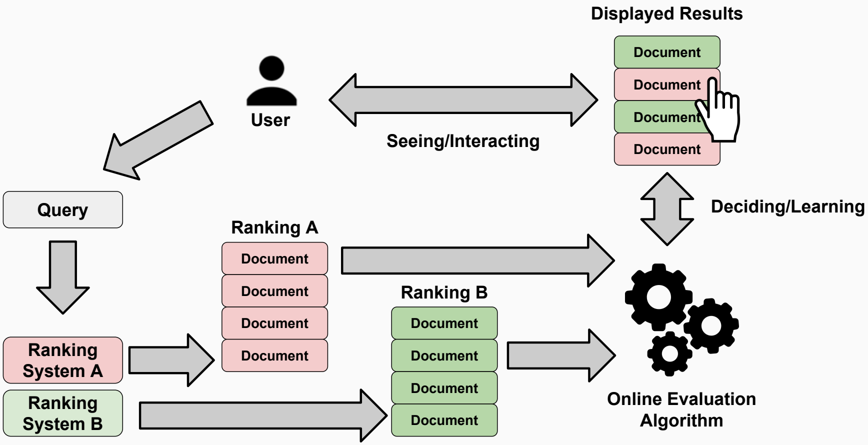
At the same time they:

- **Decide what results to display to the user.**
- **Infer preferences** from user interactions with the **chosen results**.

These methods can be **much more efficient**,
because they have (some) **control over what data is gathered**.

Online Evaluation: Visualization

Online Evaluation methods have **control over what to display** to the user.



Online Evaluation: Requirements

An **online evaluation** algorithm should:

Online Evaluation: Requirements

An **online evaluation** algorithm should:

- Give **reliable** and **correct** results:

Online Evaluation: Requirements

An **online evaluation** algorithm should:

- Give **reliable** and **correct** results:
  - **Guarantee** to provide correct comparison outcomes.
    i.e. theoretical proof.

Online Evaluation: Requirements

An **online evaluation** algorithm should:

- Give **reliable** and **correct** results:
  - **Guarantee** to provide correct comparison outcomes.
    i.e. theoretical proof.
  - **Robust** to interaction **noise**.
  - Handle **biases** in user interactions.

Online Evaluation: Requirements

An **online evaluation** algorithm should:

- Give **reliable** and **correct** results:
  - **Guarantee** to provide correct comparison outcomes.
    i.e. theoretical proof.
  - **Robust** to interaction **noise**.
  - Handle **biases** in user interactions.
- Provide **good user experience**:
  - Methods should **not interfere** with user task.

Balanced Interleaving

Balanced Interleaving

First online evaluation method by Joachims (2002a), introduced the concept **interleaving for evaluation**.

Main idea:

1. Take the rankings of two systems (A & B) for a query.
2. Created an interleaved result list by **combining the two lists**.
3. **Clicks indicate preferences** between rankers.
4. A large number of clicks give a **reliable** preference signal.

Balanced Interleaving: Algorithm

Algorithm 1 Balanced interleaving #1: construction

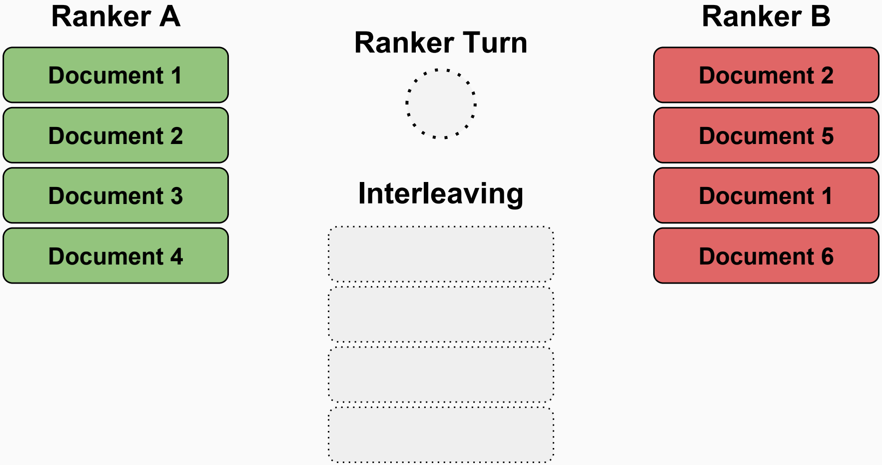
Input: rankings R_A, R_B, number of documents k
$R_1, R_2 \leftarrow$ *shuffle*(R_A, R_B)
$L \leftarrow []; i_1 \leftarrow 0; i_2 \leftarrow 0$
for $i \leftarrow 1, \ldots, k$ **do**
    if $i_1 \leq i_2$ **then**
        if $R_1[li] \notin L$ **then**
            append$(L, R_1[i_1])$
        end if
        $i_1 \leftarrow i_1 + 1$
    else
        if $R_2[li] \notin L$ **then**
            append$(L, R_2[i_2])$
        end if
        $i_2 \leftarrow i_2 + 1$
    end if
end for

Balanced Interleaving: Algorithm

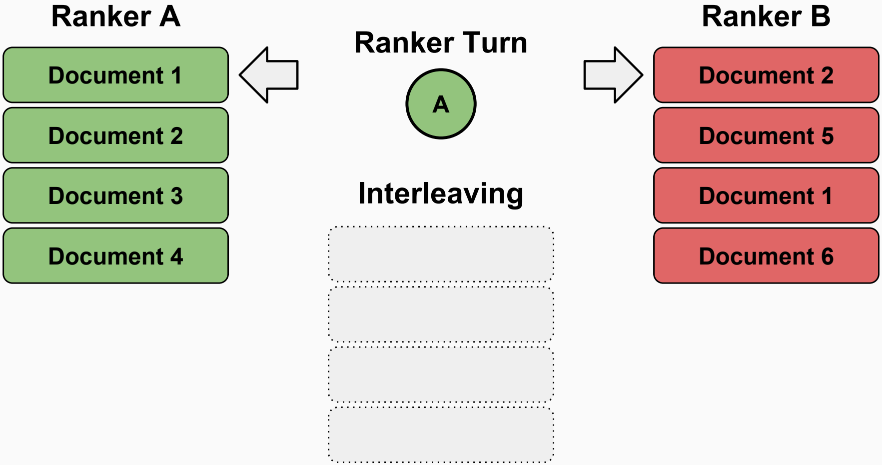
In plain English:

1. **randomly choose** one of the rankers to **begin**
2. then the **rankers take turns**:
   1. **chosen ranker places** their next document **unless it has already been placed**
   2. **turn** goes to the **other ranker**
   3. **repeat** until k documents are placed
3. **display resulting interleaving** to user, observe clicks

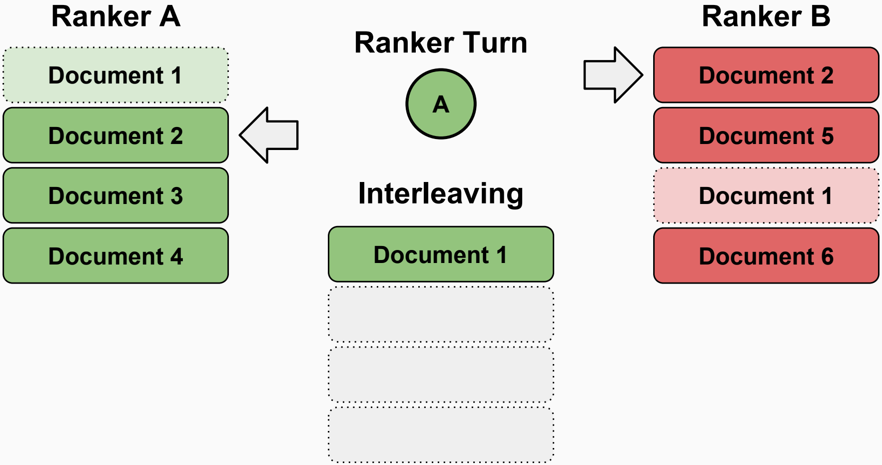
Balanced Interleaving: Visualization



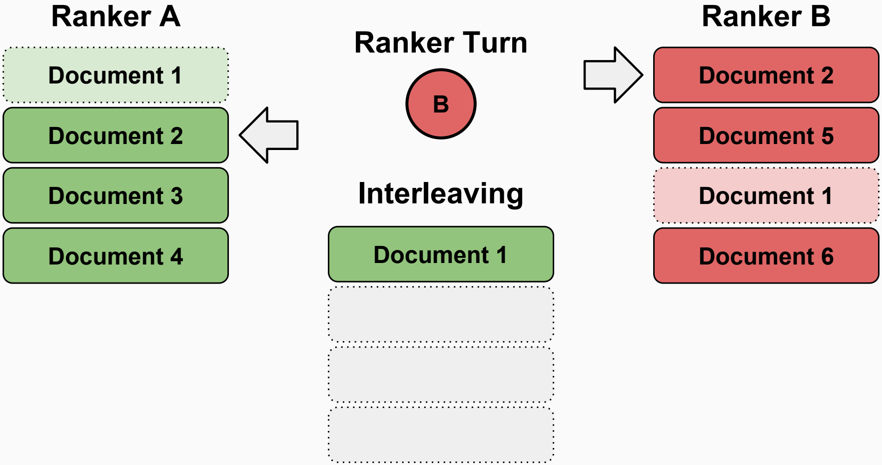
Balanced Interleaving: Visualization



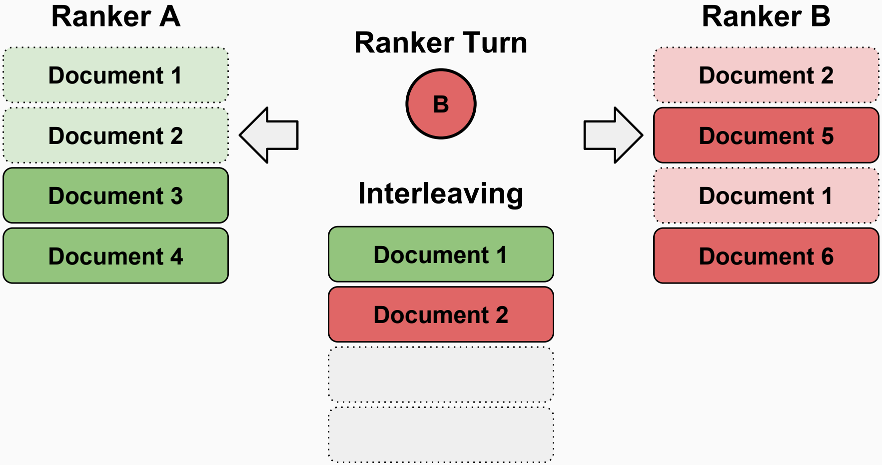
Balanced Interleaving: Visualization



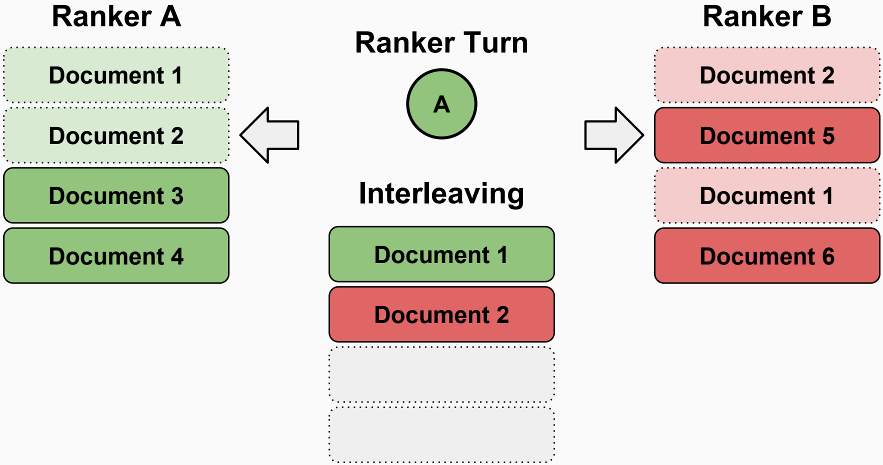
Balanced Interleaving: Visualization



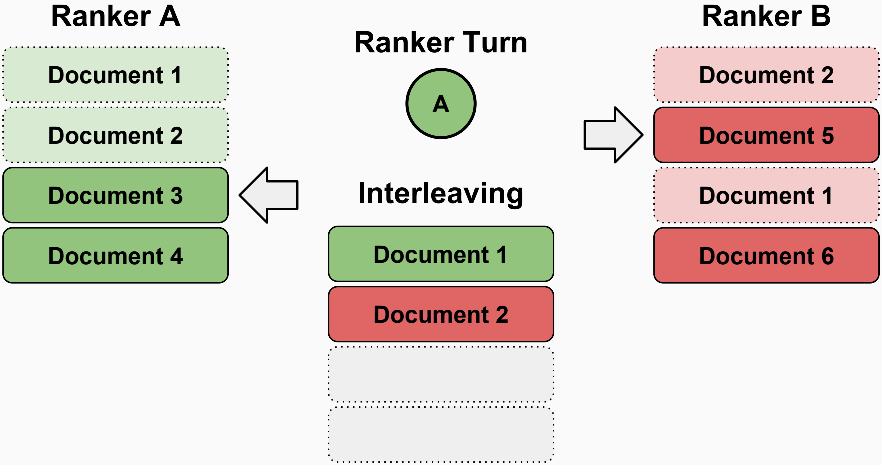
Balanced Interleaving: Visualization



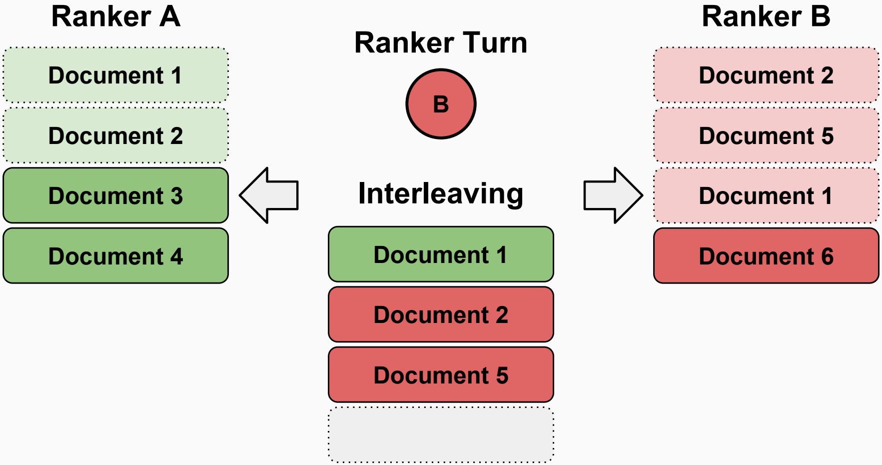
Balanced Interleaving: Visualization



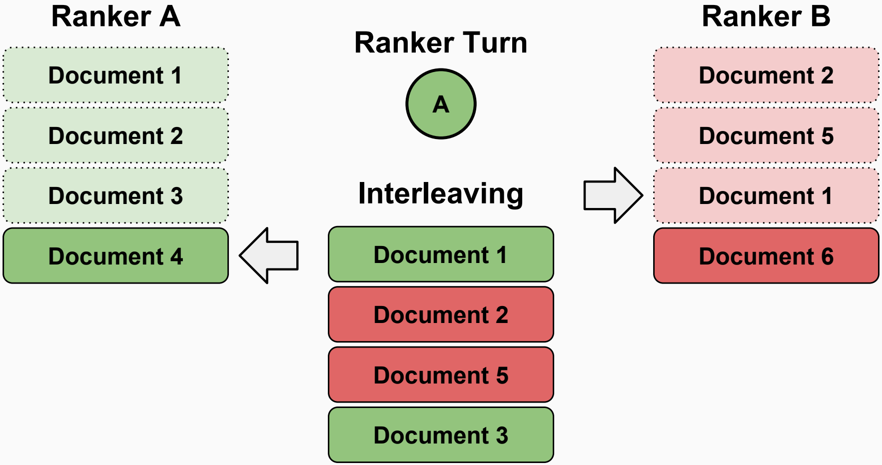
Balanced Interleaving: Visualization



Balanced Interleaving: Visualization



Balanced Interleaving: Visualization

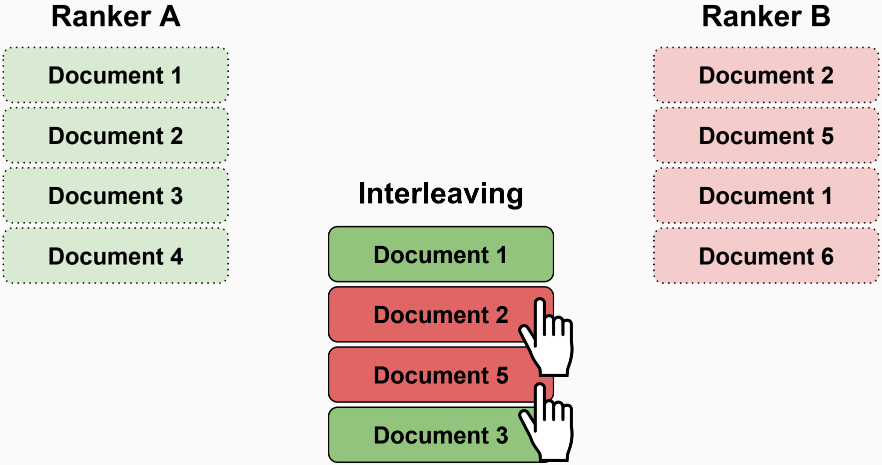


Balanced Interleaving: Algorithm

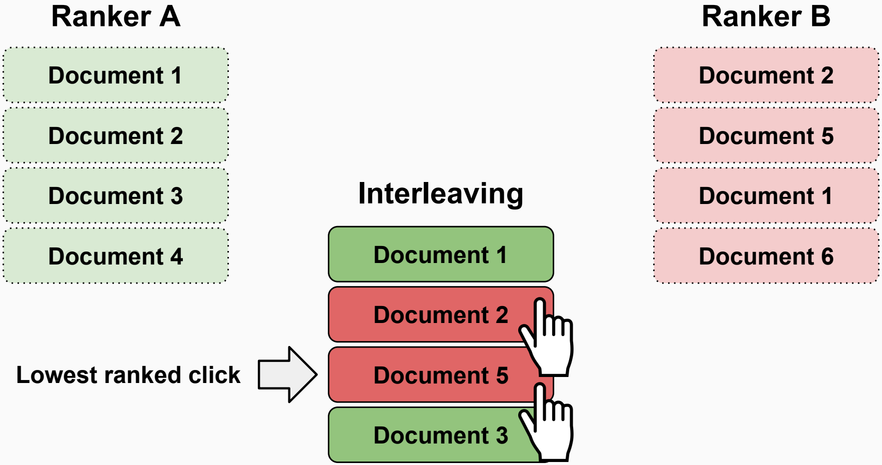
Inference of preference from clicks:

1. Determine the clicked document with the **lowest displayed rank**: d_{max}
2. Take the **highest rank** for d_{max} over the two rankers : i_{min}
3. **Count the clicked documents** for each ranker at i_{min} or above.
4. The ranker with the **most clicks** is preferred.

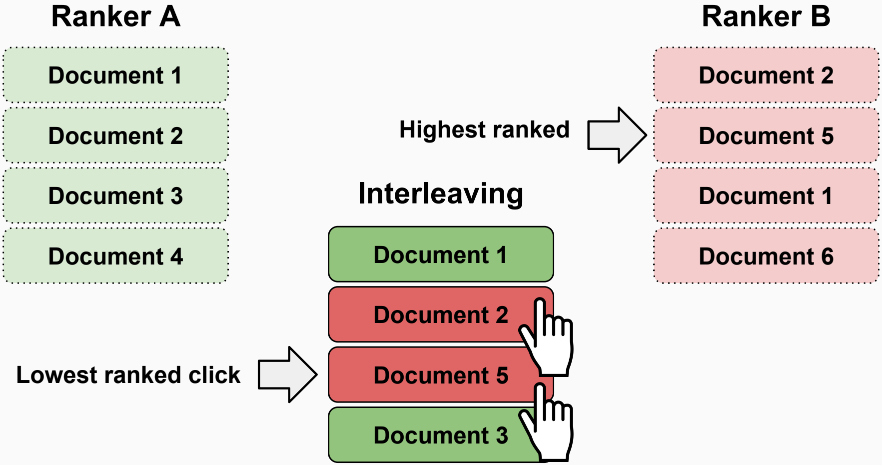
Balanced Interleaving: Visualization



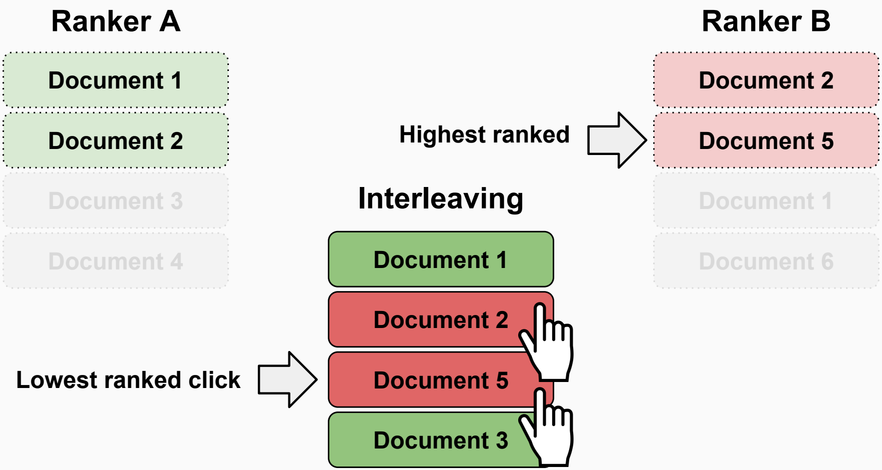
Balanced Interleaving: Visualization



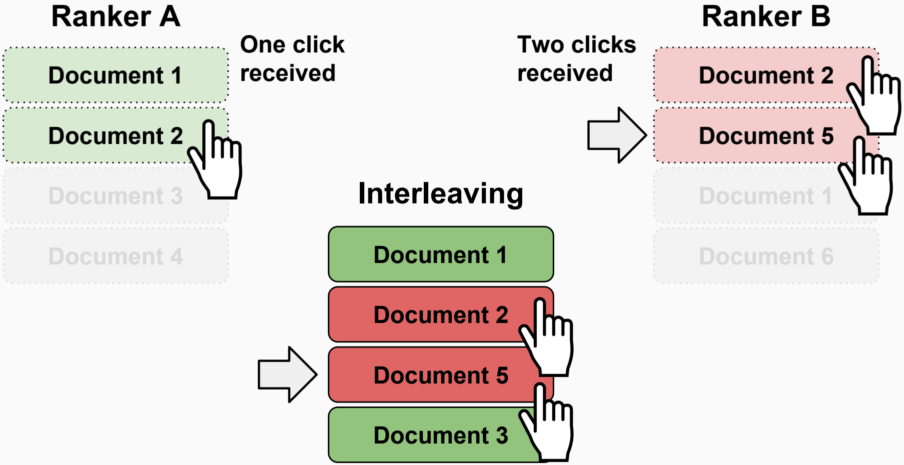
Balanced Interleaving: Visualization



Balanced Interleaving: Visualization



Balanced Interleaving: Visualization



Balanced Interleaving: Properties

The properties of **balanced interleaving**:

Balanced Interleaving: Properties

The properties of **balanced interleaving**:

- **Good user experience**:
  - User **hardly affected** by method.
  - Experience will **not be worse** than that of the worst ranker.

Balanced Interleaving: Properties

The properties of **balanced interleaving**:

- **Good user experience**:
    - User **hardly affected** by method.
    - Experience will **not be worse** than that of the worst ranker.
- **Correctness**:

Balanced Interleaving: Properties

The properties of **balanced interleaving**:

- **Good user experience**:
  - User **hardly affected** by method.
  - Experience will **not be worse** than that of the worst ranker.
- **Correctness**:
  - Ranker that **places relevant documents** usually wins.

Balanced Interleaving: Properties

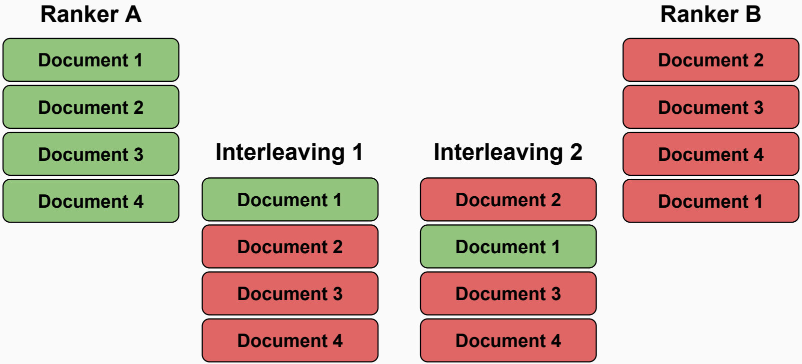
The properties of **balanced interleaving**:

- **Good user experience**:
  - User **hardly affected** by method.
  - Experience will **not be worse** than that of the worst ranker.
- **Correctness**:
  - Ranker that **places relevant documents** usually wins.
  - **Correct outcomes not guaranteed.**

Balanced Interleaving: Problematic Example

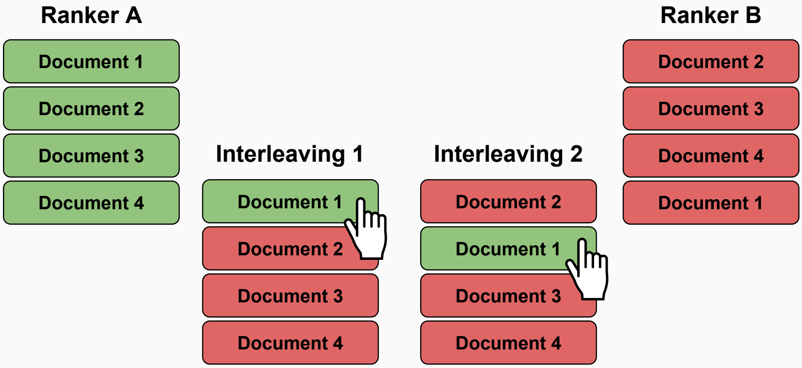
Ranker A	Ranker B
Document 1	Document 2
Document 2	Document 3
Document 3	Document 4
Document 4	Document 1

Balanced Interleaving: Problematic Example



Balanced Interleaving: Problematic Example

Only a click on Document 1 can lead to a preference for ranker **A**.



A random click is more likely to lead to a preference for ranker **B**, this is **very unfair**.

Overview: Interleaving

	User Experience	**Correctness**	**Source**
Balanced Interleaving	✓		(Joachims, 2002a)

Team-Draft Interleaving

Team-Draft Interleaving: Introduction

Reaction to these problematic cases for Balanced Interleaving introduced by Radlinski et al. (2008).
Designed to be **unbiased** under **uniform random clicks**.

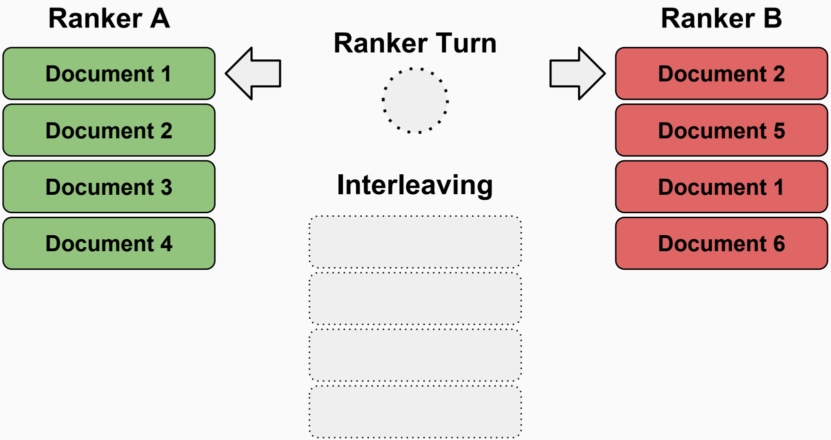
Simpler method, based on how teams are selected for sport class in school.

Team-Draft Interleaving: Algorithm Simple

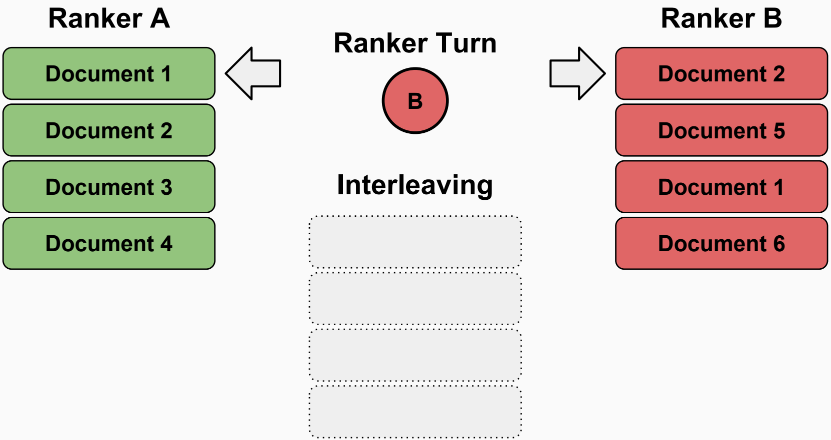
In plain English:

1. Until k documents are placed:
2.
   1. Randomly choose ranker **A** or **B**.
   2. Let **chosen ranker** place its **next** *unplaced* document.
   3. Let **other ranker** place its **next** *unplaced* document.
   4. Remember which ranker placed which document.
3. Present interleaving to user, observe clicks.
4. Ranker with the most clicks on its placed document wins.

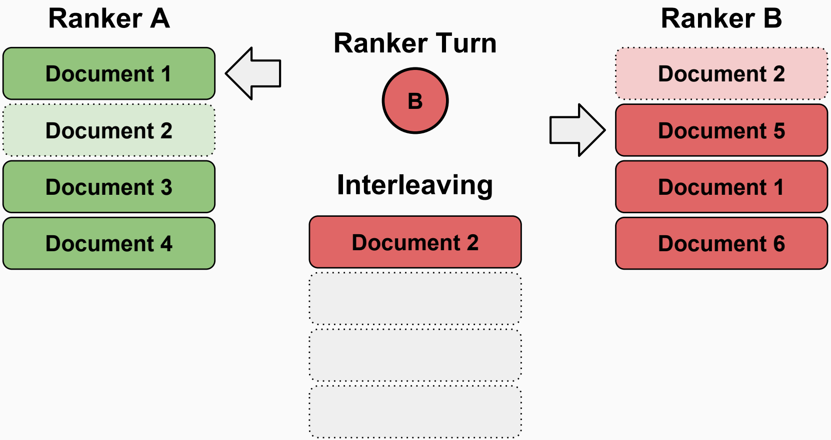
Team-Draft Interleaving: Visualization



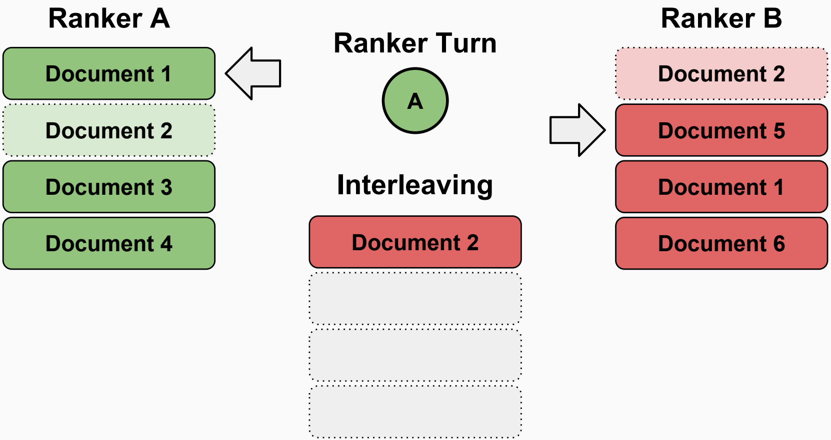
Team-Draft Interleaving: Visualization



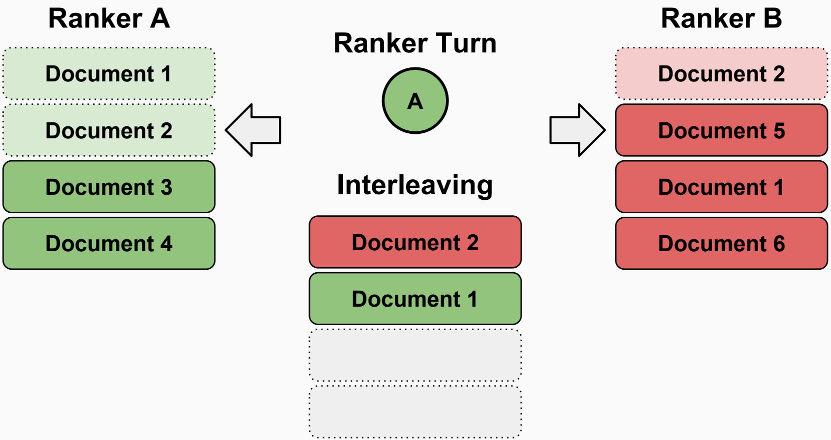
Team-Draft Interleaving: Visualization



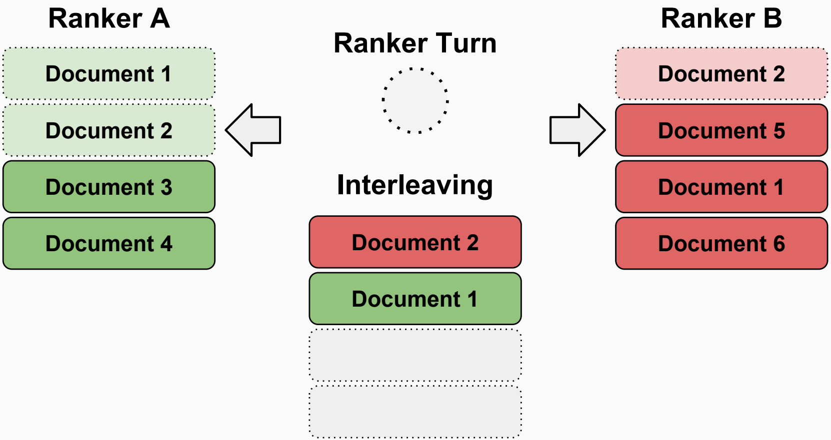
Team-Draft Interleaving: Visualization



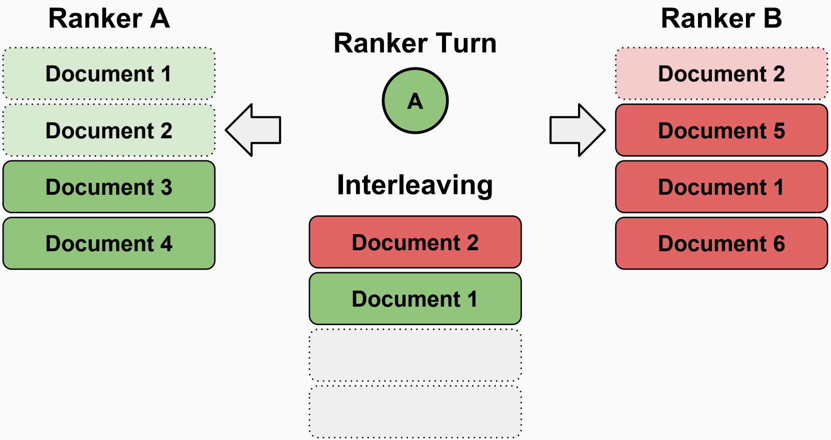
Team-Draft Interleaving: Visualization



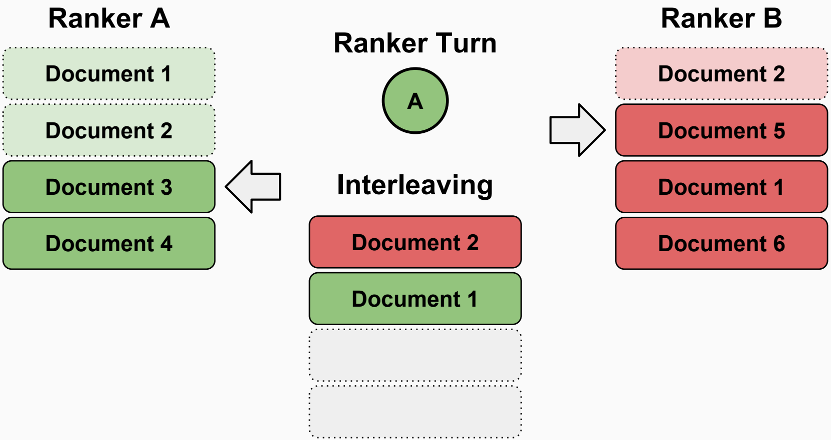
Team-Draft Interleaving: Visualization



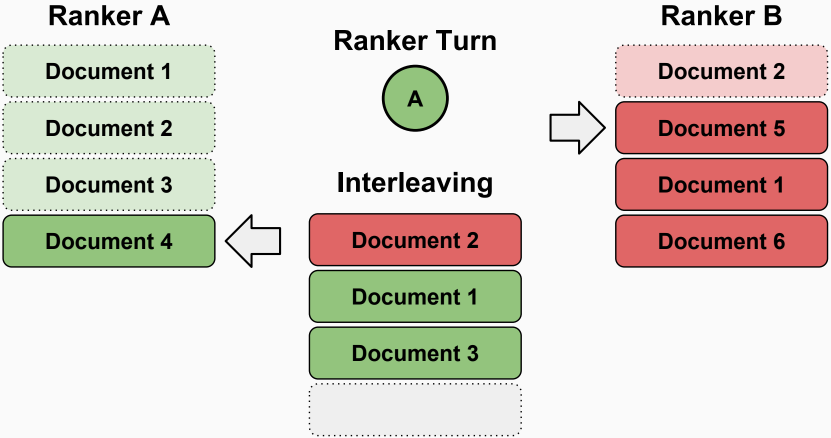
Team-Draft Interleaving: Visualization



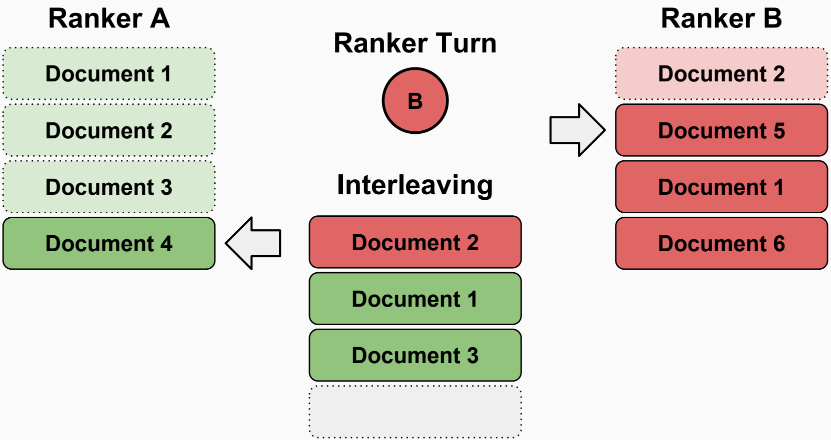
Team-Draft Interleaving: Visualization



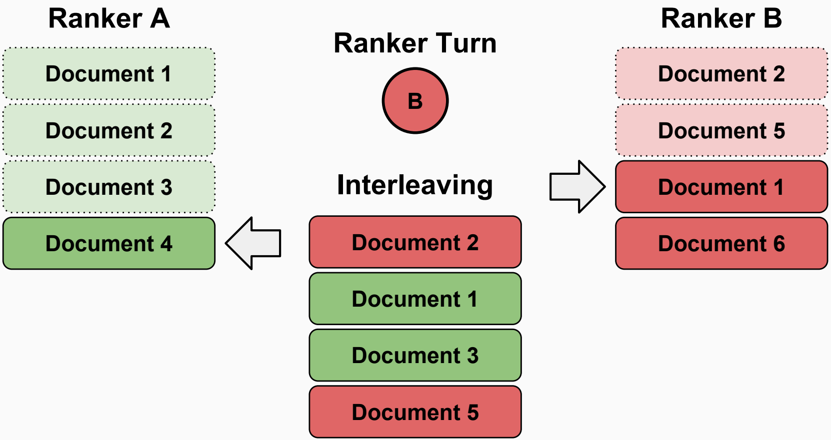
Team-Draft Interleaving: Visualization



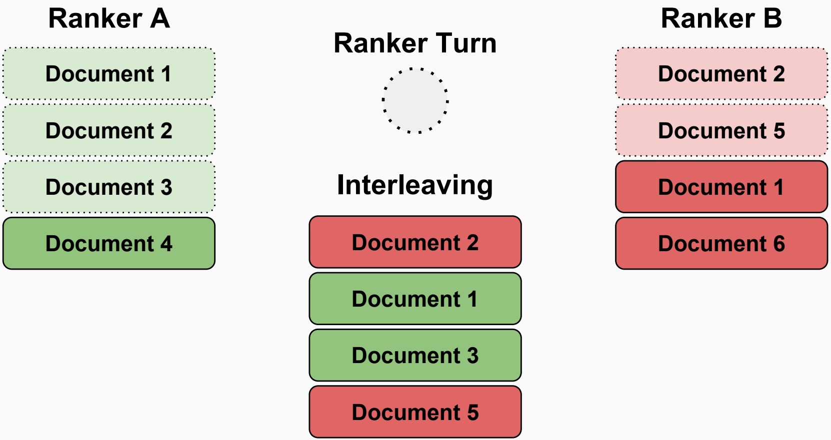
Team-Draft Interleaving: Visualization



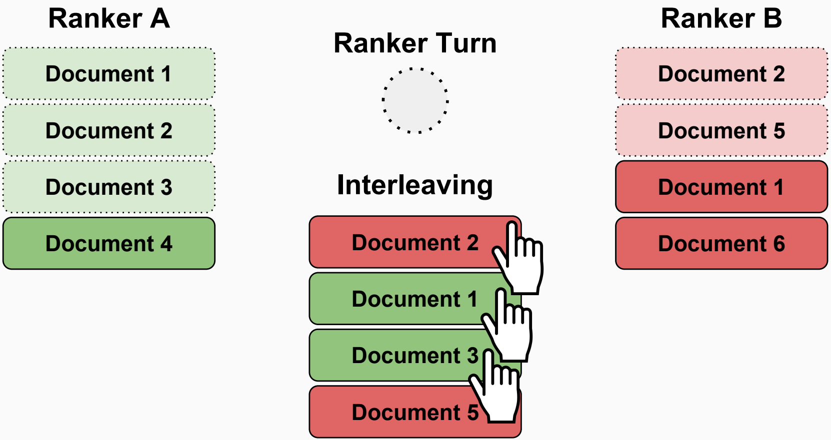
Team-Draft Interleaving: Visualization



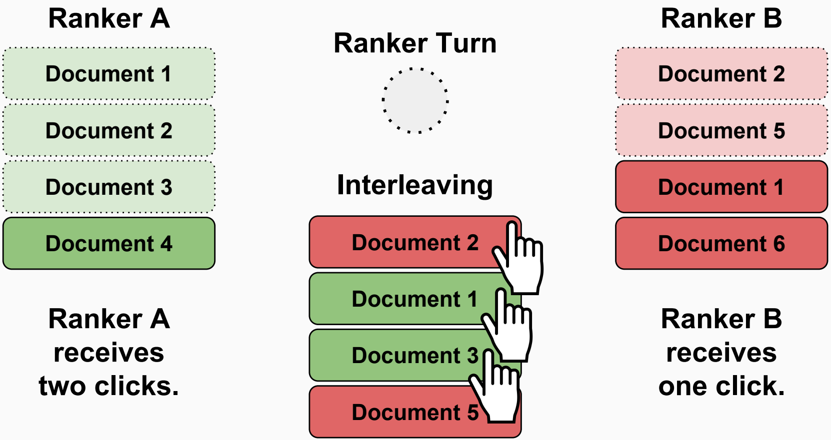
Team-Draft Interleaving: Visualization



Team-Draft Interleaving: Visualization



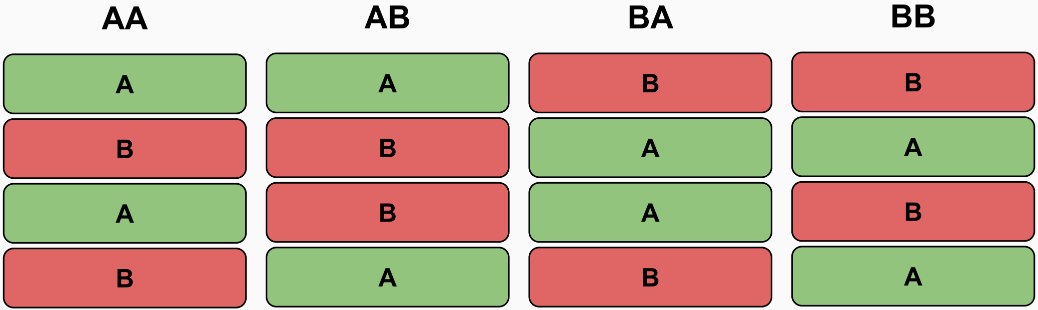
Team-Draft Interleaving: Visualization



Team-Draft Interleaving: Resolved Example

Team-Draft interleaving finds **no preferences** under **random** clicks.

All possible assignments:



Team-Draft Interleaving: Comparison to A/B testing

From (Schuth et al., 2015b), power is an indication of sensitivity.

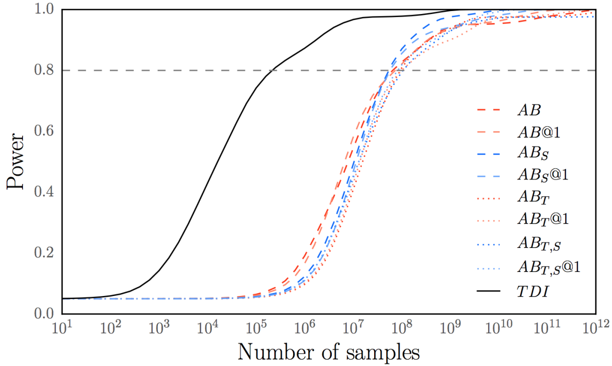


Figure 1: Power as a function of sample size, computed using the observed effect sizes for 38 interleaving and AB comparisons, averaged over all comparisons (assuming two-sided t-test with $p = 0.05$, as described in Section 4.2).

Team-Draft Interleaving: Properties

Properties of Team-Draft interleaving:

- **User experience**:
  - User **hardly affected** by method.
  - Experience will not be worse than that of the worst ranker.

Team-Draft Interleaving: Properties

Properties of Team-Draft interleaving:

- **User experience**:
  - User **hardly affected** by method.
  - Experience will not be worse than that of the worst ranker.
- **Correctness**:

Team-Draft Interleaving: Properties

Properties of Team-Draft interleaving:

- **User experience**:
  - User **hardly affected** by method.
  - Experience will not be worse than that of the worst ranker.
- **Correctness**:
  - Ranker that places relevant documents **usually wins**.
  - Does **not make the same mistakes** as Balanced Interleaving.

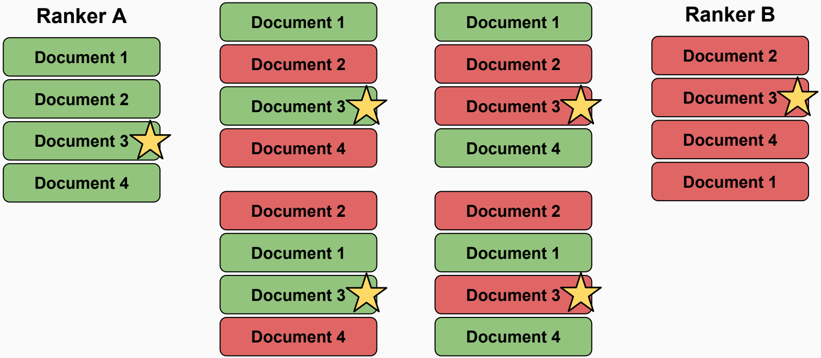
Team-Draft Interleaving: Properties

Properties of Team-Draft interleaving:

- **User experience**:
  - User **hardly affected** by method.
  - Experience will not be worse than that of the worst ranker.
- **Correctness**:
  - Ranker that places relevant documents **usually wins**.
  - Does **not make the same mistakes** as Balanced Interleaving.
  - **Correct outcomes not guaranteed.**

Team-Draft Interleaving: Problematic Example

Note this example, where document 3 is the **only relevant one**.



Ranker **B** should win, but **in expectation no preference will be found**.

Overview: Interleaving

	User Experience	Correctness	Source
Balanced Interleaving	✓		(Joachims, 2002a)
Team-Draft Interleaving	✓		(Radlinski et al., 2008)

Fidelity in Online Evaluation

Fidelity in Online Evaluation

Simply **solving some incorrect cases** of the previous methods does **not guarantee correctness** of a method.

Hofmann et al. (2013) introduced the idea of **fidelity**, which **formalizes a level of correctness** for methods to obtain.

Fidelity in Online Evaluation: Condition #1

Condition 1 for fidelity:

- If user **clicks are independent from document relevance**, i.e. random clicks, then the interleaving method should **not find any differences** between rankers.

Fidelity in Online Evaluation: Condition #1

Condition 1 for fidelity:

- If user **clicks are independent from document relevance**, i.e. random clicks, then the interleaving method should **not find any differences** between rankers.

Rankers shouldn't have an **advantage** due to factors **other than relevance**.

Fidelity in Online Evaluation: Pareto Domination

Pareto domination identifies cases where the correct winner is **unambiguous**.

Ranker **A** pareto dominates ranker **B** if and only if:

- Ranker **A** ranks **every relevant document at least as high** as ranker **B**, and there is **at least one relevant document** that ranker **A** ranks **higher**.

Fidelity in Online Evaluation: Pareto Domination

Pareto domination identifies cases where the correct winner is **unambiguous**.

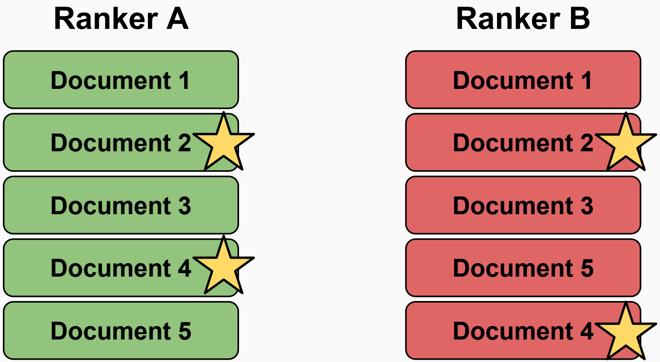
Ranker **A** pareto dominates ranker **B** if and only if:

- Ranker **A** ranks **every relevant document at least as high** as ranker **B**, and there is **at least one relevant document** that ranker **A** ranks **higher**.

Reasonably ranker **A** should always be preferred over ranker **B**.

Fidelity in Online Evaluation: Pareto Domination Visualized

Ranker **A** **Pareto dominates** ranker **B**, under any reasonable circumstances **A** should be preferred.



Fidelity in Online Evaluation: Condition #2

Condition 2 for fidelity:

- If user clicks are **correlated with document relevance**,
  i.e. relevant documents are **more likely to be clicked**,

  then a Pareto dominating ranker **should win the comparison** in expectation.

Fidelity in Online Evaluation: Condition #2

Condition 2 for fidelity:

- If user clicks are **correlated with document relevance**,
  i.e. relevant documents are **more likely to be clicked**,

  then a Pareto dominating ranker **should win the comparison** in expectation.

An unambiguous winner should always win the comparison (given enough clicks).

Fidelity in Online Evaluation: Conditions

Thus to have fidelity a method should:

1. Not give rankers an **advantage** due to factors **other than relevance**.
2. Always **prefer unambiguous winners** in expectation (given enough clicks).

Probabilistic Interleaving

Probabilistic Interleaving

Introduced by Hofmann et al. (2011) designed around the **fidelity conditions**.

Treats rankers as **probability distributions** over a set of documents.

Probabilistic Interleaving: Rankers as Probability Distributions

A ranker **A** with the ranking:

$$R_A(D) = [d_1, d_2, \ldots d_N] \tag{7}$$

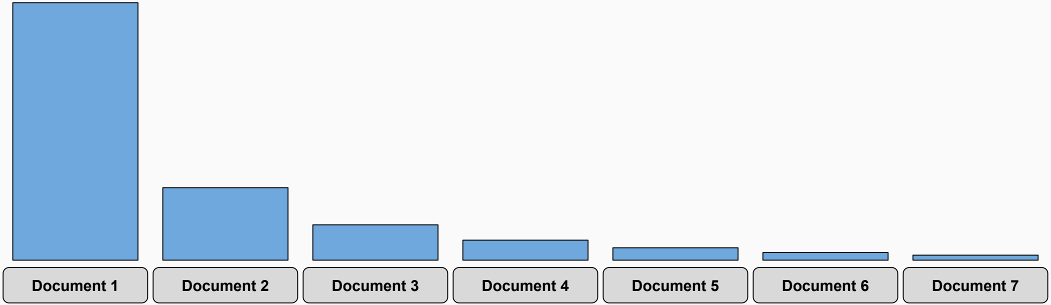
then let $\textit{rank}(d, R_A)$ be the rank of d in R_A.

The distribution for ranker **A** is modelled by:

$$P(d|D, R_A) = P_A(d) = \frac{\frac{1}{\textit{rank}(d,R_A)^{\tau}}}{\sum_{d' \in D} \frac{1}{\textit{rank}(d',R_A)^{\tau}}} \tag{8}$$

Renormalize after each document is removed, i.e. remove sampled document from D.

Probabilistic Interleaving: Rankers as Probability Distributions



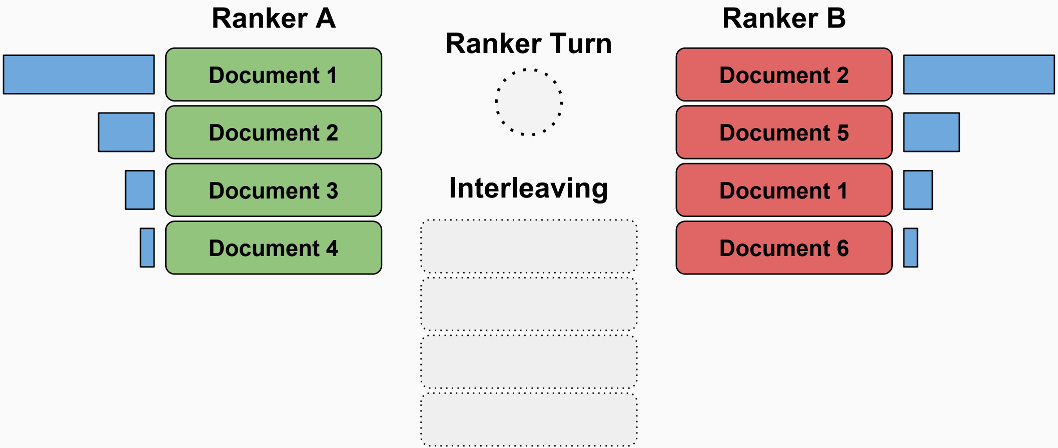
Example of a possible **document distribution**.

Probabilistic Interleaving: Proto-Method

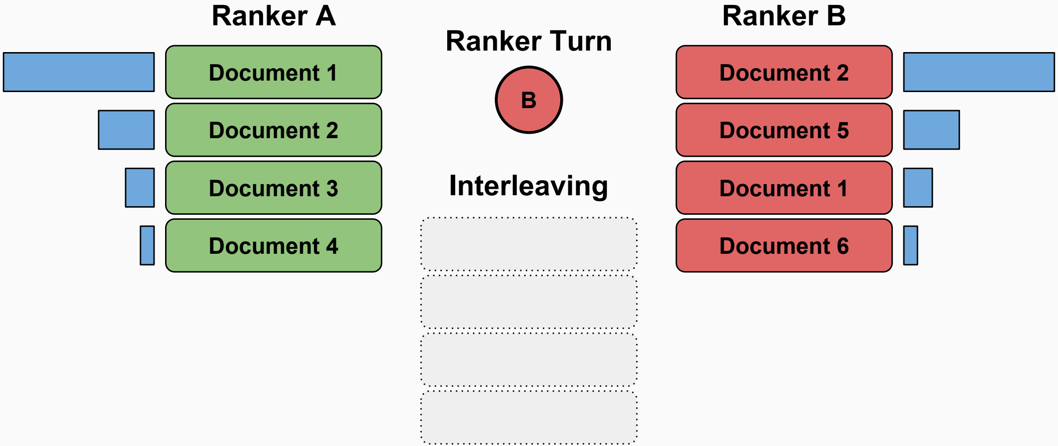
Consider this proto-version of Probabilistic Interleaving:

1. Compute P_A and P_B from ranker **A** and **B** respectively.
2. Repeat until k documents placed:
   1. Randomly choose P_A or P_B and **sample a document** d.
   2. Place d and remember whether **A** or **B** was chosen.
   3. Renormalize P_A or P_B after removing d.
3. Display to user and observe clicks.
4. Ranker with the most clicked documents wins comparison.

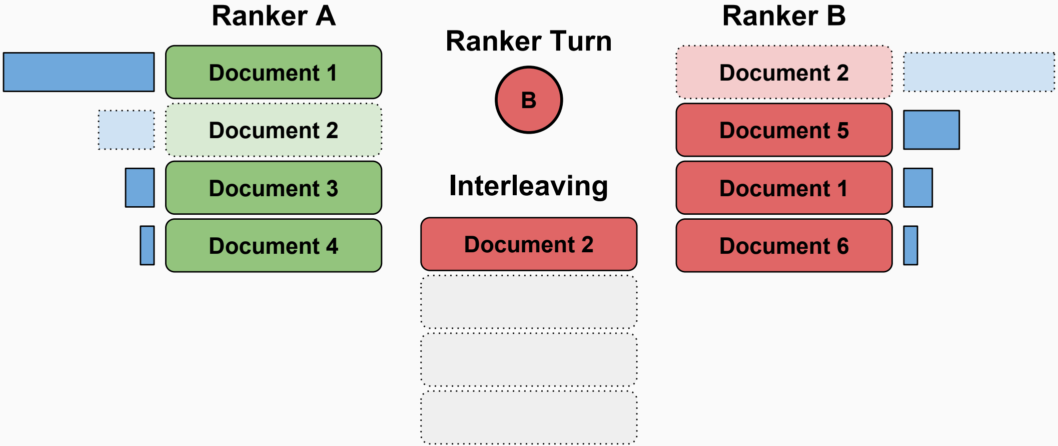
Probabilistic Interleaving: Proto-Method Visualization



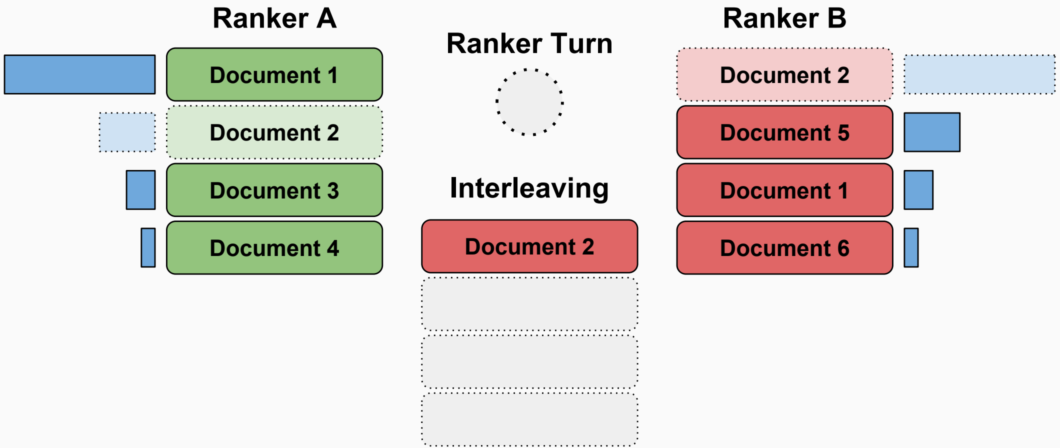
Probabilistic Interleaving: Proto-Method Visualization



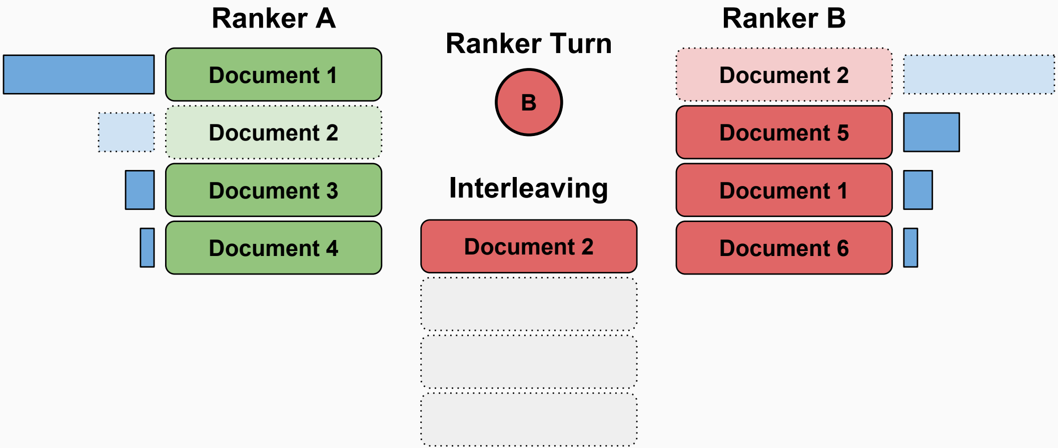
Probabilistic Interleaving: Proto-Method Visualization



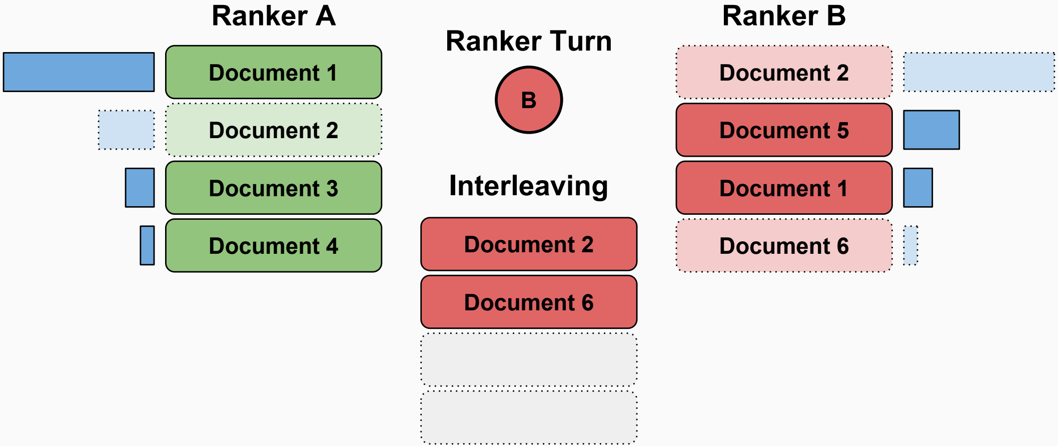
Probabilistic Interleaving: Proto-Method Visualization



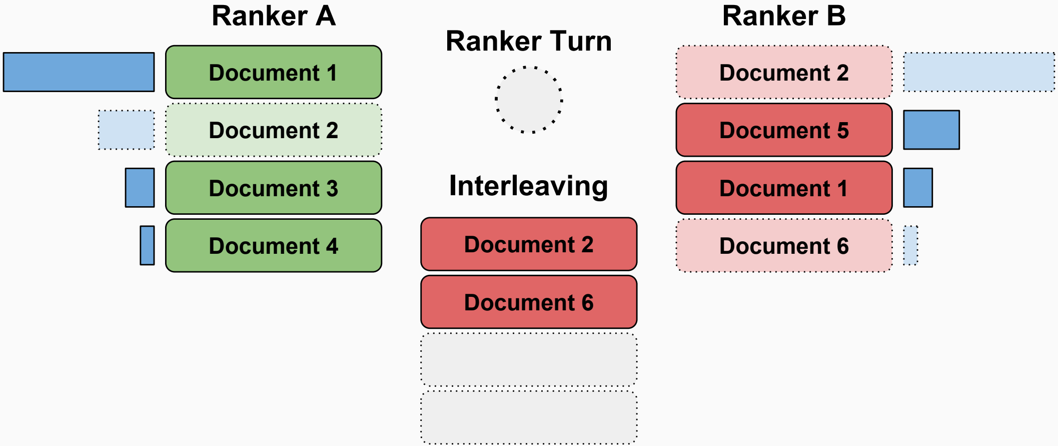
Probabilistic Interleaving: Proto-Method Visualization



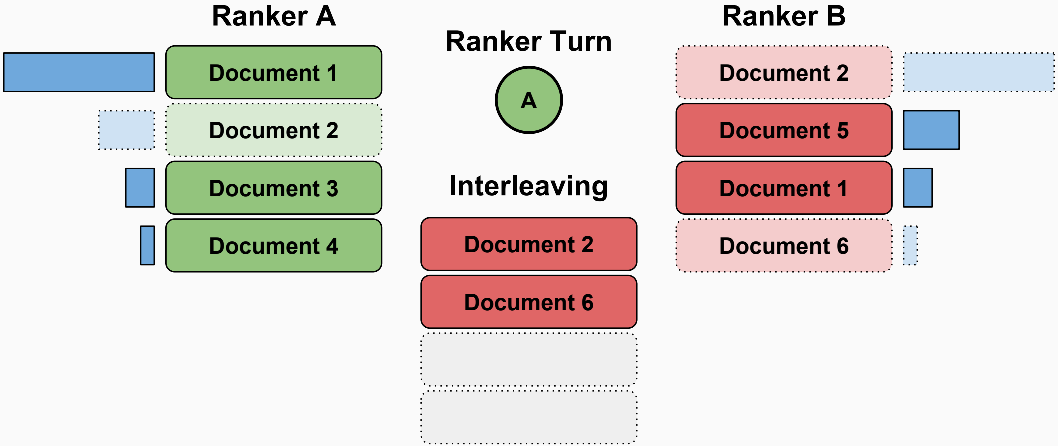
Probabilistic Interleaving: Proto-Method Visualization



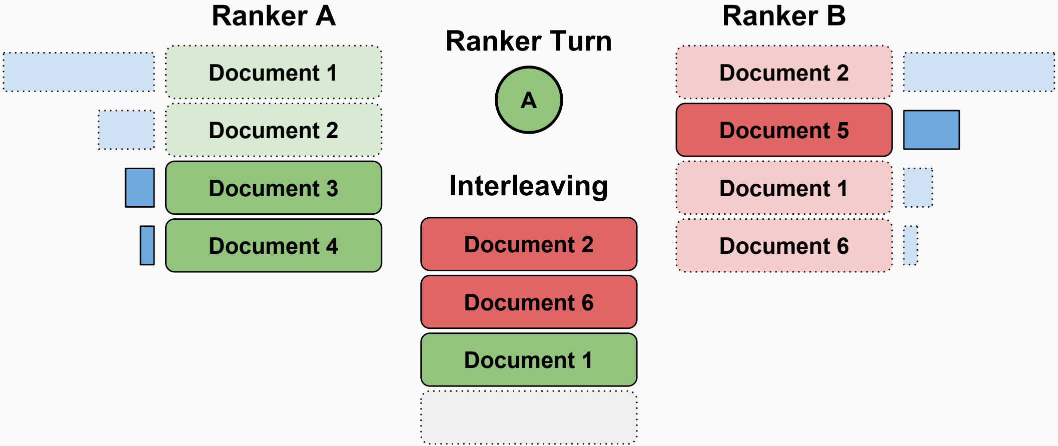
Probabilistic Interleaving: Proto-Method Visualization



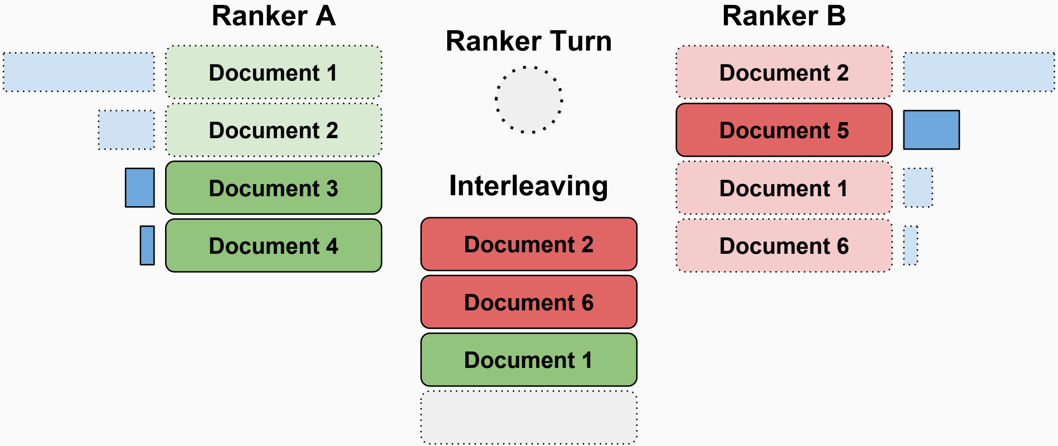
Probabilistic Interleaving: Proto-Method Visualization



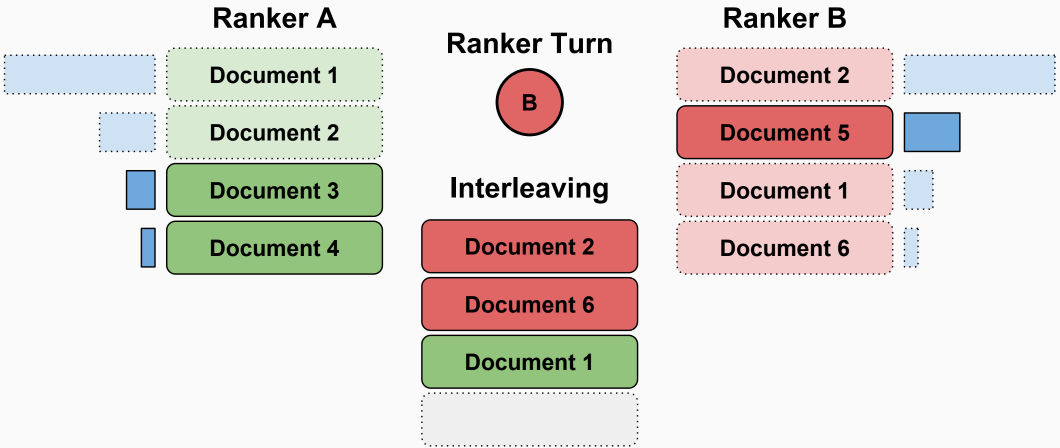
Probabilistic Interleaving: Proto-Method Visualization



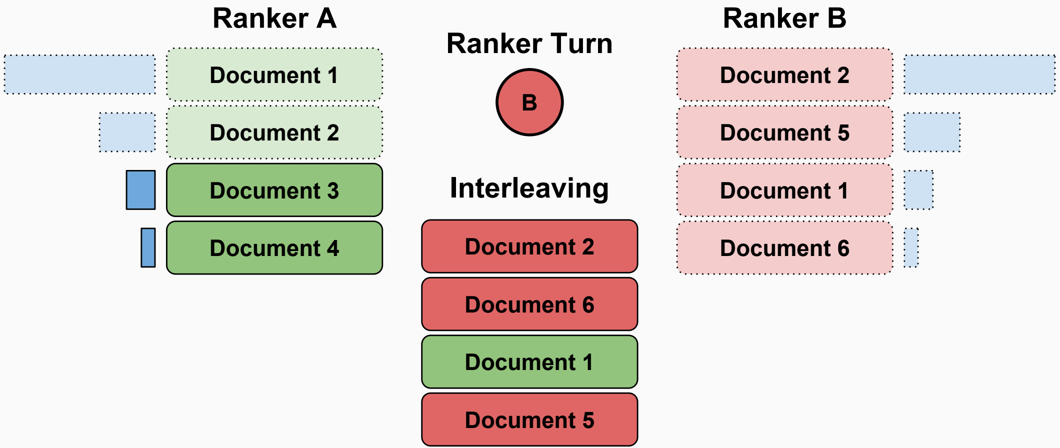
Probabilistic Interleaving: Proto-Method Visualization



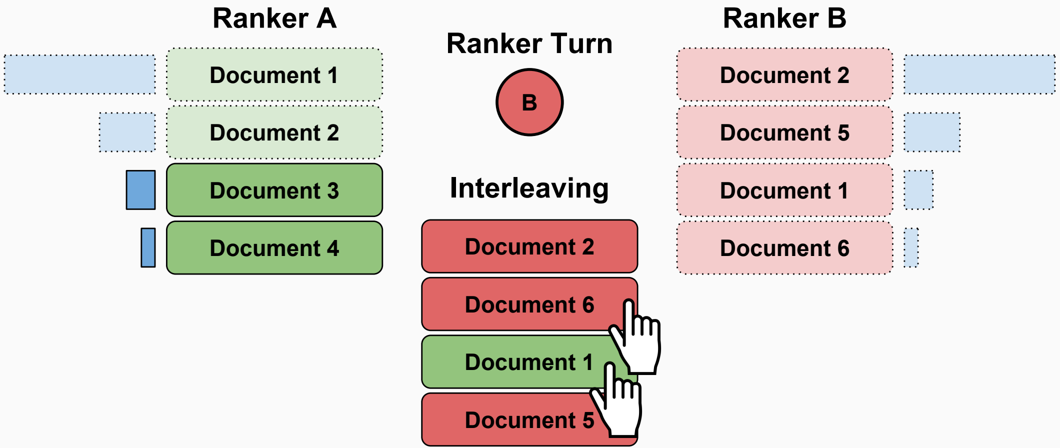
Probabilistic Interleaving: Proto-Method Visualization



Probabilistic Interleaving: Proto-Method Visualization



Probabilistic Interleaving: Proto-Method Visualization



Probabilistic Interleaving: Proto-Method Fidelity

Does this method have Fidelity?

1. Could a ranker have an **advantage** due to factors **other than relevance**?

Probabilistic Interleaving: Proto-Method Fidelity

Does this method have Fidelity?

1. Could a ranker have an **advantage** due to factors **other than relevance**?
   - Every ranker is equally likely to place at every rank.
   - Thus expected number of clicks is equal under random interactions.

Probabilistic Interleaving: Proto-Method Fidelity

Does this method have Fidelity?

1. Could a ranker have an **advantage** due to factors **other than relevance**?
    - Every ranker is equally likely to place at every rank.
    - Thus expected number of clicks is equal under random interactions.
2. Will an **unambiguous winners** always win in expectation?

Probabilistic Interleaving: Proto-Method Fidelity

Does this method have Fidelity?

1. Could a ranker have an **advantage** due to factors **other than relevance**?
    - Every ranker is equally likely to place at every rank.
    - Thus expected number of clicks is equal under random interactions.
2. Will an **unambiguous winners** always win in expectation?
    - Every ranker is equally likely to place at every rank.
    - Dominating ranker is more likely to place relevant documents at each rank.
    - If relevant documents are more likely to be clicked,
      then the dominating ranker wins in expectation.

Probabilistic Interleaving: Proto-Method Fidelity

Does this method have Fidelity?

1. Could a ranker have an **advantage** due to factors **other than relevance**?
   - Every ranker is equally likely to place at every rank.
   - Thus expected number of clicks is equal under random interactions.
2. Will an **unambiguous winners** always win in expectation?
   - Every ranker is equally likely to place at every rank.
   - Dominating ranker is more likely to place relevant documents at each rank.
   - If relevant documents are more likely to be clicked,
     then the dominating ranker wins in expectation.

Quite trivial to show that **this method has fidelity**.

Probabilistic Interleaving: Marginalization

The user **does not see** which ranker placed what documents,
thus their behaviour will **not be affected** by document assignments.

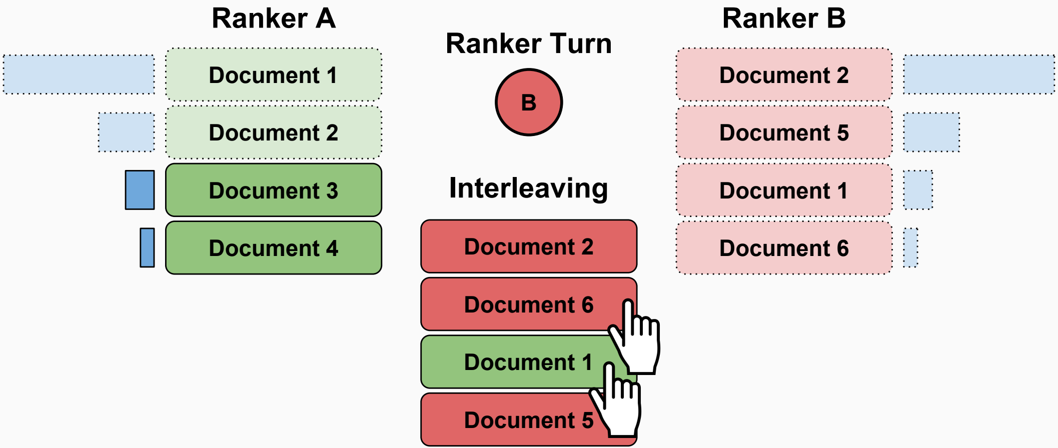
Probabilistic Interleaving: Marginalization

The user **does not see** which ranker placed what documents,
thus their behaviour will **not be affected** by document assignments.

Probabilistic interleaving takes the proto-method and **marginalizes over the assignments**:

- Instead of using the outcome based on the *true* ranker assignment, calculate the **expected outcome over all possible assignments**.

Probabilistic Interleaving: Expected Outcome Visualized



Probabilistic Interleaving: Expected Outcome Visualized

Probabilistic Interleaving: Expected Outcome

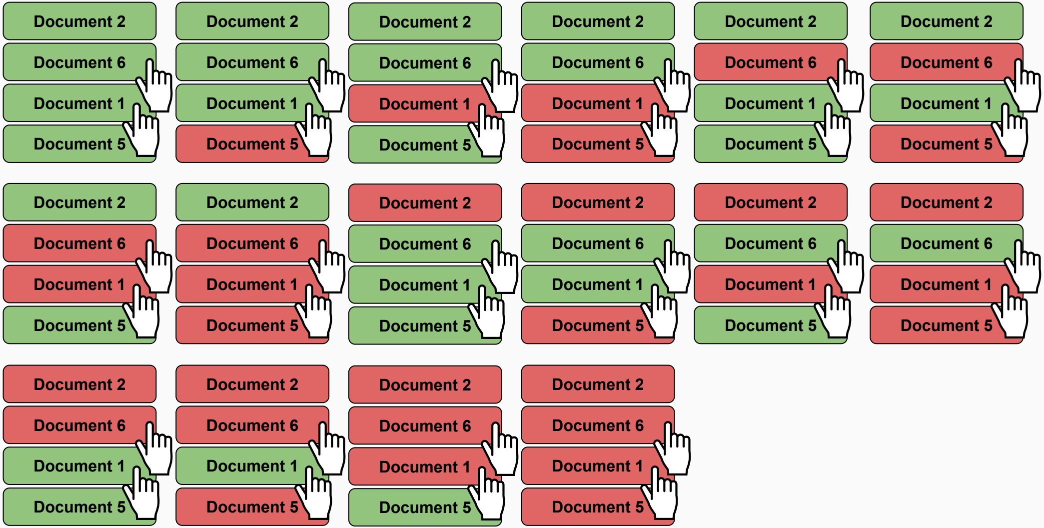
For rankings R_A, R_B, the interleaved list L, assignments T, and clicks c, the **outcome of a comparison** can be noted as:

$$O(R_A, R_B, L, T, c) \in \{-1, 0, 1\} \tag{9}$$

Since clicks are **independent** of the assignment T, we can **marginalize over all possible assignments** to reach an **expected outcome**:

$$E[O(R_A, R_B, L, c)] = \sum_T P(T|R_A, R_B, L)O(R_A, R_B, L, T, c) \tag{10}$$

Probabilistic Interleaving: Expected Outcome Visualized



Probabilistic Interleaving: Placement Probability

How do we calculate these probabilities?

Probabilistic Interleaving: Placement Probability

How do we calculate these probabilities?

We know the following:

$$P(T_i = A) = \frac{1}{2} \tag{11}$$

$$P(L_i = d | T_i = A) = P_A(d) = \frac{\frac{1}{\textit{rank}(d, R_A)^\tau}}{\sum_{d' \in D} \frac{1}{\textit{rank}(d', R_A)^\tau}} \tag{12}$$

$$P(T_i = A | L_i = d) = \quad ??? \tag{13}$$

Probabilistic Interleaving: Placement Probability

How do we calculate these probabilities?

We know the following:

$$P(T_i = A) = \frac{1}{2} \tag{14}$$

$$P(L_i = d \mid T_i = A) = P_A(d) = \frac{\frac{1}{\mathit{rank}(d, R_A)^\tau}}{\sum_{d' \in D} \frac{1}{\mathit{rank}(d', R_A)^\tau}} \tag{15}$$

$$P(T_i = A \mid L_i = d) = \frac{P(L_i = d, T_i = A)}{P(L_i = d)} \tag{16}$$

Probabilistic Interleaving: Placement Probability

Using Bayes rule:

$$P(T_i = A | L_i = d) = \frac{P(L_i = d, T_i = A)}{P(L_i = d)}$$

Probabilistic Interleaving: Placement Probability

Using Bayes rule:

$$
\begin{aligned}
P(T_i = A | L_i = d) &= \frac{P(L_i = d, T_i = A)}{P(L_i = d)} \\
&= \frac{P(L_i = d | T_i = A) P(T_i = A)}{P(L_i = d | T_i = A) P(T_i = A) + P(L_i = d | T_i = B) P(T_i = B)}
\end{aligned}
$$

Probabilistic Interleaving: Placement Probability

Using Bayes rule:

$$
\begin{aligned}
P(T_i = A | L_i = d) &= \frac{P(L_i = d, T_i = A)}{P(L_i = d)} \\
&= \frac{P(L_i = d | T_i = A) P(T_i = A)}{P(L_i = d | T_i = A) P(T_i = A) + P(L_i = d | T_i = B) P(T_i = B)} \\
&= \frac{P(L_i = d | T_i = A)}{P(L_i = d | T_i = A) + P(L_i = d | T_i = B)}
\end{aligned}
$$

Probabilistic Interleaving: Placement Probability

Using Bayes rule:

$$
\begin{aligned}
P(T_i = A | L_i = d) &= \frac{P(L_i = d, T_i = A)}{P(L_i = d)} \\
&= \frac{P(L_i = d | T_i = A) P(T_i = A)}{P(L_i = d | T_i = A) P(T_i = A) + P(L_i = d | T_i = B) P(T_i = B)} \\
&= \frac{P(L_i = d | T_i = A)}{P(L_i = d | T_i = A) + P(L_i = d | T_i = B)} \\
&= \frac{P_A(d)}{P_A(d) + P_B(d)}
\end{aligned}
$$

Probabilistic Interleaving: Placement Probability

Thus we can calculate the placement probability for each document.:

$$P(T_i = A) = \frac{1}{2} \tag{17}$$

$$P(L_i = d | T_i = A) = P_A(d) = \frac{\frac{1}{\textit{rank}(d, R_A)^\tau}}{\sum_{d' \in D} \frac{1}{\textit{rank}(d', R_A)^\tau}} \tag{18}$$

$$P(T_i = A | L_i = d) = \frac{P_A(d)}{P_A(d) + P_B(d)} \tag{19}$$

Probabilistic Interleaving: Placement Probability

Thus we can calculate the placement probability for each document.:

$$P(T_i = A) = \frac{1}{2} \tag{17}$$

$$P(L_i = d | T_i = A) = P_A(d) = \frac{\frac{1}{\textit{rank}(d, R_A)^\tau}}{\sum_{d' \in D} \frac{1}{\textit{rank}(d', R_A)^\tau}} \tag{18}$$

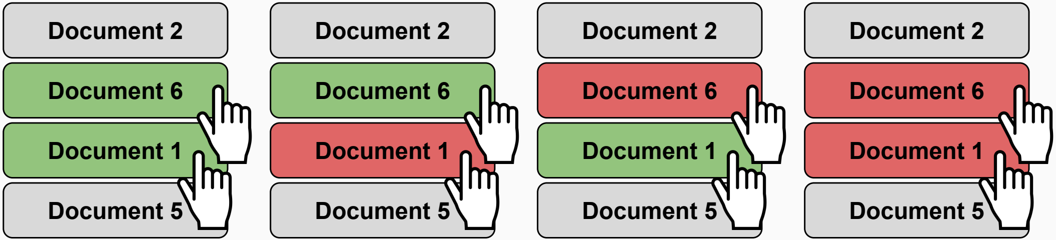
$$P(T_i = A | L_i = d) = \frac{P_A(d)}{P_A(d) + P_B(d)} \tag{19}$$

Two important observations:

- The outcome of a comparison is **only dependent on the clicked documents**.
- The assignment of a document is **not dependent on other assignments**.

Probabilistic Interleaving: Smarter Marginalization

Thus we only have to consider the possible assignments of clicked documents:



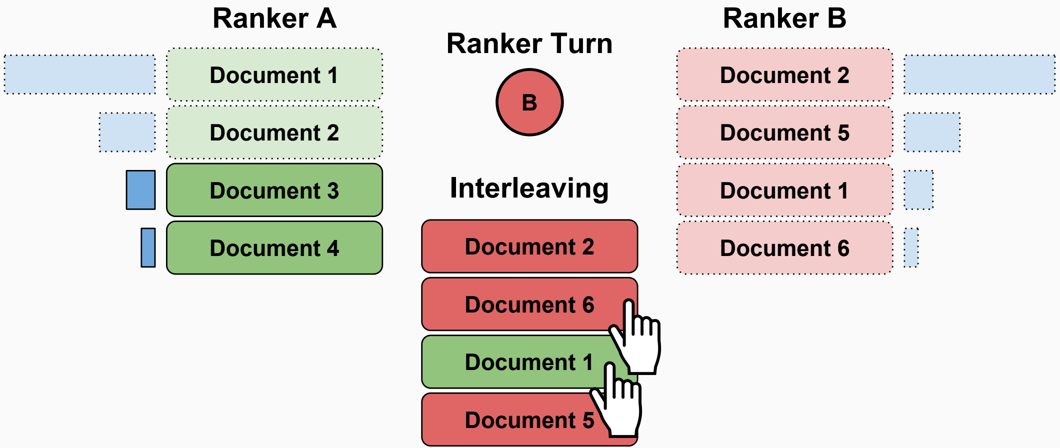
Bringing the complexity from 2^k to 2^c.

Probabilistic Interleaving: Method

This gives us the following method:

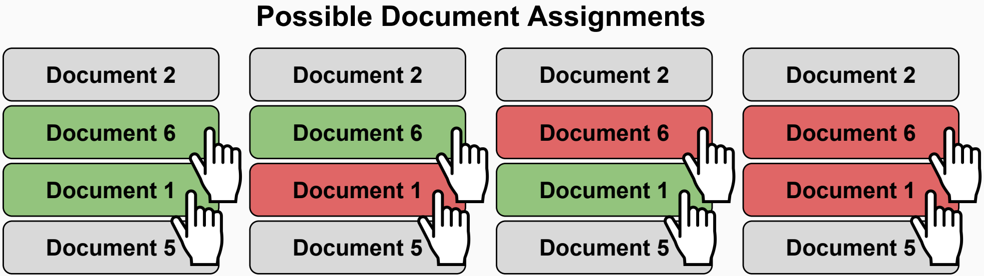
1. Compute P_A and P_B from ranker **A** and **B** respectively.
2. Repeat until k documents placed:
    1. Randomly choose P_A or P_B and sample a document d.
    2. Place d **without remembering** whether **A** or **B** was chosen.
    3. Renormalize P_A or P_B after removing d.
3. Display to user and observe clicks.
4. Calculate the **expected outcome** marginalizing over all possible placements.
5. Expected winner wins the comparison.

Probabilistic Interleaving: Visualization



Probabilistic Interleaving: Visualization

Probabilistic Interleaving: Visualization



Probabilistic Interleaving: Properties

Properties of Probabilistic Interleaving:

- **Correctness**:

Probabilistic Interleaving: Properties

Properties of Probabilistic Interleaving:

- **Correctness**:
  - Correct outcomes guaranteed w.r.t. fidelity.

Probabilistic Interleaving: Properties

Properties of Probabilistic Interleaving:

- **Correctness**:
    - Correct outcomes guaranteed w.r.t. fidelity.
    - **Marginalization does not affect the expected outcomes.**
    - Thus if proto-method has fidelity, so has this method.

Probabilistic Interleaving: Properties

Properties of Probabilistic Interleaving:

- **Correctness**:
  - Correct outcomes guaranteed w.r.t. fidelity.
  - **Marginalization does not affect the expected outcomes.**
  - Thus if proto-method has fidelity, so has this method.
- **User experience**:

Probabilistic Interleaving: Properties

Properties of Probabilistic Interleaving:

- **Correctness**:
  - Correct outcomes guaranteed w.r.t. fidelity.
  - **Marginalization does not affect the expected outcomes.**
  - Thus if proto-method has fidelity, so has this method.
- **User experience**:
  - User experience not guaranteed.

Probabilistic Interleaving: Properties

Properties of Probabilistic Interleaving:

- **Correctness**:
  - Correct outcomes guaranteed w.r.t. fidelity.
  - **Marginalization does not affect the expected outcomes.**
  - Thus if proto-method has fidelity, so has this method.
- **User experience**:
  - User experience not guaranteed.
  - **Every possible ranking can be displayed**, albeit with different probabilities.

Overview: Interleaving

	User Experience	Correctness	Source
Balanced Interleaving	✓		(Joachims, 2002a)
Team-Draft Interleaving	✓		(Radlinski et al., 2008)
Probabilistic Interleaving		✓	(Hofmann et al., 2011)

Optimized Interleaving

Optimized Interleaving

Introduced by Radlinski and Craswell (2013) casts interleaving as an **optimization problem**.

Interleavings should **only contain top-documents** from rankers, i.e. rankers should always add their top document.

First step: determine the set of **allowed interleavings**.

Optimized Interleaving: Allowed Interleavings

Allowed Interleavings

Ranker A	Ranker B
Document 1	Document 2
Document 2	Document 4
Document 3	Document 3
Document 4	Document 1

Optimized Interleaving: Allowed Interleavings

Allowed Interleavings

Document 1 | Document 2

Ranker A	Ranker B
Document 1	Document 2
Document 2	Document 4
Document 3	Document 3
Document 4	Document 1

Optimized Interleaving: Allowed Interleavings

Allowed Interleavings

Document 1	Document 2	Document 2
Document 2	Document 1	Document 4

Ranker A	Ranker B
Document 1	Document 2
Document 2	Document 4
Document 3	Document 3
Document 4	Document 1

Optimized Interleaving: Allowed Interleavings

Allowed Interleavings

Document 1	Document 1	Document 2
Document 2	Document 2	Document 1
Document 3	Document 4	Document 3

Document 2	Document 2	Document 2
Document 1	Document 4	Document 4
Document 4	Document 1	Document 3

Ranker A	Ranker B
Document 1	Document 2
Document 2	Document 4
Document 3	Document 3
Document 4	Document 1

Optimized Interleaving: Allowed Interleavings

Allowed Interleavings

Document 1	Document 1	Document 2
Document 2	Document 2	Document 1
Document 3	Document 4	Document 3
Document 4	Document 3	Document 4

Document 2	Document 2	Document 2
Document 1	Document 4	Document 4
Document 4	Document 1	Document 3
Document 3	Document 3	Document 1

Ranker A	Ranker B
Document 1	Document 2
Document 2	Document 4
Document 3	Document 3
Document 4	Document 1

Optimized Interleaving: Scoring Function

Optimized interleaving can use **different scoring functions** that meet its requirements. A click on a document d gives a **preference score** determined by how its ranked.

Common choices are:

1. **Linear Rank Difference**:

$$s(d, R_A, R_B) = \delta_d = \mathit{rank}(d, R_A) - \mathit{rank}(d, R_B) \tag{20}$$

2. **Inverse Rank Difference**:

$$s(d, R_A, R_B) = \delta_d = \frac{1}{\mathit{rank}(d, R_A)} - \frac{1}{\mathit{rank}(d, R_B)} \tag{21}$$

Optimized Interleaving: Scoring Function Example

Given two rankers where $R_A = [1, 2, 3, 4]$ and $R_B = [2, 4, 3, 1]$ and using the *Linear Rank Difference* scoring function, we see:

Interleaving	δ_{L_1}	δ_{L_2}	δ_{L_3}	δ_{L_4}
$[1, 2, 3, 4]$	3	-1	0	-2
$[1, 2, 4, 3]$	3	-1	-2	0
$[2, 1, 3, 4]$	-1	3	0	-2
$[2, 1, 4, 3]$	-1	3	-2	0
$[2, 4, 1, 3]$	-1	-2	3	0
$[2, 4, 3, 1]$	-1	-2	0	3

Optimized Interleaving: Position Bias Assumption

Let's assume that a user is **only position biased**, that means that only the position determines the **click probability**:

$$P(click(position))$$

Optimized Interleaving: Scoring Function Example

Given two rankers where $R_A = [1, 2, 3, 4]$ and $R_B = [2, 4, 3, 1]$ and using the *Linear Rank Difference* scoring function, **what distribution over interleavings should be chosen?**

Interleaving	δ_{L_1}	δ_{L_2}	δ_{L_3}	δ_{L_4}	$E[O]$	p_L
$[1, 2, 3, 4]$	3	-1	0	-2	$3P(click(1)) - P(click(2)) - 2P(click(4))$	
$[1, 2, 4, 3]$	3	-1	-2	0	$3P(click(1)) - P(click(2)) - 2P(click(3))$	
$[2, 1, 3, 4]$	-1	3	0	-2	$-P(click(1)) + 3P(click(2)) - 2P(click(4))$	
$[2, 1, 4, 3]$	-1	3	-2	0	$-P(click(1)) + 3P(click(2)) - 2P(click(3))$	
$[2, 4, 1, 3]$	-1	-2	3	0	$-P(click(1)) - 2P(click(2)) + 3P(click(3))$	
$[2, 4, 3, 1]$	-1	-2	0	3	$-P(click(1)) - 2P(click(2)) + 3P(click(4))$	

Optimized Interleaving: Optimization for Bias

If we take p_L for the **probability** of interleaving L being **displayed**, then the **expected outcome** can be written as:

$$E[O] = \sum_{L \in \mathcal{L}} \left(p_L \sum_{i=1}^{|L|} P(click(i)) s(L_i, R_A, R_B) \right) = 0 \tag{22}$$

This becomes a **linear optimization** (or linear programming) task to find a p_L to meet this **requirement**.

Optimized Interleaving: Scoring Function Example

Given two rankers where $R_A = [1, 2, 3, 4]$ and $R_B = [2, 4, 3, 1]$ and using the *Linear Rank Difference* scoring function, a possible solution is:

Interleaving	δ_{L_1}	δ_{L_2}	δ_{L_3}	δ_{L_4}	$E[O]$	p_L
$[1, 2, 3, 4]$	3	-1	0	-2	$3P(click(1)) - P(click(2)) - 2P(click(4))$	0%
$[1, 2, 4, 3]$	3	-1	-2	0	$3P(click(1)) - P(click(2)) - 2P(click(3))$	25%
$[2, 1, 3, 4]$	-1	3	0	-2	$-P(click(1)) + 3P(click(2)) - 2P(click(4))$	0%
$[2, 1, 4, 3]$	-1	3	-2	0	$-P(click(1)) + 3P(click(2)) - 2P(click(3))$	35%
$[2, 4, 1, 3]$	-1	-2	3	0	$-P(click(1)) - 2P(click(2)) + 3P(click(3))$	40%
$[2, 4, 3, 1]$	-1	-2	0	3	$-P(click(1)) - 2P(click(2)) + 3P(click(4))$	0%

Optimized Interleaving: Scoring Function Example

Given two rankers where $R_A = [1, 2, 3, 4]$ and $R_B = [2, 4, 3, 1]$ and using the *Linear Rank Difference* scoring function, a possible solution is:

Interleaving	δ_{L_1}	δ_{L_2}	δ_{L_3}	δ_{L_4}	$E[O]$	p_L
$[1, 2, 4, 3]$	3	-1	-2	0	$3P(click(1)) - P(click(2)) - 2P(click(3))$	25%
$[2, 1, 4, 3]$	-1	3	-2	0	$-P(click(1)) + 3P(click(2)) - 2P(click(3))$	35%
$[2, 4, 1, 3]$	-1	-2	3	0	$-P(click(1)) - 2P(click(2)) + 3P(click(3))$	40%

$$\begin{aligned}
P(click(1))(3 \times 0.25 - 1 \times 0.35 - 1 \times 0.4) &= 0 \\
P(click(2))(-0.25 + 3 \times 0.35 - 2 \times 0.4) &= 0 \\
P(click(3))(-2 \times 0.25 - 2 \times 0.35 + 3 \times 0.4) &= 0
\end{aligned}$$

Optimized Interleaving: Properties

Properties of Optimized Interleaving:

- **User experience**:
    - **Strongest guarantees** of all interleaving methods.

Optimized Interleaving: Properties

Properties of Optimized Interleaving:

- **User experience**:
  - **Strongest guarantees** of all interleaving methods.
- **Correctness**:
  - Method has **fidelity** (if optimized for it),

Optimized Interleaving: Properties

Properties of Optimized Interleaving:

- **User experience**:
  - **Strongest guarantees** of all interleaving methods.
- **Correctness**:
  - Method has **fidelity** (if optimized for it),
    as long as the linear optimization is successful.

Optimized Interleaving: Properties

Properties of Optimized Interleaving:

- **User experience**:
  - **Strongest guarantees** of all interleaving methods.
- **Correctness**:
  - Method has **fidelity** (if optimized for it),
    as long as the linear optimization is successful.
  - Proven by **brute-forcing** that there is **always a solution** for top-10 rankings.

Optimized Interleaving: Properties

Properties of Optimized Interleaving:

- **User experience**:
  - **Strongest guarantees** of all interleaving methods.
- **Correctness**:
  - Method has **fidelity** (if optimized for it), as long as the linear optimization is successful.
  - Proven by **brute-forcing** that there is **always a solution** for top-10 rankings.
  - Can be correct **under other definitions as well**.

Overview: Interleaving

	User Experience	Correctness	Source
Balanced	✓		(Joachims, 2002a)
Team-Draft	✓		(Radlinski et al., 2008)
Probabilistic		✓	(Hofmann et al., 2011)
Optimized	✓	✓	(Radlinski and Craswell, 2013)

Multileaving

Extending Interleaving

Interleaving provides a reliable way to compare two rankers.

However, in many cases **more than two rankers** need to be compared:

- Parameter tuning.
- Multiple teams researching & developing.

In these cases A/B testing would be **even more strenuous**.

Multileaving

Multileaving: extension of interleaving by Schuth et al. (2014).

Comparisons over a set of rankers $\mathcal{R} = \{A, B, \ldots\}$.

Goal of comparison is usually either:

- Find the **best ranker** in $\mathcal{R}$.
- Find the **preferences between every pair of rankers** in $\mathcal{R}$.

Fidelity for Multileaving: Condition #1

Condition 1 for fidelity:

- If user clicks are independent from document relevance, i.e. random clicks, then the interleaving method should not find any differences between any rankers.

Fidelity for Multileaving: Condition #1

Condition 1 for fidelity:

- If user clicks are independent from document relevance, i.e. random clicks, then the interleaving method should not find any differences between any rankers.

Rankers shouldn't have an **advantage** due to factors **other than relevance**.

This condition remains unchanged from interleaving.

Fidelity for Multileaving: Condition #2

Condition 2 for fidelity:

- If user clicks are correlated with document relevance,
  i.e. relevant documents are more likely to be clicked,

  then a ranker that pareto dominates all other rankers should be expected to win.

Fidelity for Multileaving: Condition #2

Condition 2 for fidelity:

- If user clicks are correlated with document relevance,
  i.e. relevant documents are more likely to be clicked,

  then a ranker that pareto dominates all other rankers should be expected to win.

An **unambiguous best ranker** should **always win** the comparison (given enough clicks).

Same as interleaving when there are only two rankers.
Not the strongest condition possible.

Fidelity for Multileaving: Conditions

Thus to have fidelity a method should:

1. Not give rankers an **advantage** due to factors **other than relevance**.
2. Always **prefer an unambiguous best ranker** in expectation.

Team-Draft Multileaving

Team-Draft Multileaving

A **straightforward extension of Team-Draft Interleaving** introduced by Schuth et al. (2014).

Same idea as Team-Draft interleaving:

- Let every ranker add a document in random order.
- Remember what ranker added which document.
- Rankers with more clicked documents are preferred over others.

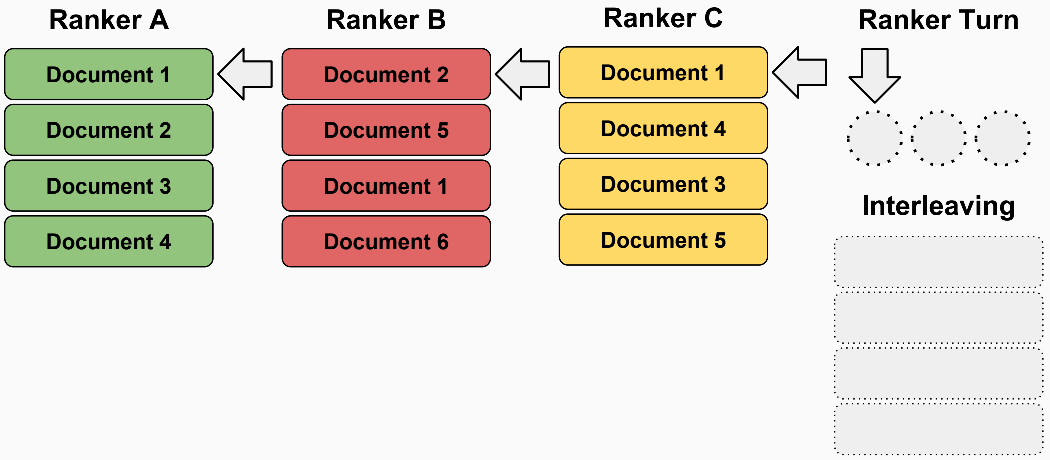
When $|\mathcal{R}| = 2$ it is **reduced** to Team-Draft interleaving.

Team-Draft Multileaving: Method

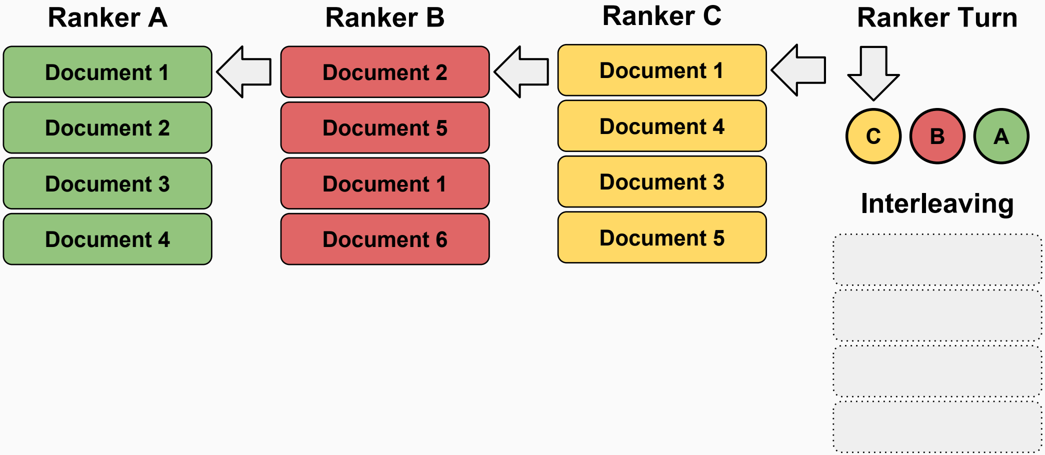
In plain English:

1. Until k documents are placed:
2.
   1. Create a **random permutation** of $\mathcal{R}$: $\hat{\mathcal{R}}$
   2. For every ranker X in order of $\hat{\mathcal{R}}$
   3.
      1. Let **ranker** X place its **next** *unplaced* document.
      2. **Remember** that ranker X placed this document.
3. Present interleaving to user, observe clicks.
4. Ranker with the most clicks on its placed document wins.

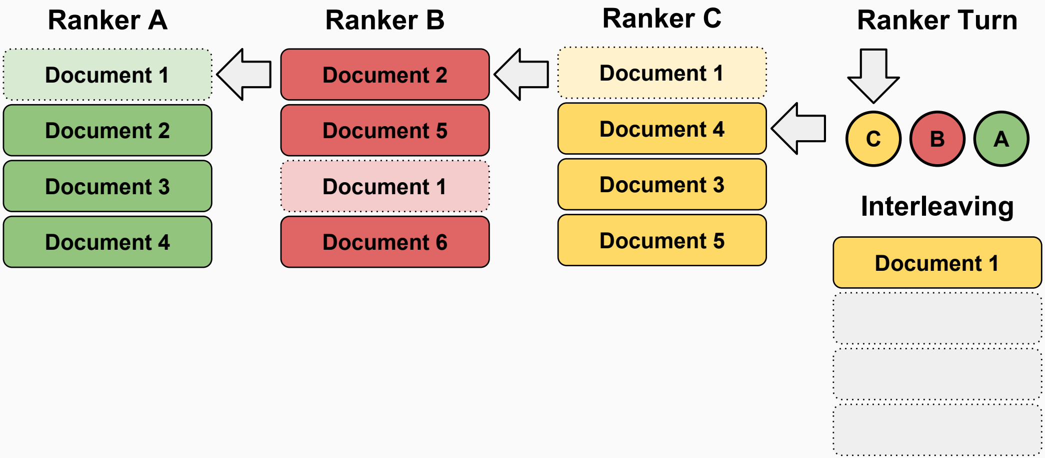
Team-Draft Multileaving: Visualized



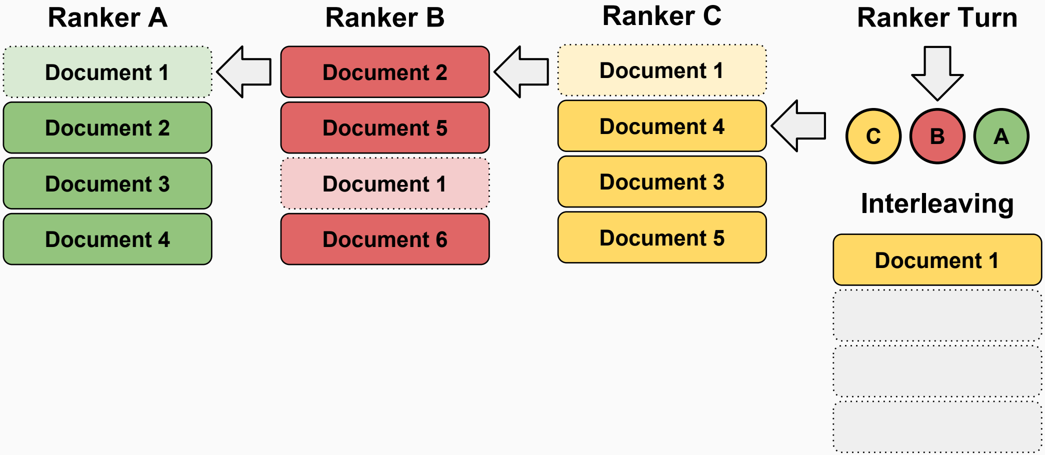
Team-Draft Multileaving: Visualized



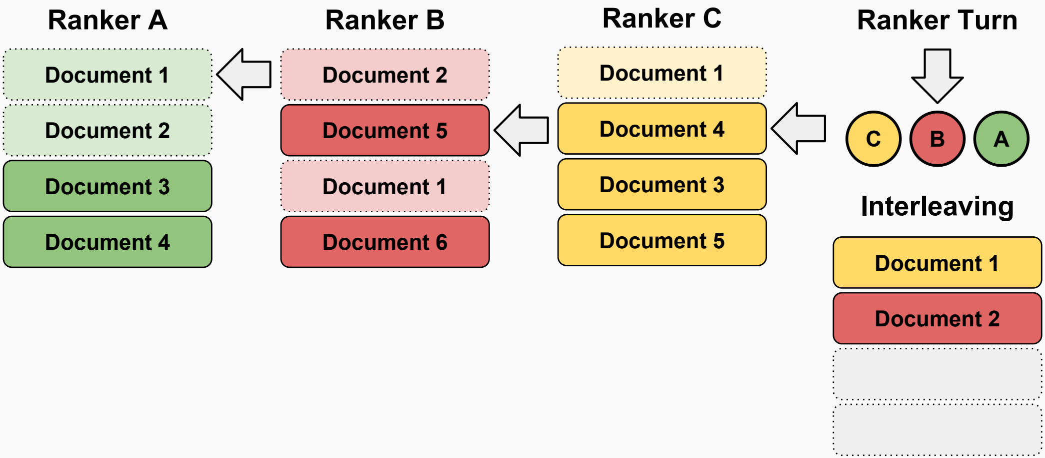
Team-Draft Multileaving: Visualized



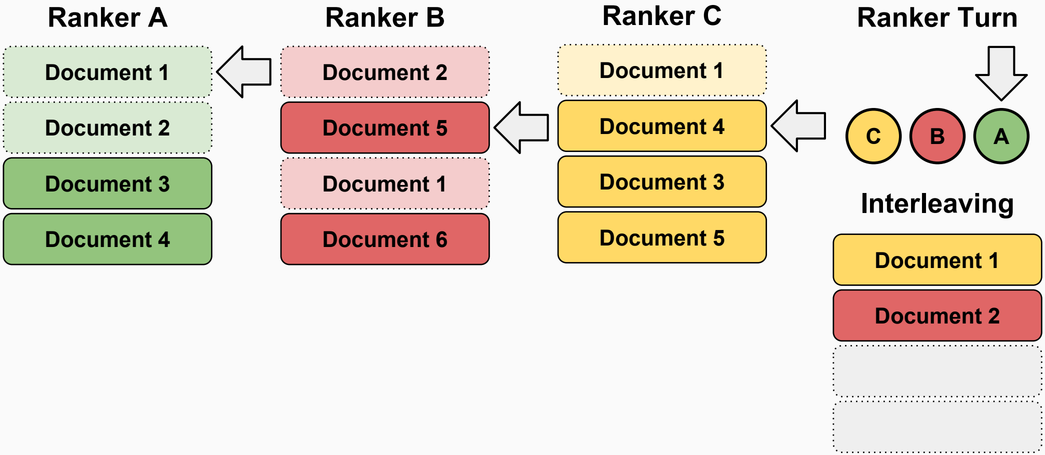
Team-Draft Multileaving: Visualized



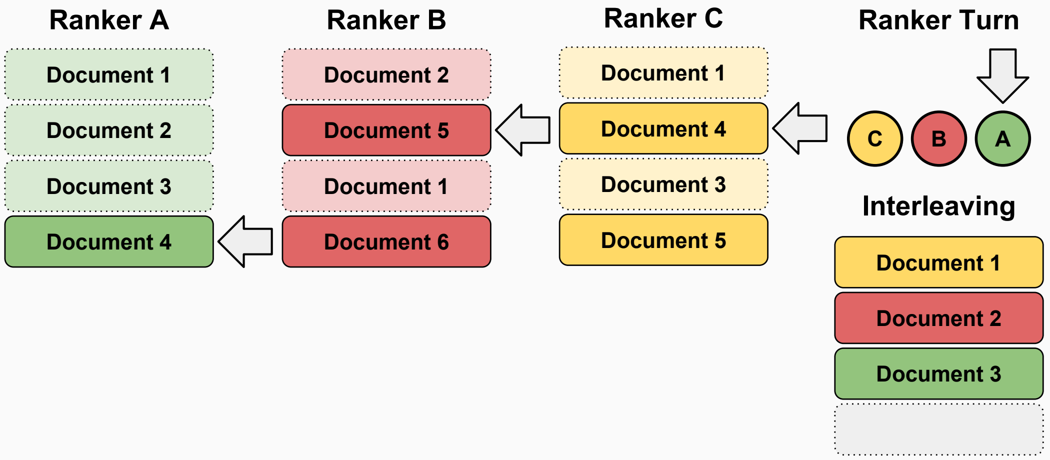
Team-Draft Multileaving: Visualized



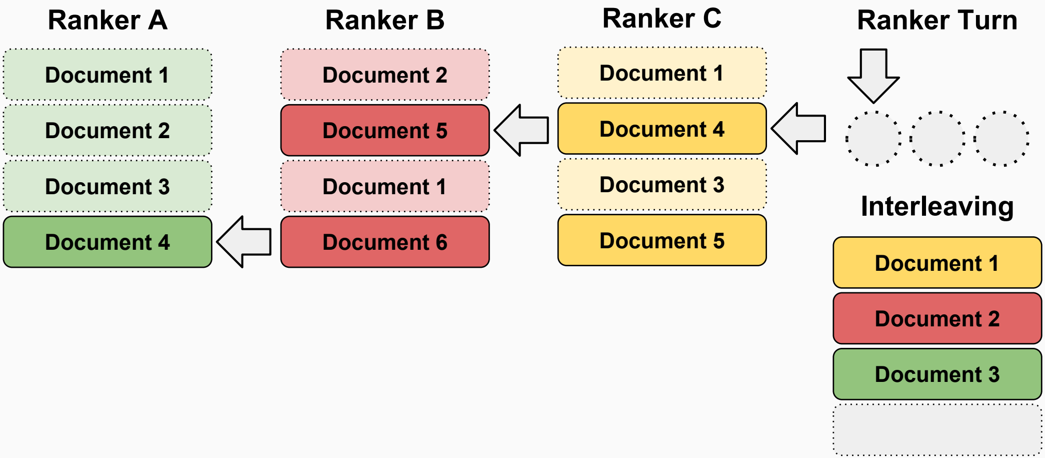
Team-Draft Multileaving: Visualized



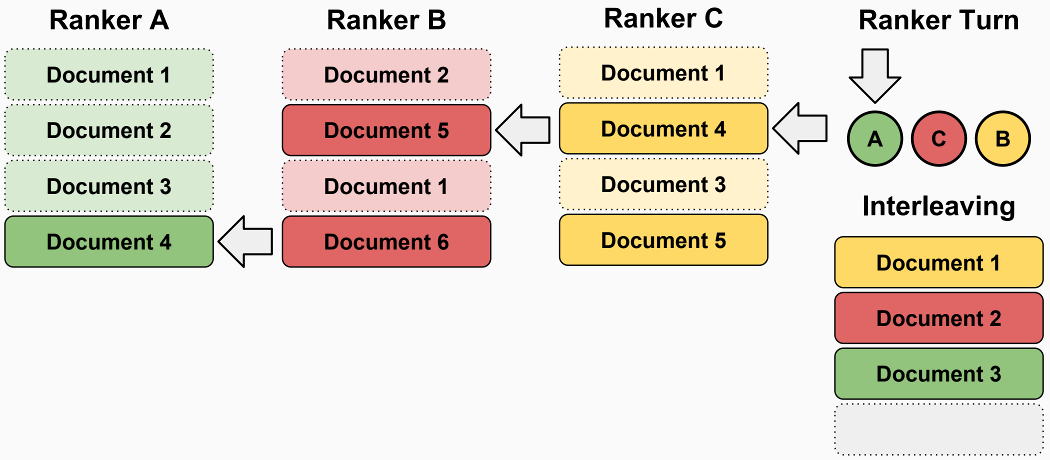
Team-Draft Multileaving: Visualized



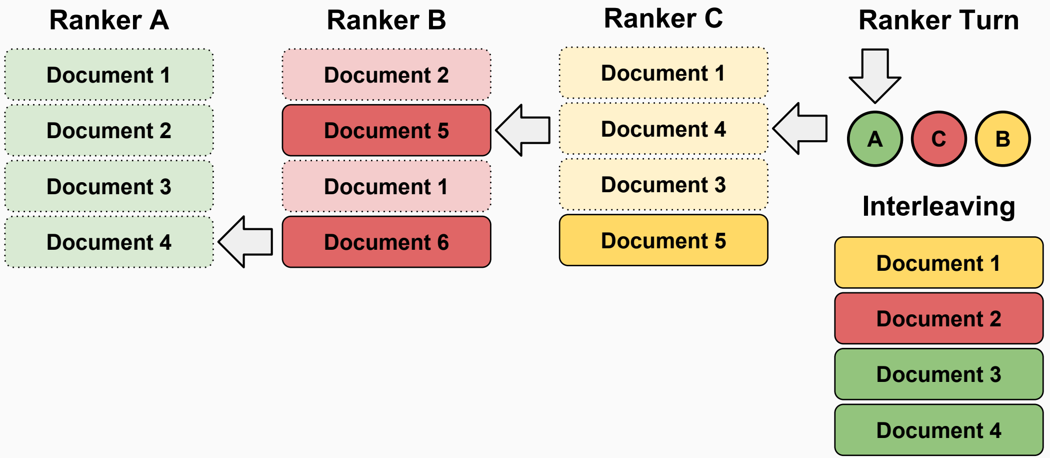
Team-Draft Multileaving: Visualized



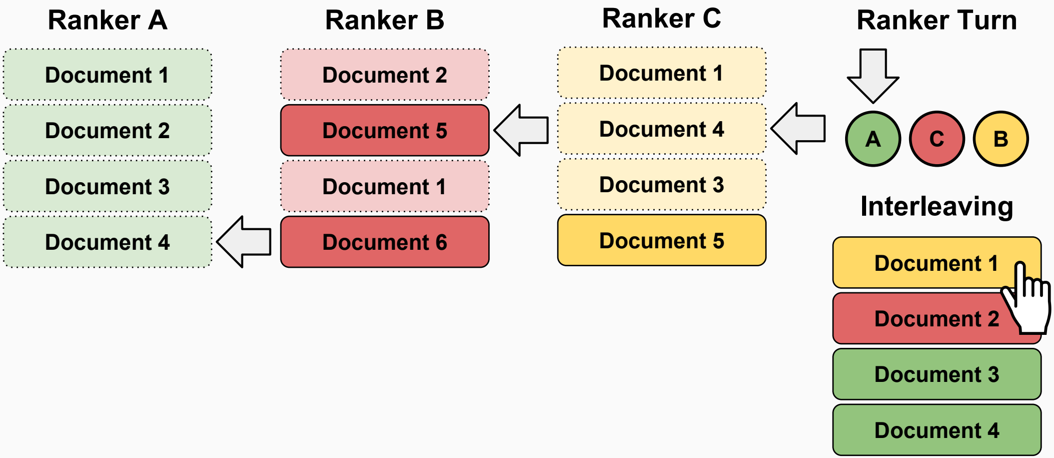
Team-Draft Multileaving: Visualized



Team-Draft Multileaving: Visualized



Team-Draft Multileaving: Visualized



Team-Draft Multileaving: Properties

Properties of Team-Draft Multileaving:

- **User experience**:

Team-Draft Multileaving: Properties

Properties of Team-Draft Multileaving:

- **User experience**:
    - **Not worse** than the worst ranker in $\mathcal{R}$.

Team-Draft Multileaving: Properties

Properties of Team-Draft Multileaving:

- **User experience**:
  - **Not worse** than the worst ranker in $\mathcal{R}$.
- **Correctness**:

Team-Draft Multileaving: Properties

Properties of Team-Draft Multileaving:

- **User experience**:
  - **Not worse** than the worst ranker in $\mathcal{R}$.
- **Correctness**:
  - **Same problems as Team-Draft Interleaving.**
  - Correctness not guaranteed.

Overview: Multileaving

	User Experience	Correctness	Computable	Source
Team-Draft	✓		✓	(Schuth et al., 2014)

Probabilistic Multileaving

Probabilistic Multileaving

Introduced by Schuth et al. (2015a),
at first glance, a **straightforward extension of probabilistic interleaving**.

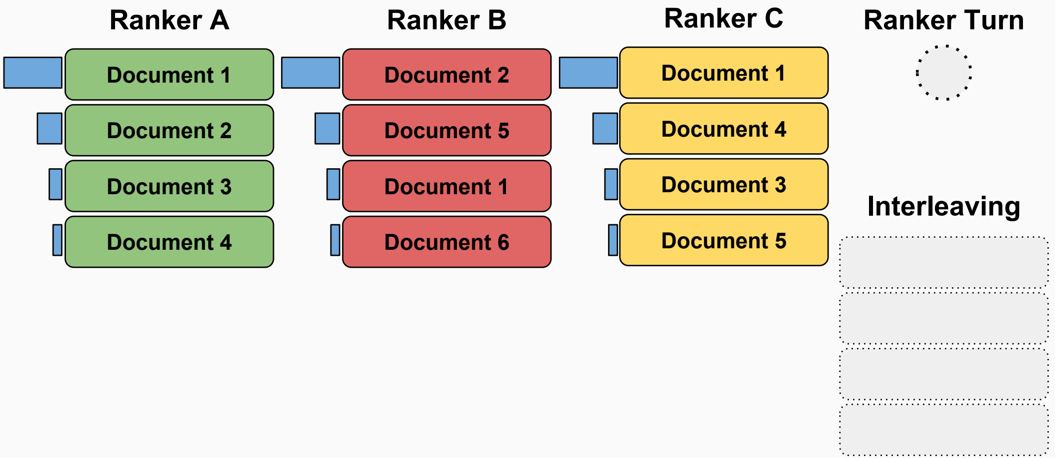
Rankers are interpreted as **distributions** again:

$$P_X(d) = \frac{\frac{1}{\textit{rank}(d, R_X)^\tau}}{\sum_{d' \in D} \frac{1}{\textit{rank}(d', R_X)^\tau}} \tag{23}$$

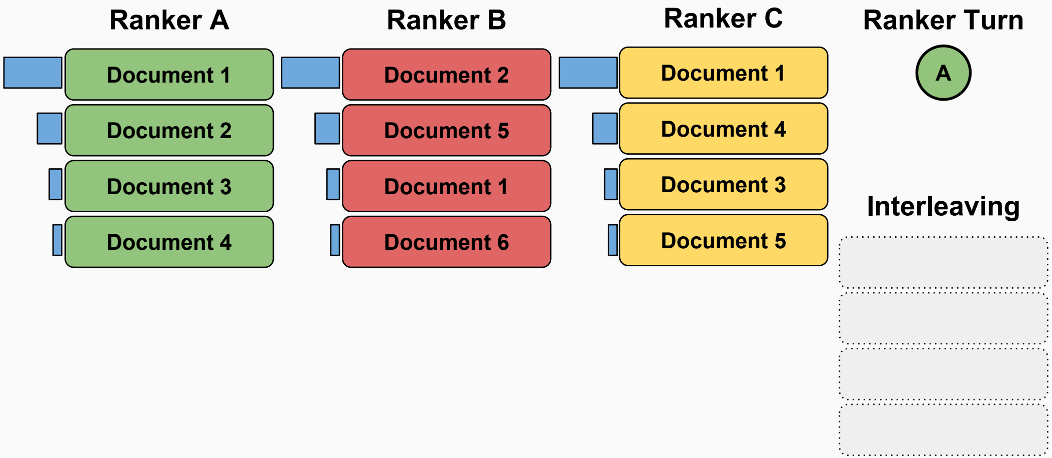
Probabilistic Multileaving: Method

1. Compute P_X for every $X \in \mathcal{R}$.
2. Repeat until k documents placed:
    1. Randomly choose P_X from $X \in \mathcal{R}$.
    2. Sample a document from $d \sim P_X$.
    3. Place d **without remembering which X was chosen**.
    4. Renormalize P_X after removing d for every $X \in \mathcal{R}$.
3. Display to user and observe clicks.
4. Calculate the **expected outcome marginalizing over the possible placements**.
5. Expected winners determine preferences.

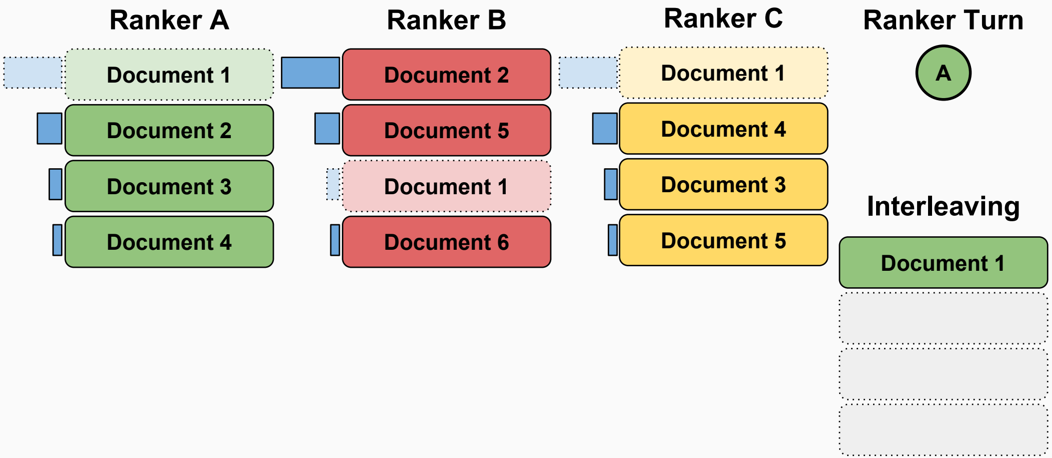
Probabilistic Multileaving: Visualization



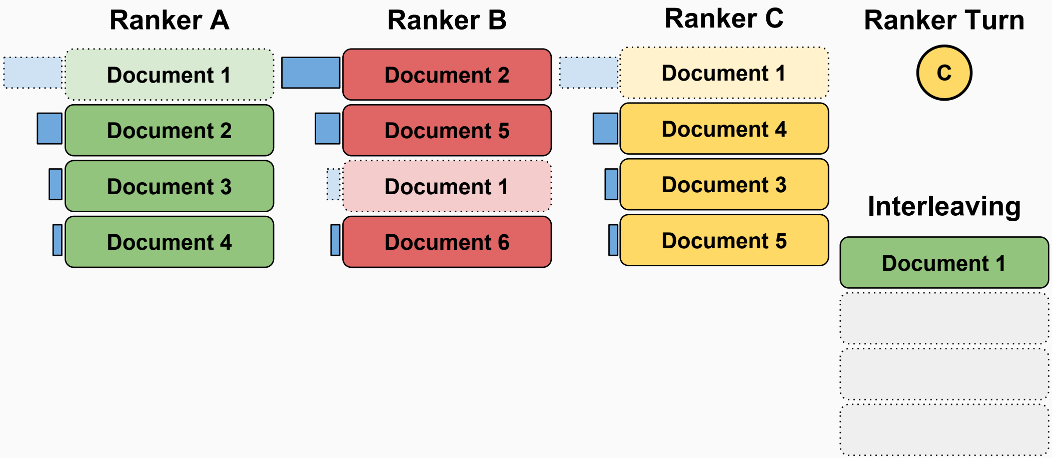
Probabilistic Multileaving: Visualization



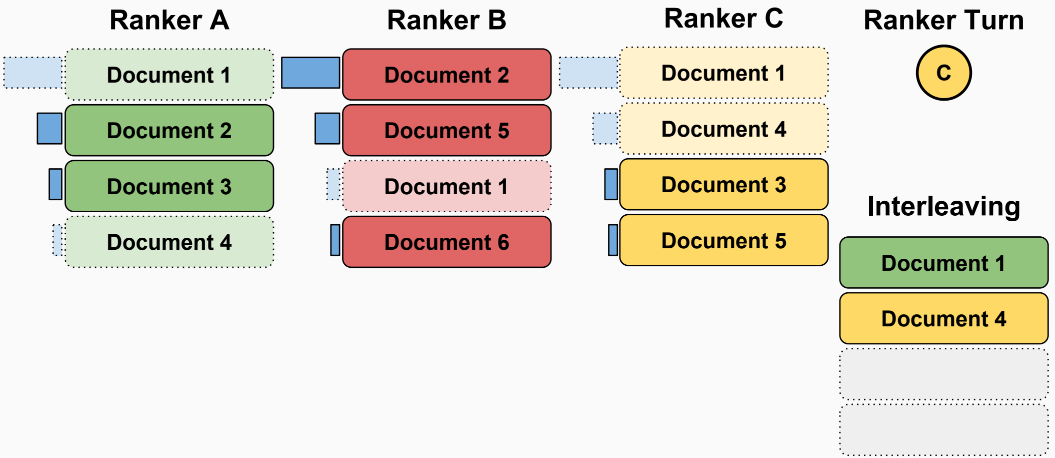
Probabilistic Multileaving: Visualization



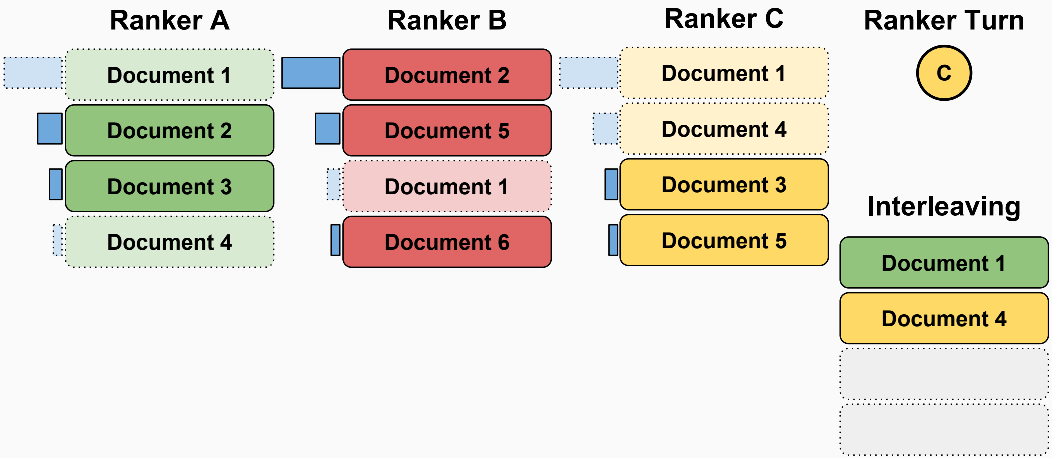
Probabilistic Multileaving: Visualization



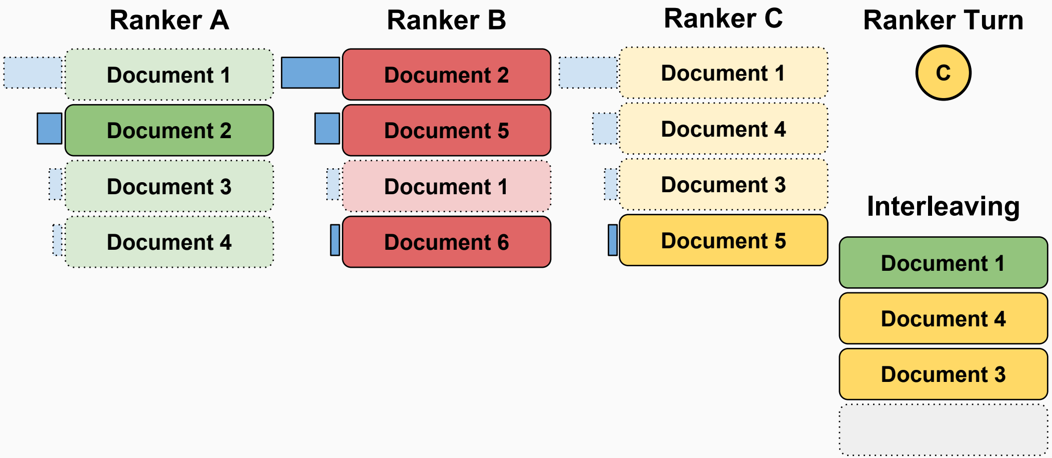
Probabilistic Multileaving: Visualization



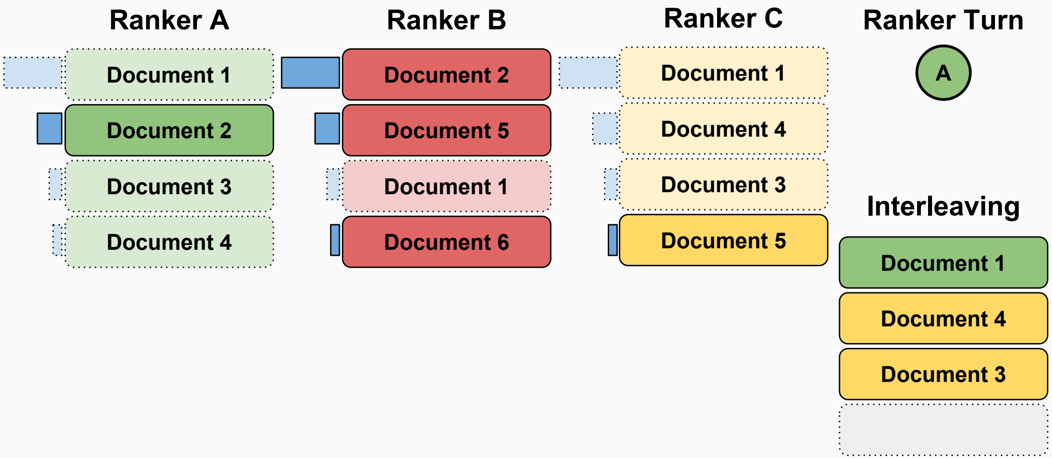
Probabilistic Multileaving: Visualization



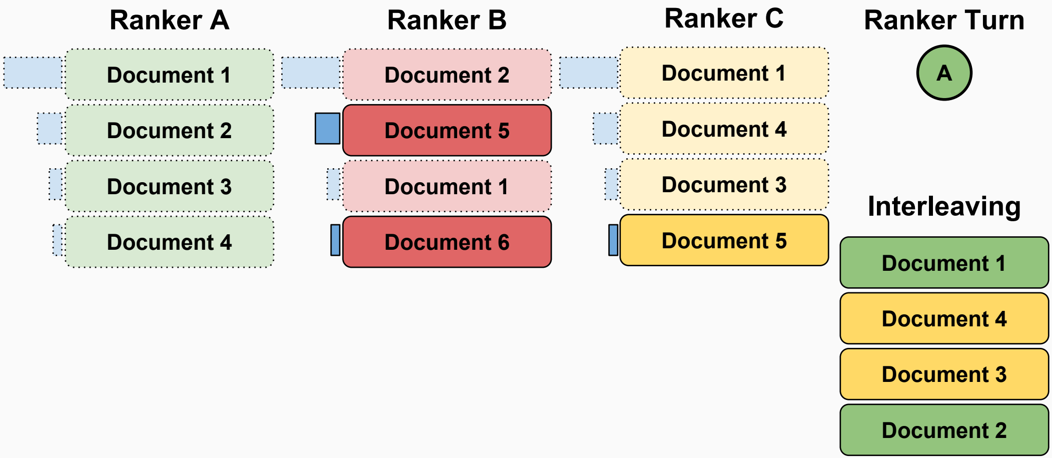
Probabilistic Multileaving: Visualization



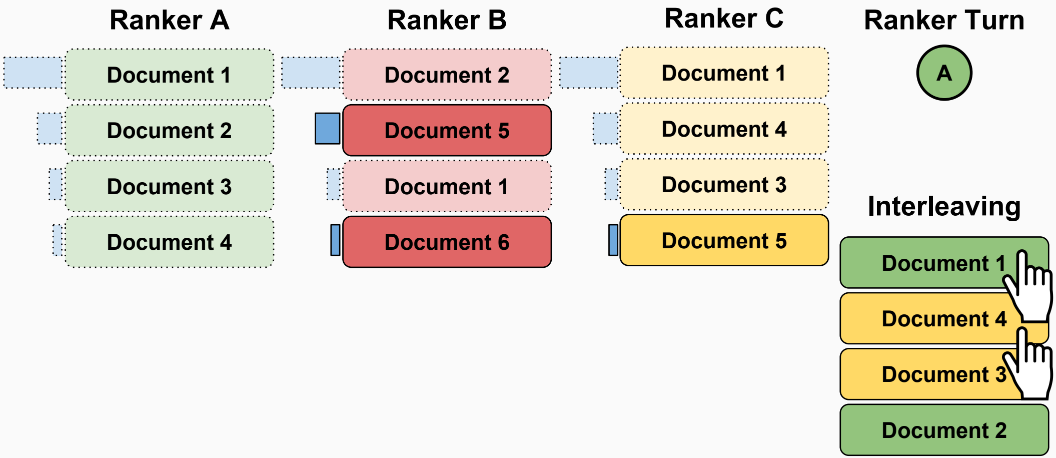
Probabilistic Multileaving: Visualization



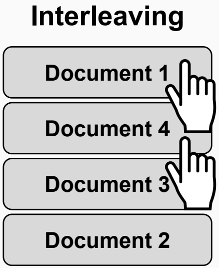
Probabilistic Multileaving: Visualization



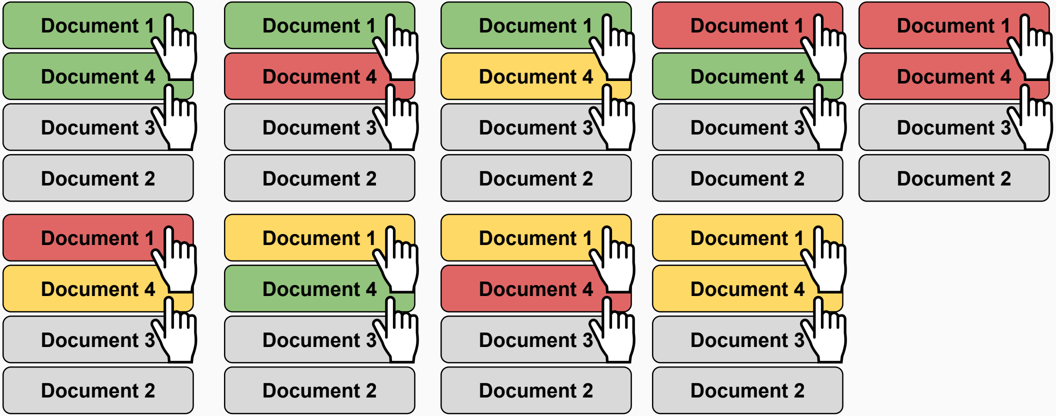
Probabilistic Multileaving: Visualization



Probabilistic Multileaving: Visualization



Probabilistic Multileaving: Visualization



Probabilistic Multileaving: Probabilities

For the interleaved list L, assignments T, and clicks c, the relevant probabilities can now be calculated as:

$$P(T_i = A) = \frac{1}{|\mathcal{R}|} \tag{24}$$

$$P(L_i = d | T_i = A) = P_A(d) = \frac{\frac{1}{\textit{rank}(d, R_A)^\tau}}{\sum_{d' \in D} \frac{1}{\textit{rank}(d', R_A)^\tau}} \tag{25}$$

$$P(T_i = A | L_i = d) = \frac{P_A(d)}{\sum_{X \in \mathcal{R}} P_X(d)} \tag{26}$$

Probabilistic Multileaving: Expected Outcome

Thus we can again compute the expected outcome O:

$$E[O(\mathcal{R}, L, c)] = \sum_{T} P(T|\mathcal{R}, L) O(\mathcal{R}, L, T, c) \tag{27}$$

Probabilistic Multileaving: Expected Outcome

Thus we can again compute the expected outcome O:

$$E[O(\mathcal{R}, L, c)] = \sum_{T} P(T|\mathcal{R}, L) O(\mathcal{R}, L, T, c) \tag{27}$$

What may be a problem here?

Probabilistic Multileaving: Expected Outcome

Thus we can again compute the expected outcome O:

$$E[O(\mathcal{R}, L, c)] = \sum_{T} P(T|\mathcal{R}, L)O(\mathcal{R}, L, T, c) \tag{27}$$

What may be a problem here?

The number of possible assignments is $|\mathcal{R}|^c$.
This is a problem for a **large number of rankers or clicked documents**.

Probabilistic Multileaving: Expected Outcome Approximation

Luckily the expected outcome O can be **approximated by sampling assignments**.

Let $\hat{\mathbf{T}}$ be a **set of assignments sampled** from $P(T|\mathcal{R}, L)$:

$$\hat{\mathbf{T}}_i \sim P(T|\mathcal{R}, L) \tag{28}$$

The **expected outcome** can then be **approximated** by:

$$E[O(\mathcal{R}, L, c)] \approx \sum_{T' \in \hat{\mathbf{T}}} O(\mathcal{R}, L, T', c) \tag{29}$$

Probabilistic Multileaving: Properties

Properties of Probabilistic Multileaving:

- **Correctness**:

Probabilistic Multileaving: Properties

Properties of Probabilistic Multileaving:

- **Correctness**:
  - Method has **provable fidelity**.
  - In expectation method the **preference of every ranker pair** will be correct.

Probabilistic Multileaving: Properties

Properties of Probabilistic Multileaving:

- **Correctness**:
  - Method has **provable fidelity**.
  - In expectation method the **preference of every ranker pair** will be correct.
- **User Experience**:

Probabilistic Multileaving: Properties

Properties of Probabilistic Multileaving:

- **Correctness**:
  - Method has **provable fidelity**.
  - In expectation method the **preference of every ranker pair** will be correct.
- **User Experience**:
  - **Every ranking possible.**
  - Hard to say what happens when $\mathcal{R}$ is large.

Probabilistic Multileaving: Properties

Properties of Probabilistic Multileaving:

- **Correctness**:
  - Method has **provable fidelity**.
  - In expectation method the **preference of every ranker pair** will be correct.
- **User Experience**:
  - **Every ranking possible.**
  - Hard to say what happens when $\mathcal{R}$ is large.
- **Computational Costs**:

Probabilistic Multileaving: Properties

Properties of Probabilistic Multileaving:

- **Correctness**:
  - Method has **provable fidelity**.
  - In expectation method the **preference of every ranker pair** will be correct.
- **User Experience**:
  - **Every ranking possible.**
  - Hard to say what happens when $\mathcal{R}$ is large.
- **Computational Costs**:
  - Becomes **quite high** for many clicks and rankers in $\mathcal{R}$.
  - Sampling assignments can be very expensive.

Overview: Multileaving

	User Experience	Correctness	Computable	Source
Team-Draft	✓		✓	(Schuth et al., 2014)
Probabilistic		✓	✓	(Schuth et al., 2015a)

Optimized Multileaving

Optimized Multileaving

Introduced by Schuth et al. (2014), straightforward extension of optimized interleaving.

Again different scoring functions can be chosen which create an optimization problem of a distribution over the allowed interleavings.

Optimized Interleaving: Scoring Function Example

Given **three** rankers where $R_A = [1, 2, 3, 4]$, $R_B = [2, 4, 3, 1]$ and $R_C = [3, 2, 4, 1]$ the allowed interleavings are:

$$
\begin{array}{cc}
[1, 2, 3, 4] & [1, 2, 4, 3] \\
[1, 3, 2, 4] & [2, 1, 3, 4] \\
[2, 1, 4, 3] & [2, 3, 1, 4] \\
[2, 3, 4, 1] & [2, 4, 1, 3] \\
[2, 4, 3, 1] & [3, 1, 2, 4] \\
[3, 2, 1, 4] & [3, 2, 4, 1]
\end{array}
$$

Optimized Multileaving: Optimization

If we take p_L for the **probability** of interleaving L being **displayed**, then the **expected outcome** can be written as:

$$E[O] = \sum_{X \in \mathcal{R}} \sum_{Y \in \mathcal{R}} \sum_{L \in \mathcal{L}} \left(p_L \sum_{i=1}^{|L|} P(click(i)) s(L_i, R_X, R_Y) \right) = 0 \tag{30}$$

The **complexity** of this problem is multiplied by the **number of pairs in $\mathcal{R}$** i.e. becomes **quadratically** more complex with $|\mathcal{R}|$.

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:
  - **Strongest of the multileaving method.**
  - Only displays top documents.

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:
  - **Strongest of the multileaving method.**
  - Only displays top documents.
- **Correctness**:

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:
  - **Strongest of the multileaving method.**
  - Only displays top documents.
- **Correctness**:
  - Method can be optimized for **fidelity**.
  - Also possible to optimize for **other definitions of correctness**.

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:
  - **Strongest of the multileaving method.**
  - Only displays top documents.
- **Correctness**:
  - Method can be optimized for **fidelity**.
  - Also possible to optimize for **other definitions of correctness**.
  - **Linear optimization not guaranteed to be solvable**, thus correctness not guaranteed in all cases.

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:
    - **Strongest of the multileaving method.**
    - Only displays top documents.
- **Correctness**:
    - Method can be optimized for **fidelity**.
    - Also possible to optimize for **other definitions of correctness**.
    - **Linear optimization not guaranteed to be solvable**, thus correctness not guaranteed in all cases.
- **Computational Costs**:

Optimized Multileaving: Properties

Properties of Optimized Multileaving:

- **User Experience**:
  - **Strongest of the multileaving method.**
  - Only displays top documents.
- **Correctness**:
  - Method can be optimized for **fidelity**.
  - Also possible to optimize for **other definitions of correctness**.
  - **Linear optimization not guaranteed to be solvable**, thus correctness not guaranteed in all cases.
- **Computational Costs**:
  - For **many rankers** in $\mathcal{R}$, the linear optimization problem can become **unmanageable**.

Overview: Multileaving

	User Experience	Correctness	Computable	Source
Team-Draft	✓		✓	(Schuth et al., 2014)
Probabilistic		✓	✓	(Schuth et al., 2015a)
Optimized	✓	✓	?	(Schuth et al., 2014)

Pairwise Preference Multileaving

Pairwise Preference Multileaving

Method designed **specifically for multileaving**, introduced recently by Oosterhuis and de Rijke (2017).

Based on inferring preferences between rankers from **preferences between document pairs**.

Pairwise Preference Multileaving: Document Preferences

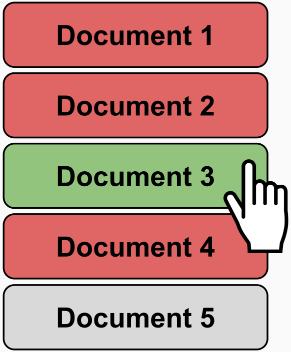
Users tend to **start at the top** of the result list and **work their way down**.

If a document is clicked but a **previous document is not**,
we can infer that the **user has a preference** between the two.

This assumption is well-established (Joachims, 2002b),
and famously used for pairwise learning to rank by Joachims (2002a).

Pairwise Preference Multileaving: Document Preferences Visualization

A clicked document is **inferred** to be **preferred** over the **previous unclicked** documents and the **first unclicked** document.



Pairwise Preference Multileaving: List Construction

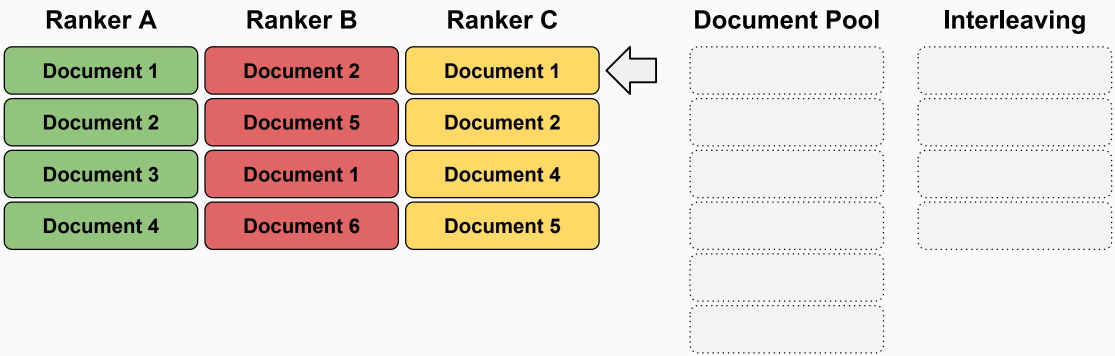
Pairwise Preference Multileaving **never places a document higher** than any ranker in $\mathcal{R}$.

At every rank there is a set of *'safe'* documents:

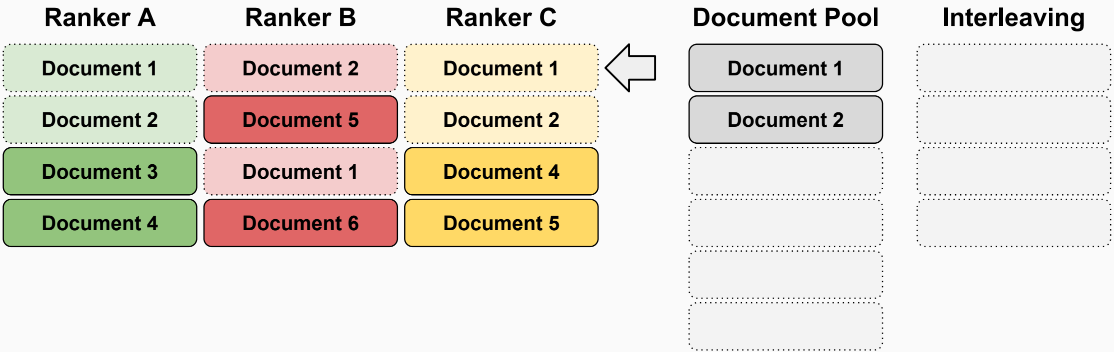
$$\boldsymbol{\Omega}(i, \mathcal{R}, D) = \{d | d \in D \wedge \exists X \in \mathcal{R}, \mathit{rank}(d, R_X) \leq i\} \tag{31}$$

Pairwise Preference Multileaving simply **samples from this set** at every rank (with the previously placed documents removed).

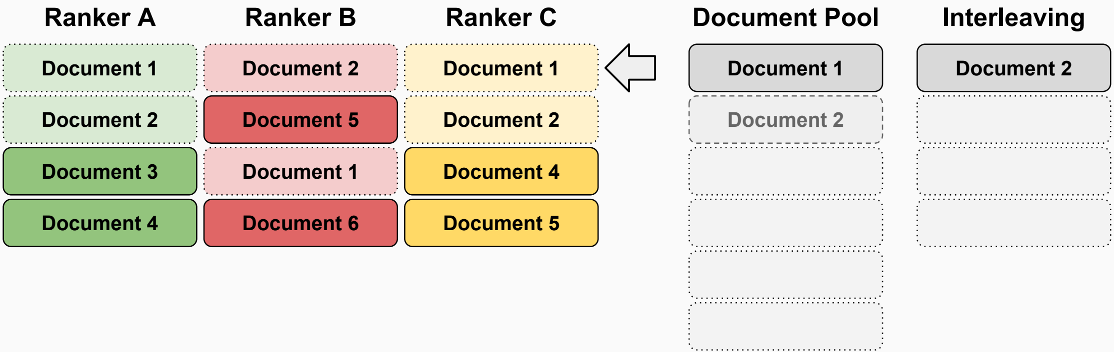
Pairwise Preference Multileaving: Visualization



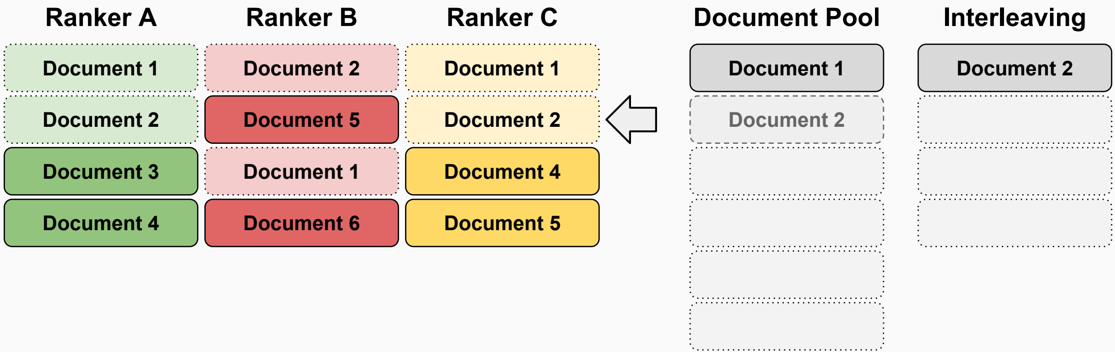
Pairwise Preference Multileaving: Visualization



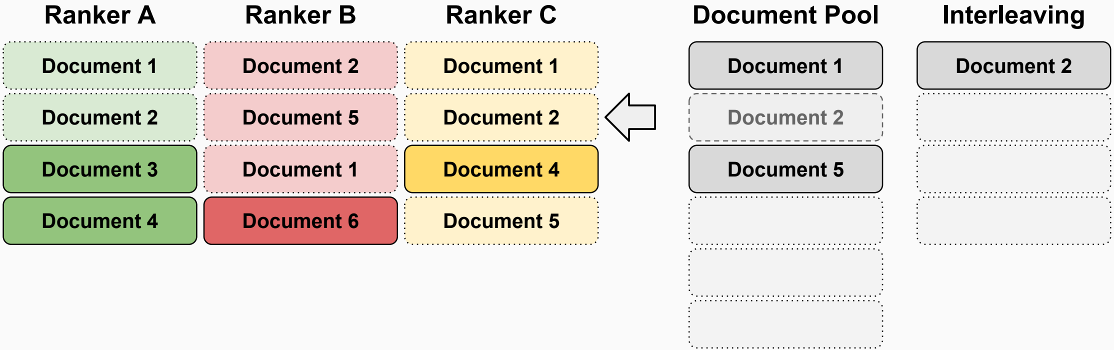
Pairwise Preference Multileaving: Visualization



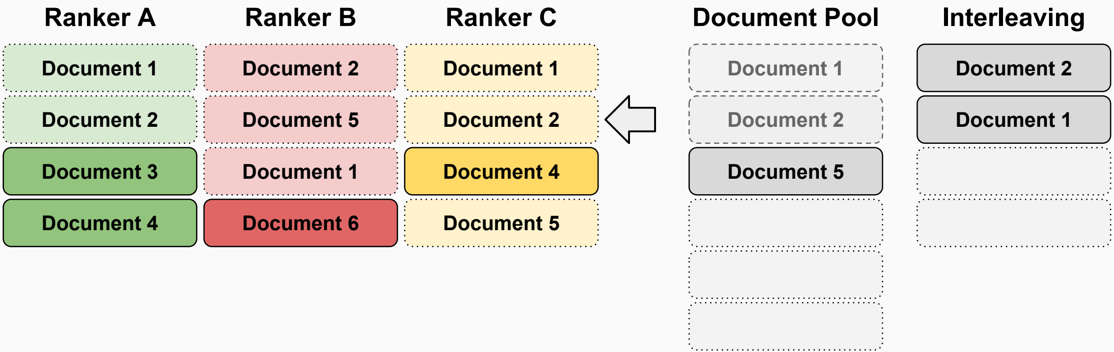
Pairwise Preference Multileaving: Visualization



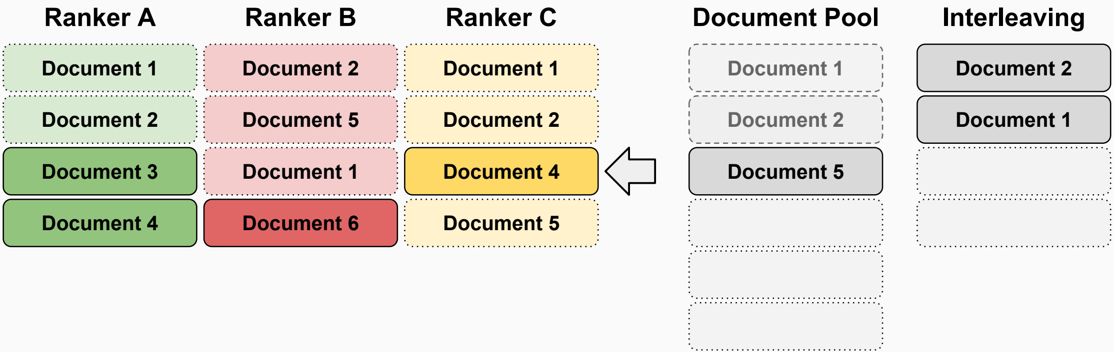
Pairwise Preference Multileaving: Visualization



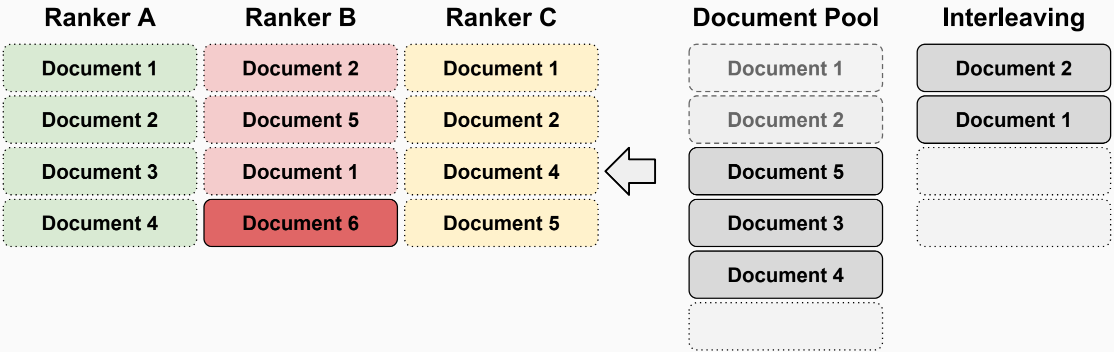
Pairwise Preference Multileaving: Visualization



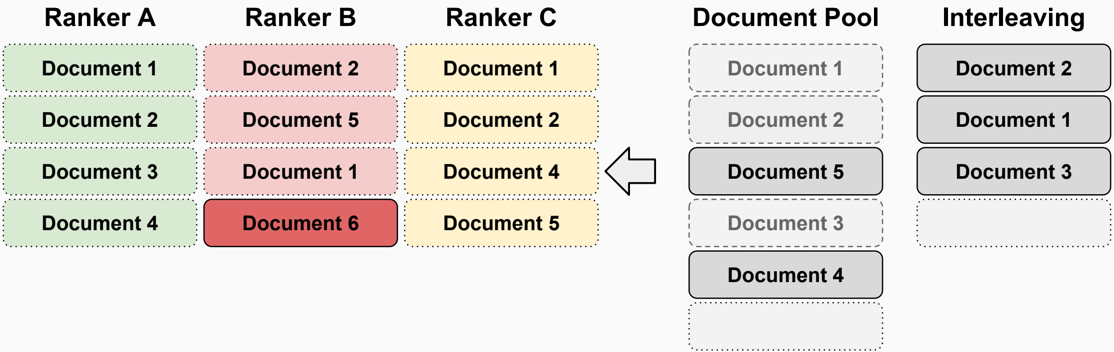
Pairwise Preference Multileaving: Visualization



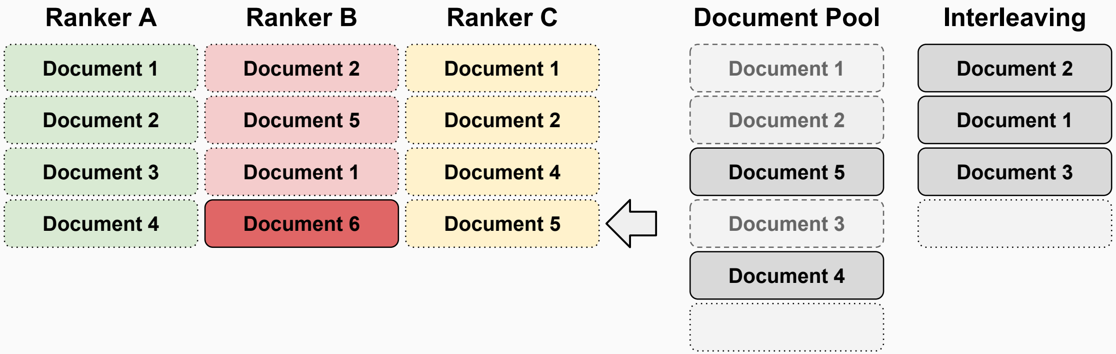
Pairwise Preference Multileaving: Visualization



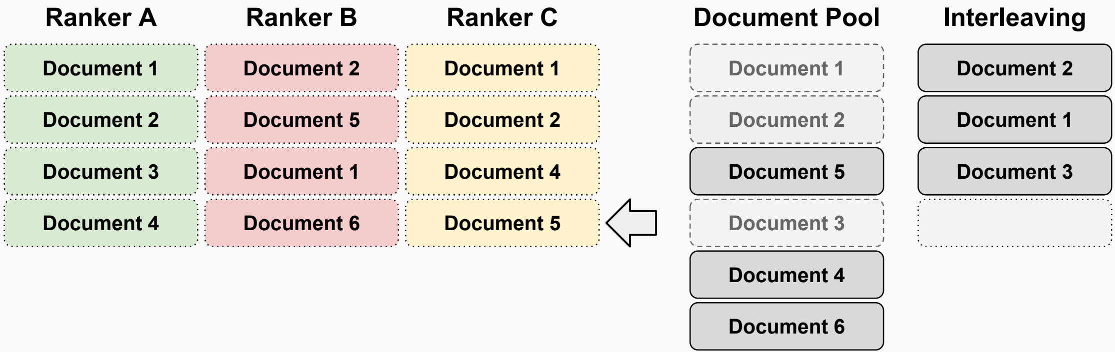
Pairwise Preference Multileaving: Visualization



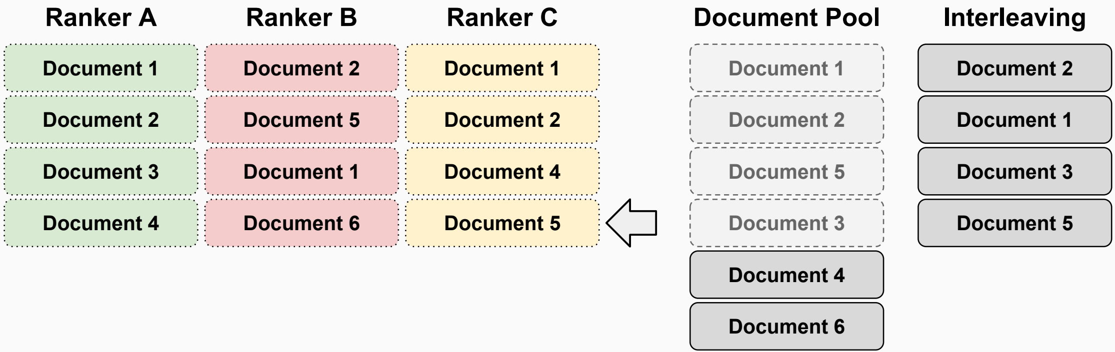
Pairwise Preference Multileaving: Visualization



Pairwise Preference Multileaving: Visualization



Pairwise Preference Multileaving: Visualization



Pairwise Preference Multileaving: Inference

Given **inferred document pair preferences**, we want to **give credit** to rankers that agree:

Ranker A	Ranker B	Ranker C
Document 1	**Document 2**	**Document 1**
Document 2	**Document 5**	**Document 2**
Document 3	**Document 1**	**Document 4**
Document 4	**Document 6**	**Document 5**

Document Preferences:	Ranker A	Ranker B	Ranker C
doc. 1 > doc. 2	correct	incorrect	correct
doc. 1 > doc. 5	correct	incorrect	correct
doc. 3 > doc. 2	incorrect	incorrect	incorrect
doc. 3 > doc. 5	correct	incorrect	incorrect

Interleaving

- **Document 2**
- **Document 1**
- **Document 3**
- **Document 5**

Pairwise Preference Multileaving: Inference

Given **inferred document pair preferences**, we want to **give credit** to rankers that agree:

Ranker A	Ranker B	Ranker C
Document 1	Document 2	Document 1
Document 2	Document 5	Document 2
Document 3	Document 1	Document 4
Document 4	Document 6	Document 5

Document Preferences:	Ranker A	Ranker B	Ranker C
doc. 1 > doc. 2	correct	incorrect	correct
doc. 1 > doc. 5	correct	incorrect	correct
doc. 3 > doc. 2	incorrect	incorrect	incorrect
doc. 3 > doc. 5	correct	incorrect	incorrect

Interleaving

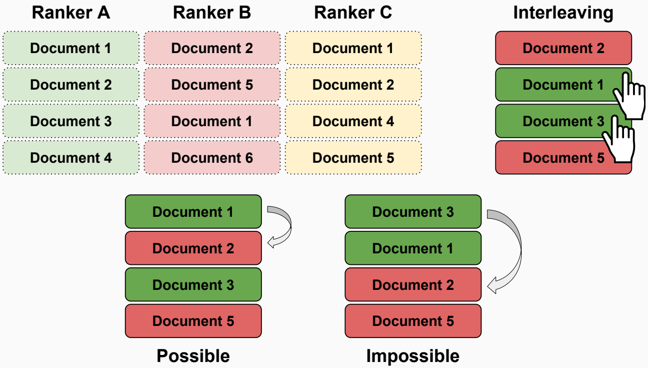
- Document 2
- Document 1
- Document 3
- Document 5

What may be a **problem** with this approach?

Pairwise Preference Multileaving: Symmetry

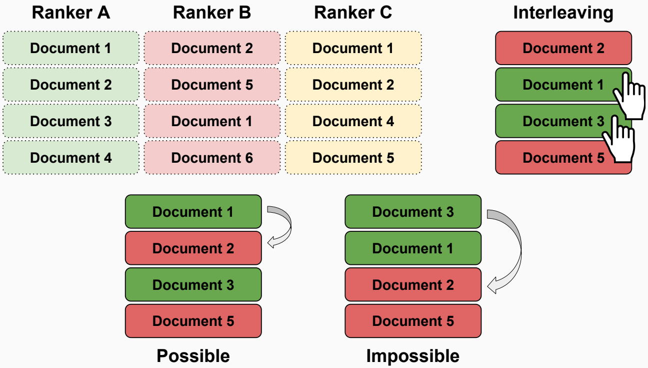
Some documents **cannot appear in certain places**,

thus **some preferences** will be **more likely** to be **observed**.



Pairwise Preference Multileaving: Symmetry

Some documents **cannot appear in certain places**,
thus **some preferences** will be **more likely** to be **observed**.



Solution: only give credit if the **same ranking** with the **documents flipped** is possible.

Pairwise Preference Multileaving: Pair bias

Some pairs are **more likely to appear where they are scored**.

Pairwise Preference Multileaving: Pair bias

Some pairs are **more likely to appear where they are scored**.

Solution: inversely weigh credit to the **probability of both documents appearing in the pool together**:

$$\phi(d_i, d_j, \mathbf{L}, \mathcal{R}) = \begin{cases} 0, & \textit{ranking with flipped pair is impossible} \\ \frac{1}{P(d_i \text{ and } d_j \text{ appear in pool together})}, & \textit{otherwise} \end{cases} \tag{32}$$

Pairwise Preference Multileaving: Fidelity

Fidelity of Pairwise Preference Multileaving can be **proven**, the general outline is:

- Document pairs with **equal click likelihood** do not affect the **expected preferences**:

Pairwise Preference Multileaving: Fidelity

Fidelity of Pairwise Preference Multileaving can be **proven**, the general outline is:

- Document pairs with **equal click likelihood** do not affect the **expected preferences**:
    - Under random clicks no preferences are found.

Pairwise Preference Multileaving: Fidelity

Fidelity of Pairwise Preference Multileaving can be **proven**, the general outline is:

- Document pairs with **equal click likelihood** do not affect the **expected preferences**:
  - Under random clicks no preferences are found.
  - Under correlated clicks, pairs with the same relevance have no effect.

Pairwise Preference Multileaving: Fidelity

Fidelity of Pairwise Preference Multileaving can be **proven**, the general outline is:

- Document pairs with **equal click likelihood** do not affect the **expected preferences**:
  - Under random clicks no preferences are found.
  - Under correlated clicks, pairs with the same relevance have no effect.
- Under correlated clicks, only pairs with **relevance differences give credit** in expectation:

Pairwise Preference Multileaving: Fidelity

Fidelity of Pairwise Preference Multileaving can be **proven**, the general outline is:

- Document pairs with **equal click likelihood** do not affect the **expected preferences**:
  - Under random clicks no preferences are found.
  - Under correlated clicks, pairs with the same relevance have no effect.
- Under correlated clicks, only pairs with **relevance differences give credit** in expectation:
  - Rankers that rank a **relevant document** at the **highest rank**:
    - Receive equal credit than rankers that rank the document the same.
    - Receive more credit than rankers that rank the document lower.

Pairwise Preference Multileaving: Fidelity

Fidelity of Pairwise Preference Multileaving can be **proven**, the general outline is:

- Document pairs with **equal click likelihood** do not affect the **expected preferences**:
  - Under random clicks no preferences are found.
  - Under correlated clicks, pairs with the same relevance have no effect.
- Under correlated clicks, only pairs with **relevance differences give credit** in expectation:
  - Rankers that rank a **relevant document** at the **highest rank**:
    - Receive equal credit than rankers that rank the document the same.
    - Receive more credit than rankers that rank the document lower.
- A **Pareto dominating ranker** ranks all documents at the **highest rank**, and **at least one higher** than every other ranker.
  - Thus the dominating ranker in $\mathcal{R}$ will receive the most credit.

Pairwise Preference Multileaving: Properties

Properties of Pairwise Preference Multileaving:

- **User experience**:
  - **Never** places a document **higher than any ranker** would.

Pairwise Preference Multileaving: Properties

Properties of Pairwise Preference Multileaving:

- **User experience**:
  - **Never** places a document **higher than any ranker** would.
- **Correctness**:
  - Has **proven Fidelity**.
  - A Pareto dominating ranker in $\mathcal{R}$ wins in expectation.

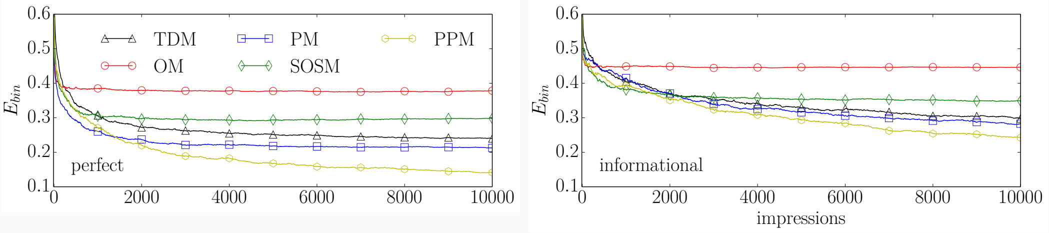
Pairwise Preference Multileaving: Properties

Properties of Pairwise Preference Multileaving:

- **User experience**:
  - **Never** places a document **higher than any ranker** would.
- **Correctness**:
  - Has **proven Fidelity**.
  - A Pareto dominating ranker in $\mathcal{R}$ wins in expectation.
- **Computational complexity**:
  - Very fast method, **polynomial complexity**.

Overview: Multileaving Simulations

Results from simulations with **15 rankers**, user behaviour **simulated** by simple click models, E_{bin} is **the ratio of errors** in preferences between ranker pairs (Oosterhuis and de Rijke, 2017).



Overview: Multileaving

	User Experience	Correctness	Computable	Source
Team-Draft	✓		✓	(Schuth et al., 2014)
Probabilistic		✓	✓	(Schuth et al., 2015a)
Optimized	✓	✓	?	(Schuth et al., 2014)
Pairwise-Preference	✓	✓	✓	(Oosterhuis and de Rijke, 2017)

Future of Online Evaluation

Future directions

Interleaving and Multileaving provide many ways to **reliably compare** ranking systems, however, there is still **room for improvement**:

Continuing on previous work:

- The guaranteed user experience of multileaving with fidelity can be better.
- No multileaving method that guarantees:
  1. a good user experience
  2. finds all preferences in expectation.

Future directions: Other Interesting Directions

Other **interesting directions** that could be further looked into:

- **Go beyond clicks**:
    - learn from other aspects of clicks (reaction time, dwell time, etc), how indicative a click is of a true preference.
    - See (Kharitonov et al., 2013; Yue et al., 2010).
- Further than the *ten blue links*:
    - For instance, how do we apply interleaving to grid-based displays, e.g. image search?
    - See (Kharitonov et al., 2015).

Future directions: Far Future

As we are able to interact with search systems in more ways,
user behaviour will become more complex
and better evaluation will be **necessary**.

As we get better at modelling users and proving properties of algorithms,
better evaluation will be **possible**.

Conclusion

Conclusion

Covered in the first part:

- Don't trust human annotators, **trust user interactions**.
- **Online approaches** can **effectively** and **reliably** make comparisons:
  - Be careful with **noise and bias in user interactions**.
  - Algorithms should **not interfere with user behaviour**.
  - Rankers should be **compared fairly**: unbiased and correctly.

What's next:

- Can we use the online approach to **optimize ranking systems**?

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

K. Hofmann, S. Whiteson, and M. De Rijke. A probabilistic method for inferring preferences from clicks. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 249–258. ACM, 2011.

K. Hofmann, S. Whiteson, and M. D. Rijke. Fidelity, soundness, and efficiency of interleaved comparison methods. *ACM Transactions on Information Systems (TOIS)*, 31(4):17, 2013.

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, 2002a.

T. Joachims. Unbiased evaluation of retrieval quality using clickthrough data. 2002b.

T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, volume 51, pages 4–11. Acm, 2017a.

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, 2017b.

E. Kharitonov, C. Macdonald, P. Serdyukov, and I. Ounis. Using historical click data to increase interleaving sensitivity. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 679–688. ACM, 2013.

E. Kharitonov, C. Macdonald, P. Serdyukov, and I. Ounis. Generalized team draft interleaving. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 773–782. ACM, 2015.

R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann. Online controlled experiments at large scale. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1168–1176. ACM, 2013.

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.

H. Oosterhuis and M. de Rijke. Sensitive and scalable online evaluation with theoretical guarantees. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 77–86. ACM, 2017.

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

F. Radlinski and N. Craswell. Optimized interleaving for online retrieval evaluation. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 245–254. ACM, 2013.

F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 43–52. ACM, 2008.

T. Sakai. On the reliability of information retrieval metrics based on graded relevance. *Information processing & management*, 43(2):531–548, 2007.

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

A. Schuth, F. Sietsma, S. Whiteson, D. Lefortier, and M. de Rijke. Multileaved comparisons for fast online evaluation. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 71–80. ACM, 2014.

A. Schuth, R.-J. Bruintjes, F. Buüttner, J. van Doorn, C. Groenland, H. Oosterhuis, C.-N. Tran, B. Veeling, J. van der Velde, R. Wechsler, et al. Probabilistic multileave for online retrieval evaluation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 955–958. ACM, 2015a.

A. Schuth, K. Hofmann, and F. Radlinski. Predicting search satisfaction metrics with interleaved comparisons. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 463–472. ACM, 2015b.

X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to rank with selection bias in personal search. In *SIGIR*, 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, 2018.

Y. Yue, Y. Gao, O. Chapelle, Y. Zhang, and T. Joachims. Learning more powerful test statistics for click-based retrieval evaluation. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 507–514. ACM, 2010.

Acknowledgments

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

Learning to Rank and Evaluation in the Online Setting - Online Learning to Rank

Harrie Oosterhuis

August 27, 2018

University of Amsterdam

oosterhuis@uva.nl

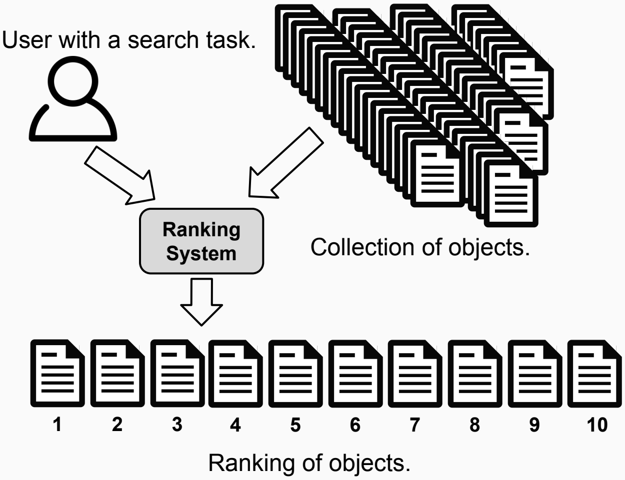
Introduction: Ranking Systems

Ranking Systems

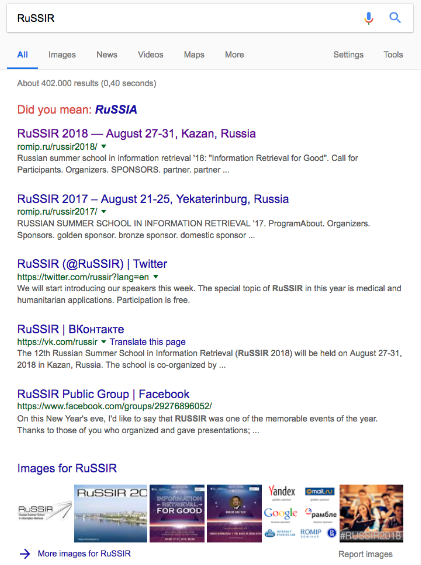
Let's go back to the beginning:

- Ranking systems are vital for **making the internet accessible**.
- They can present users **a small comprehensible selection** out of **millions of unordered results**.
- Search and recommendation are **practically everywhere**.

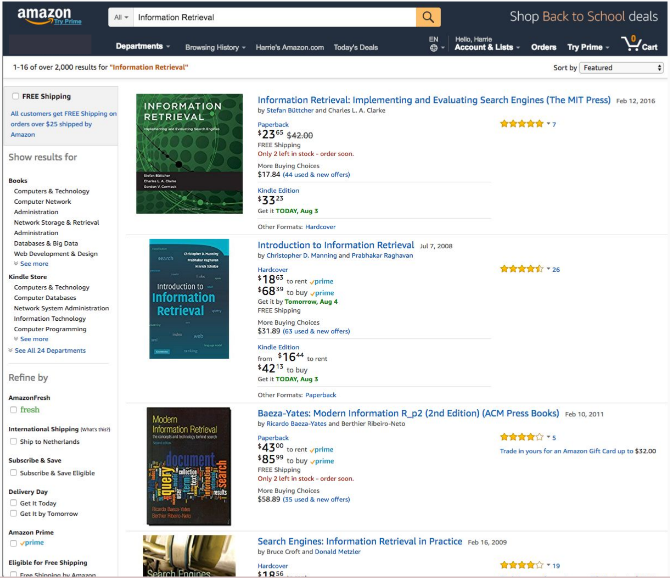
Ranking Systems: Schematic Example



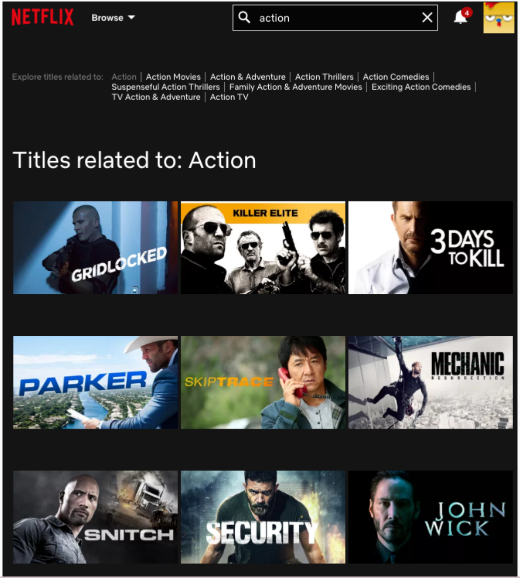
Ranking Systems: Examples



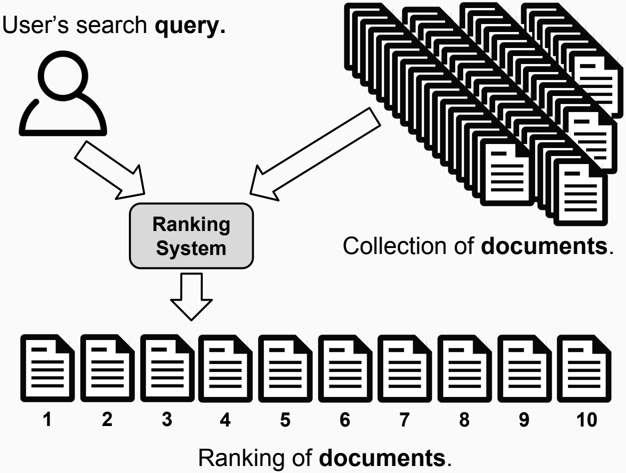
Ranking Systems: Examples



Ranking Systems: Examples



Ranking Systems: Schematic Example Naming



Importance of Ranking Quality

The quality of a ranking system is **very important** as it **directly impacts the user experience**.

Previously discussed:

- **Reliable evaluation** is important for improving a ranking system.

In this lecture:

- Algorithms that **automatically optimize** ranking systems, i.e. learning to rank.

Relevance Signals for Ranking

Relevance Signals: Introduction

The **big question** of information retrieval:

Is document d relevant for query q?

In other words, a function is desired that can **predict relevancy** given d and q:

$$f(q, d) = \textit{relevancy of document } d \textit{ w.r.t. query } q \tag{1}$$

Relevance Signals: Introduction

The **big question** of information retrieval:

Is document d relevant for query q?

In other words, a function is desired that can **predict relevancy** given d and q:

$$f(q, d) = \textit{relevancy of document } d \textit{ w.r.t. query } q \tag{1}$$

The **oldest** and **simplest** functions that **approximate** this are **relevance signals**:

Do you know any relevance signals?

Relevance Signals: Binary Matching

Simplest signal possible:

Does the query q appear in document d?

For a single word:

$$b(w, d) = \begin{cases} 1, & w \in d \\ 0, & w \notin d \end{cases}, \tag{2}$$

then for multiple words:

$$f(q, d) = \frac{1}{|q|} \sum_{w \in q} b(w, q). \tag{3}$$

Relevance Signals: Binary Matching

Simplest signal possible:

Does the query q appear in document d?

For a single word:

$$b(w,d) = \begin{cases} 1, & w \in d \\ 0, & w \notin d \end{cases}, \tag{2}$$

then for multiple words:

$$f(q,d) = \frac{1}{|q|} \sum_{w \in q} b(w,q). \tag{3}$$

What may be **problematic** with this signal?

Relevance Signals: TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) deals with **document length**, and the **rarity of words** in the document collection D:

How frequent is q in d and how frequent is q in D?

Relevance Signals: TF-IDF

The frequency of a word in a document, **term frequency**:

$$TF(w,d) = \frac{\textit{number of occurrences of } w \textit{ in document } d}{|d|}, \tag{4}$$

the frequency of a word in the document collection, **document frequency**:

$$DF(w,D) = \frac{\textit{number of documents in } D \textit{ where } w \in d}{|D|}, \tag{5}$$

then TD-IDF:

$$TF\text{-}IDF(q,d) = \frac{1}{|q|}\sum_{w\in q}\frac{TF(w,d)}{DF(w,D)}. \tag{6}$$

Relevance Signals: BM25

Okapi BM25 (Best-Matching) is another very famous relevance signal:

$$f(q,d) = \sum_{w \in q} DF(w,D)^{-1} \frac{TF(w,d)(k_1+1)}{TF(w,d) + k_1(1 - b + \frac{b \times |D|}{\textit{average document length}})} \tag{7}$$

Much more **complicated**, we will not get into the details now.

Relevance Signals: Other

Common relevance signals (applicable to different doc. parts, i.e. body, head, url):

1. Binary Matching
2. TF-IDF
3. BM25

Relevance Signals: Other

Common relevance signals (applicable to different doc. parts, i.e. body, head, url):

1. Binary Matching
2. TF-IDF
3. BM25
4. Language Models
5. Neural Models

Relevance Signals: Other

Common relevance signals (applicable to different doc. parts, i.e. body, head, url):

1. Binary Matching
2. TF-IDF
3. BM25
4. Language Models
5. Neural Models

Other useful signals:

Relevance Signals: Other

Common relevance signals (applicable to different doc. parts, i.e. body, head, url):

1. Binary Matching
2. TF-IDF
3. BM25
4. Language Models
5. Neural Models

Other useful signals:

1. Spam-detection
2. Page-Rank
3. Document quality/popularity

What is the signal we should use?

Relevance Signals: Conclusion

There is no relevance signal to rule them all.

For reference, the number of features in industry datasets:

Dataset	Feature Count	Reference
Microsoft Learning to Rank Web 30k	136	(Qin and Liu, 2013)
Yahoo! Webscope	471	(Chapelle and Chang, 2011)
Istella	220	(Dato et al., 2016)

What should we do with these signals then?

Relevance Signals: Conclusion

There is no relevance signal to rule them all.

For reference, the number of features in industry datasets:

Dataset	Feature Count	Reference
Microsoft Learning to Rank Web 30k	136	(Qin and Liu, 2013)
Yahoo! Webscope	471	(Chapelle and Chang, 2011)
Istella	220	(Dato et al., 2016)

What should we do with these signals then?

Combine all signals into a single model.

Relevance Signals: Combining All Signals

A document **representation** out of signals:

$$\mathbf{d} = \phi(d, q) = [\mathit{BM}(d, q), \mathit{TF\text{-}IDF}(d, q), \mathit{BM25}(d, q), \mathit{Page\text{-}Rank}(d), \mathit{Spam}(d), \ldots] \quad (8)$$

Then, **for instance**, a **linear model** can combine all signals:

$$f(\mathbf{d}, \boldsymbol{\theta}) = \sum_{i=1}^{|d|} \theta_i d_i. \quad (9)$$

How do we find $\boldsymbol{\theta}$?

Relevance Signals: Combining All Signals

A document **representation** out of signals:

$$\mathbf{d} = \phi(d, q) = [\mathit{BM}(d, q), \mathit{TF\text{-}IDF}(d, q), \mathit{BM25}(d, q), \mathit{Page\text{-}Rank}(d), \mathit{Spam}(d), \ldots] \quad (8)$$

Then, **for instance**, a **linear model** can combine all signals:

$$f(\mathbf{d}, \boldsymbol{\theta}) = \sum_{i=1}^{|d|} \theta_i d_i. \quad (9)$$

How do we find $\boldsymbol{\theta}$?

Using machine learning.

Traditional Learning to Rank

Traditional Learning to Rank: Pointwise

Pointwise approaches optimize models $f(\mathbf{d}, \boldsymbol{\theta})$ to **predict the relevancy** of a document, (Liu et al., 2009).

This can be cast as a **classification** or **regression** problem, e.g. the regression loss is:

$$\mathcal{L} = \sum_{\mathbf{d}} (f(\mathbf{d}, \boldsymbol{\theta}) - \textit{relevancy}(d, q))^2 \tag{10}$$

Traditional Learning to Rank: Pointwise

Pointwise approaches optimize models $f(\mathbf{d}, \boldsymbol{\theta})$ to **predict the relevancy** of a document, (Liu et al., 2009).

This can be cast as a **classification** or **regression** problem, e.g. the regression loss is:

$$\mathcal{L} = \sum_{\mathbf{d}} (f(\mathbf{d}, \boldsymbol{\theta}) - \textit{relevancy}(d, q))^2 \tag{10}$$

However, the model $f(\mathbf{d}, \boldsymbol{\theta})$ will be **used for ranking**, and the pointwise method **does not make use of that** application.

Traditional Learning to Rank: Pairwise

Pairwise approaches optimize models $f(\mathbf{d}, \boldsymbol{\theta})$ to **predict the order** of a **document pairs**, (Joachims, 2002).

A possible pairwise loss could be:

$$\mathcal{L} = \sum_{\mathbf{d} \succ \mathbf{d}'} f(\mathbf{d}', \boldsymbol{\theta}) - f(\mathbf{d}, \boldsymbol{\theta}) \tag{11}$$

Traditional Learning to Rank: Pairwise

Pairwise approaches optimize models $f(\mathbf{d}, \boldsymbol{\theta})$ to **predict the order** of a **document pairs**, (Joachims, 2002).

A possible pairwise loss could be:

$$\mathcal{L} = \sum_{\mathbf{d} \succ \mathbf{d}'} f(\mathbf{d}', \boldsymbol{\theta}) - f(\mathbf{d}, \boldsymbol{\theta}) \tag{11}$$

However, **users** will probably only look at **the top documents**, not all document pairs.

Traditional Learning to Rank: Listwise

Listwise approaches optimize models $f(\mathbf{d}, \boldsymbol{\theta})$ to directly maximize **ranking metrics**.

A possible listwise loss could look like:

$$\mathcal{L} = -NDCG(f(\cdot, \boldsymbol{\theta})) \tag{12}$$

Traditional Learning to Rank: Listwise

Listwise approaches optimize models $f(\mathbf{d}, \boldsymbol{\theta})$ to directly maximize **ranking metrics**.

A possible listwise loss could look like:

$$\mathcal{L} = -NDCG(f(\cdot, \boldsymbol{\theta})) \tag{12}$$

Unfortunately, **most IR metrics are non-differentiable** but this can be solved by **heuristic approaches**, e.g. Lambda-MART, (Burges, 2010).

Traditional Learning to Rank: Overview

Three categories of learning to rank methods:

- **Pointwise:** Optimize model to **directly predict relevancy** of documents.
- **Pairwise:** Optimize model to predict the **order of document pairs** correctly.
- **Listwise:** Optimize model to **(heuristically) increase ranking metric**.

What is the large weak point of all of these methods?

Traditional Learning to Rank: Overview

Three categories of learning to rank methods:

- **Pointwise:** Optimize model to **directly predict relevancy** of documents.
- **Pairwise:** Optimize model to predict the **order of document pairs** correctly.
- **Listwise:** Optimize model to **(heuristically) increase ranking metric**.

What is the large weak point of all of these methods?

They require annotated data.

Problems with Offline Learning to Rank

Similar to offline evaluation, offline learning to rank requires:

- A set of **queries**.
- A collection of **documents**.
- **Annotations** indicating the **relevance** between query and document pairs, or **similar annotations** that provide us the **best rankings**.

The **problems with annotated datasets**, discussed in the previous part, are still true.

Problems with Offline Evaluation

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

- **time consuming and 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 account for **future changes in relevancy** (Lefortier et al., 2014).
- **not necessarily aligned with actual user preferences** (Sanderson, 2010), i.e. annotators and users often disagree.

Learning from User Interactions

Learning from User Interactions

Online evaluation can **reliably infer ranker preferences** from user interactions, thus probably **learning to rank from user interactions** is also a good idea.

Remember:

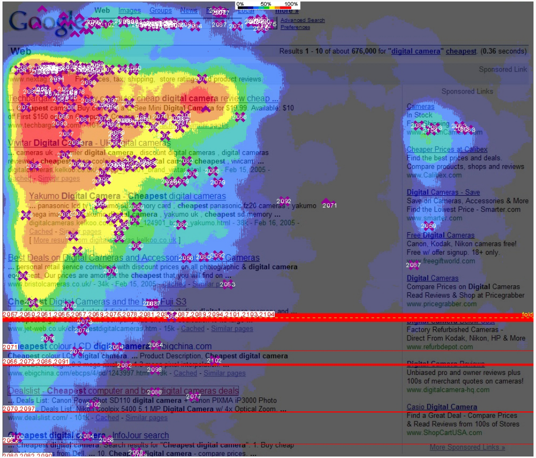
- Users **hate** giving **direct feedback**.
- **Implicit feedback** from users indicates their preferences.
- Learning from users solves the **problems with annotated datasets**.

Learning from User Interactions: Difficulties

Learning from user interactions brings its **own difficulties**:

- **Noise:** Users click on things for **unexpected reasons**.
- **Bias:** Interactions are affected by **factors other than relevancy**:
  - **Position bias:** **Higher ranked** documents get more attention.
  - **Selection bias:** Interactions are **limited** to the **presented** documents.

Learning from User Interactions: Golden Triangle



Source: http://www.mediative.com/

Learning from User Interactions: Conclusion

Learning to rank from user interactions has **large potential**:

- Learning from users solves the **problems with human annotators**.

Learning to rank from user interactions has **to deal** with:

- **Noise** in user behaviour.
- **Position Bias**: **Higher ranked** documents get more clicks.
- **Selection Bias**: Users will **only consider displayed** documents.

Related Approaches

Related Approaches: Learning from Click-Logs

The first paper on **learning from user interactions** (Joachims, 2002), also introduced the pairwise learning approach.

Infer **pairwise preferences between documents** from clicks in **historical interaction logs** and optimize a model to predict them correctly.

Though very **effective** this work does **not deal with selection bias**, and only **minimally with position bias**.

Related Approaches: Counter-Factual Learning to Rank

Recently a **counter-factual approach** was introduced by Joachims et al. (2017).

Extends a pointwise learning to rank approach to **take into account position bias**.

Related Approaches: Counter-Factual Learning to Rank

Recently a **counter-factual approach** was introduced by Joachims et al. (2017).

Extends a pointwise learning to rank approach to **take into account position bias**. Method assumes the **position bias is known or learned** and **independent from displayed documents**.

There is very recent work into **estimating the position bias from interaction data** Ai et al. (2018), still a **very active and upcoming** area of research.

Similarly **user modelling** have been shown **effective** for dealing with biases for learning to rank by Wang et al. (2018b).

Online Learning to Rank

Online Learning to Rank: Concept

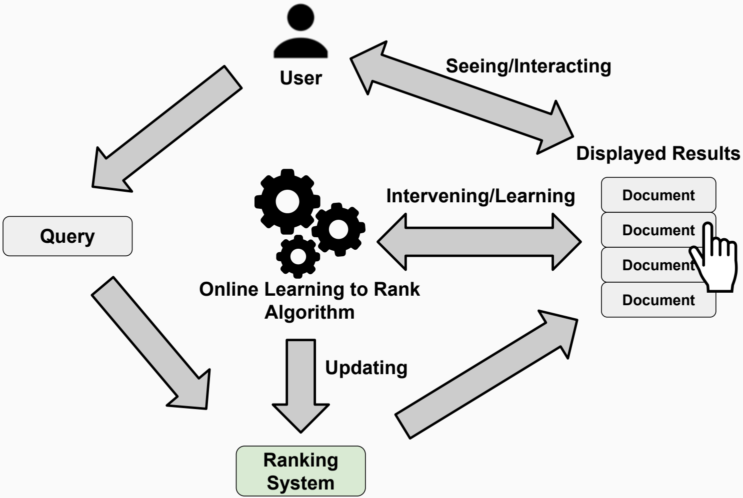
Online Learning to Rank methods have **control over what to display** to the user.

Simultantiously they:

- **Decide** what **results to display** to the user.
- **Learn** from **user interactions** with chosen results.

These methods can be much **more efficient**,
because they have (more) **control over what data is gathered**.

Online Learning to Rank: Visualization



Online Learning to Rank: Advantages

Online learning to rank methods have the **potential advantages**:

- Learn the **true preferences of users** (unlike annotator approaches).

Online Learning to Rank: Advantages

Online learning to rank methods have the **potential advantages**:

- Learn the **true preferences of users** (unlike annotator approaches).
- More **responsive** by **immediately adapting** to users.

Online Learning to Rank: Advantages

Online learning to rank methods have the **potential advantages**:

- Learn the **true preferences of users** (unlike annotator approaches).
- More **responsive** by **immediately adapting** to users.

What is a **large risk** for online learning to rank methods?

Online Learning to Rank: Advantages

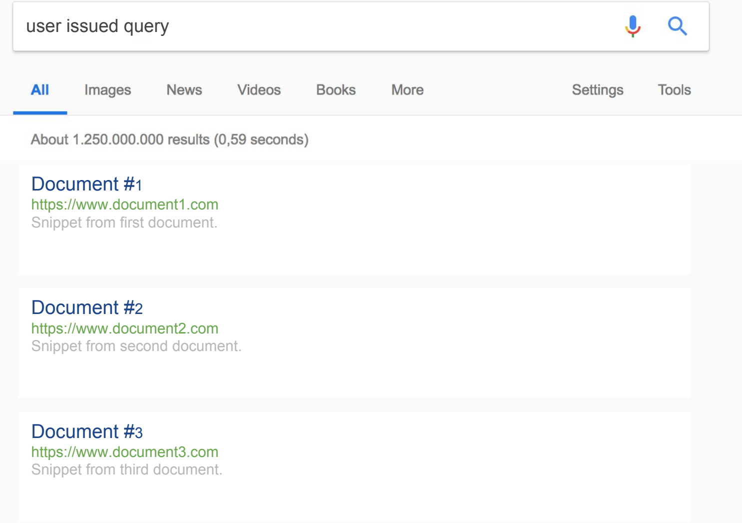
Online learning to rank methods have the **potential advantages**:

- Learn the **true preferences of users** (unlike annotator approaches).
- More **responsive** by **immediately adapting** to users.

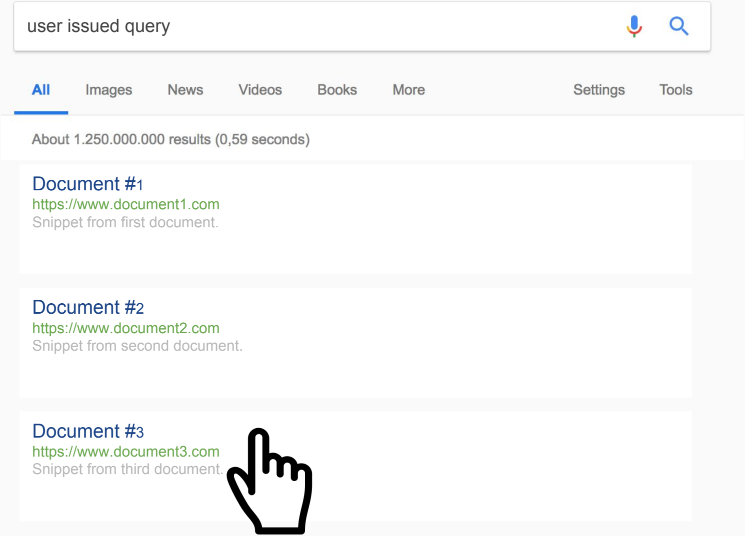
What is a **large risk** for online learning to rank methods?

- Unreliable methods could **severely worsen the user experience immediately**.

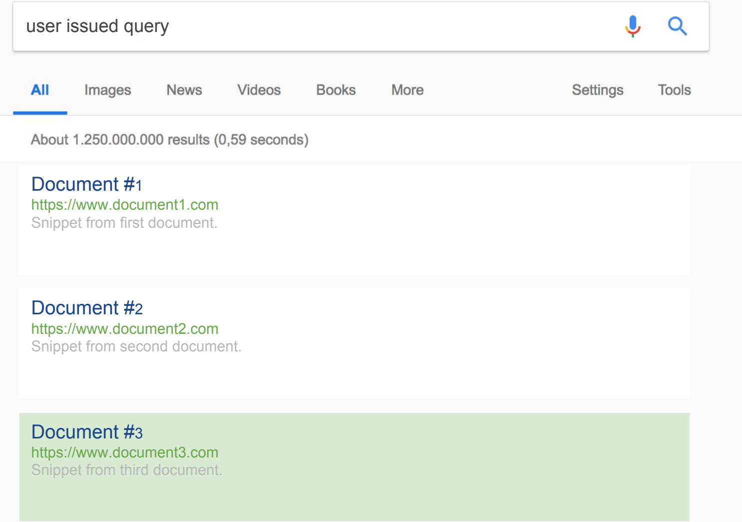
Online Learning to Rank: Example



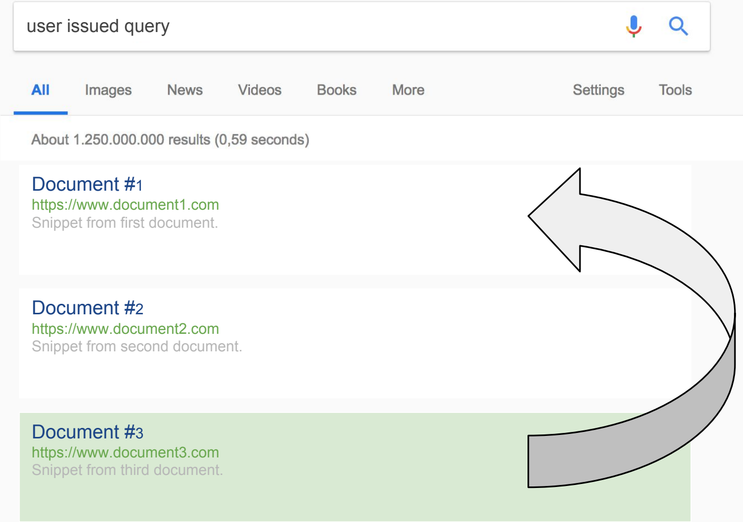
Online Learning to Rank: Example



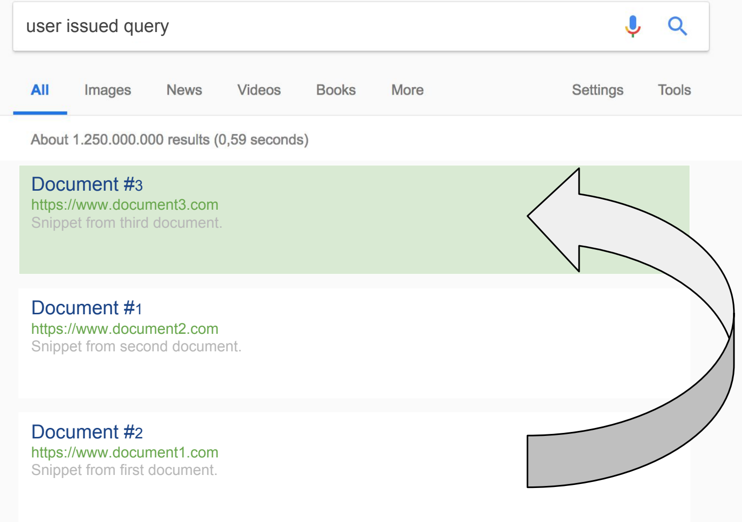
Online Learning to Rank: Example



Online Learning to Rank: Example



Online Learning to Rank: Example



Online Learning to Rank: Example

online learning

All Images News Videos Books More Settings Tools

About 1.250.000.000 results (0,59 seconds)

Khan Academy | Free Online Courses, Lessons & Practice
https://www.khanacademy.org/
You can learn anything. Expert-created content and resources for every course and level. Always free.

50 Top Online Learning Sites - Best College Reviews
https://www.bestcollegereviews.org/50-top-online-learning-sites/
Online learning may not appeal to everyone; however, the sheer number of online learning sites suggests that there is at least a strong interest in convenient, ...

The 5 Most Shocking Things I Learned About Amsterdam Online
https://www.clickbait.com
A juicy shocking list of five things that are probably not true with a title that will get a lot of attention. You will regret wasting your time on this.

Online Learning to Rank: Example

online learning

All Images News Videos Books More Settings Tools

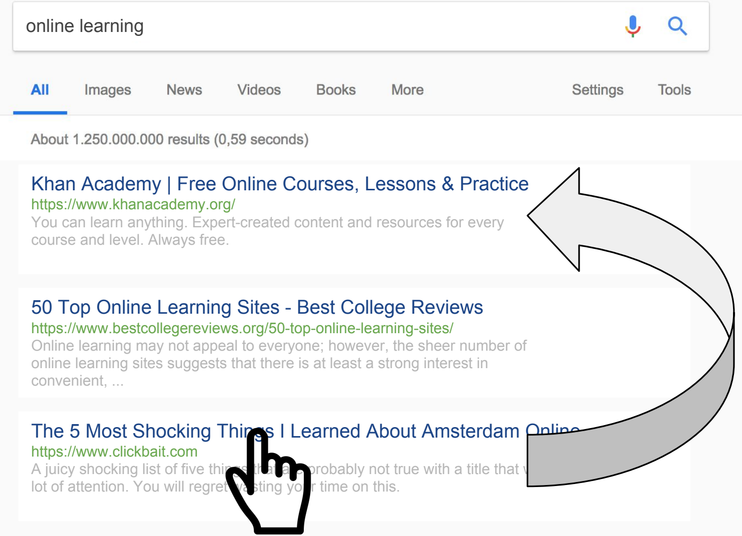
About 1.250.000.000 results (0,59 seconds)

Khan Academy | Free Online Courses, Lessons & Practice
https://www.khanacademy.org/
You can learn anything. Expert-created content and resources for every course and level. Always free.

50 Top Online Learning Sites - Best College Reviews
https://www.bestcollegereviews.org/50-top-online-learning-sites/
Online learning may not appeal to everyone; however, the sheer number of online learning sites suggests that there is at least a strong interest in convenient, ...

The 5 Most Shocking Things I Learned About Amsterdam Online
https://www.clickbait.com
A juicy shocking list of five things that are probably not true with a title that will get a lot of attention. You will regret wasting your time on this.

Online Learning to Rank: Example



Online Learning to Rank: Example

online learning

All Images News Videos Books More Settings Tools

About 1.250.000.000 results (0,59 seconds)

The 5 Most Shocking Things I Learned About Amsterdam Onli
https://www.clickbait.com
A juicy shocking list of five things that are probably not true with a title that w
lot of attention. You will regret wasting your time on this.

Khan Academy | Free Online Courses, Lessons & Practice
https://www.khanacademy.org/
You can learn anything. Expert-created content and resources for every
course and level. Always free.

50 Top Online Learning Sites - Best College Reviews
https://www.bestcollegereviews.org/50-top-online-learning-sites/
Online learning may not appeal to everyone; however, the sheer number of
learning sites suggests that there is at least a strong interest in convenient, ...

Online Learning to Rank: Example

online learning

All Images News Videos Books More Settings Tools

About 1.250.000.000 results (0,59 seconds)

The 5 Most Shocking Things I Learned About Amsterdam Online
https://www.clickbait.com
A juicy shocking list of five things that are probably not true with a title that will get a lot of attention. You will regret wasting your time on this.

Unbelievable Things We Learned About Online Shopping
https://www.moreclickbait.com
More things that are not researched and just to get your attention quickly.

Three Things Strangers Can Learn From Your Online Profile
https://www.evenmoreclickbait.com
Scaremongering for clicks, websites without any dignity exist.

Online Learning to Rank: Self-Confirming Loop

What happened here?

- A **click** on an **irrelevant** document happened for a **random reason**.

Online Learning to Rank: Self-Confirming Loop

What happened here?

- A **click** on an **irrelevant** document happened for a **random reason**.
- The algorithm **assumed** the document is **relevant**, and changes the ranking model to place it on top.

Online Learning to Rank: Self-Confirming Loop

What happened here?

- A **click** on an **irrelevant** document happened for a **random reason**.
- The algorithm **assumed** the document is **relevant**,
  and changes the ranking model to place it on top.
- Now documents **similar** to the clicked **irrelevant** document are **ranked on top**.

Online Learning to Rank: Self-Confirming Loop

What happened here?

- A **click** on an **irrelevant** document happened for a **random reason**.
- The algorithm **assumed** the document is **relevant**,
  and changes the ranking model to place it on top.
- Now documents **similar** to the clicked **irrelevant** document are **ranked on top**.
- Most likely the **next clicked document** will also be **irrelevant**.

Online Learning to Rank: Self-Confirming Loop

What happened here?

- A **click** on an **irrelevant** document happened for a **random reason**.
- The algorithm **assumed** the document is **relevant**,
  and changes the ranking model to place it on top.
- Now documents **similar** to the clicked **irrelevant** document are **ranked on top**.
- Most likely the **next clicked document** will also be **irrelevant**.

Online Learning to Rank: Self-Confirming Loop

What happened here?

- A **click** on an **irrelevant** document happened for a **random reason**.
- The algorithm **assumed** the document is **relevant**,
  and changes the ranking model to place it on top.
- Now documents **similar** to the clicked **irrelevant** document are **ranked on top**.
- Most likely the **next clicked document** will also be **irrelevant**.

We've entered a **self-confirming loop**:

- Due to **noise and bias**, a document was **incorrectly inferred relevant**.
- Due to **bias**, this inference is **most likely to occur again**.
- The algorithm's **confidence** in this **incorrect inference** continues to **increase**.

This behaviour is one of the biggest dangers in online learning.

Dueling Bandit Gradient Descent

Dueling Bandit Gradient Descent: Introduction

Introduced by Yue and Joachims (2009) as the **first online learning to rank** method.

Intuition:

- if **online evaluation** can tell us if a **ranker is better** than another, then we can use it to **find an improvement** of our system.

By **sampling model variants** and **comparing** them with **interleaving**, the *gradient* of a model w.r.t. user satisfaction can be **estimated**.

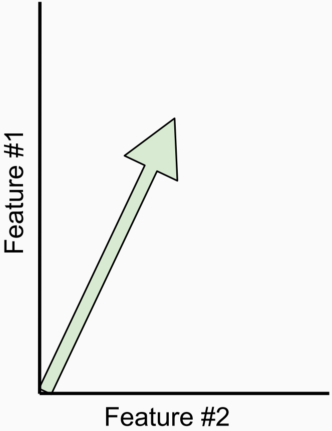
Dueling Bandit Gradient Descent: Method

Start with the **current** ranking model **parameters**: θ_b.

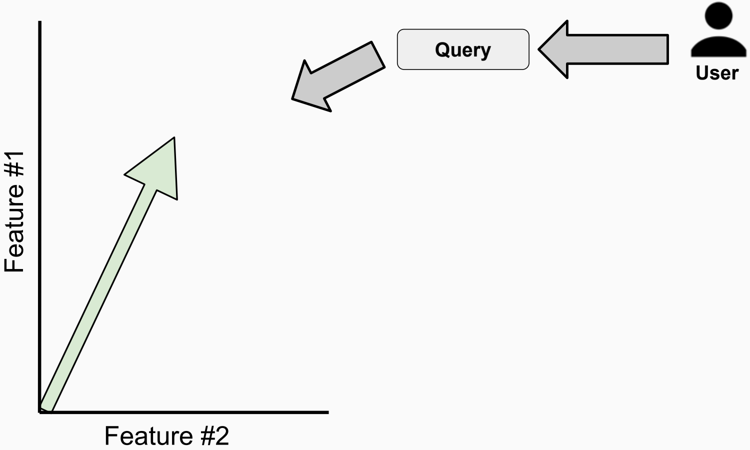
Then indefinitely:

1. Wait for a user query.
2. **Sample a random direction** from the unit sphere: u, (thus $|u| = 1$).
3. Compute the **candidate ranking model** $\theta_c = \theta_b + u$, (thus $|\theta_b - \theta_c| = 1$).
4. Get the **rankings** of θ_b and θ_c.
5. **Compare** θ_b and θ_c using interleaving.
6. If θ_c wins the **comparison**:
    - **Update** the current model: $\theta_b \leftarrow \theta_b + \eta(\theta_c - \theta_b)$

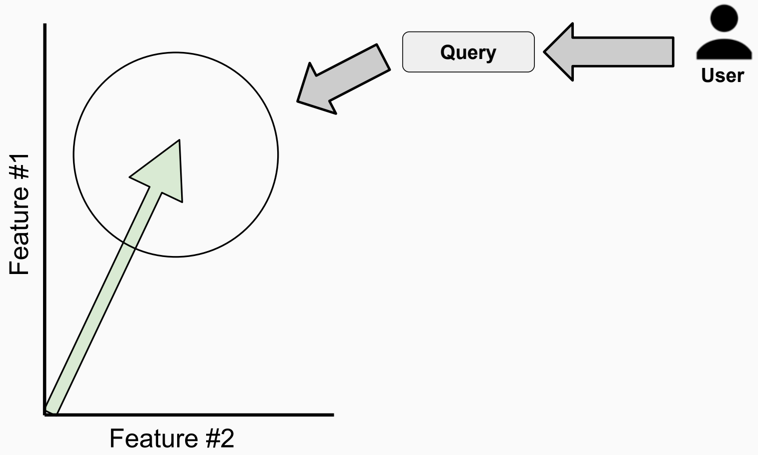
Dueling Bandit Gradient Descent: Visualization



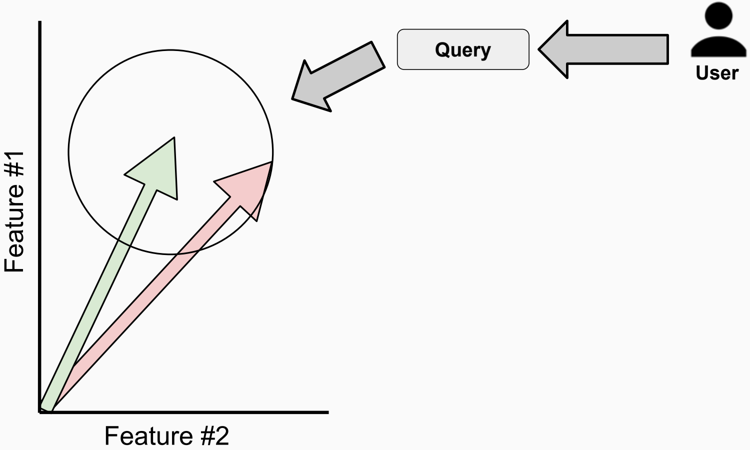
Dueling Bandit Gradient Descent: Visualization



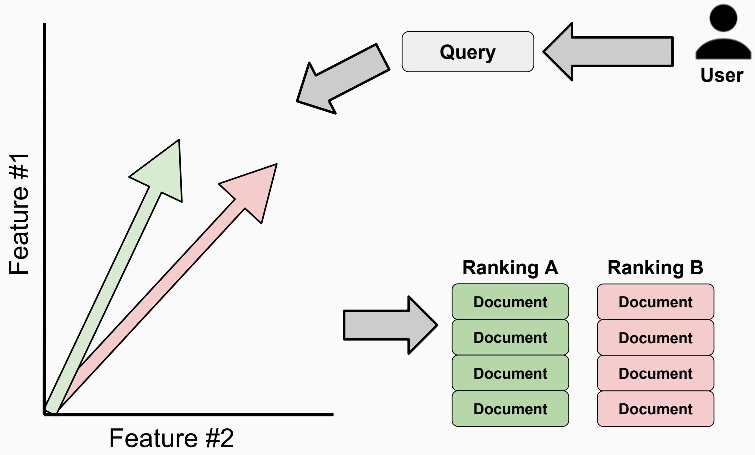
Dueling Bandit Gradient Descent: Visualization



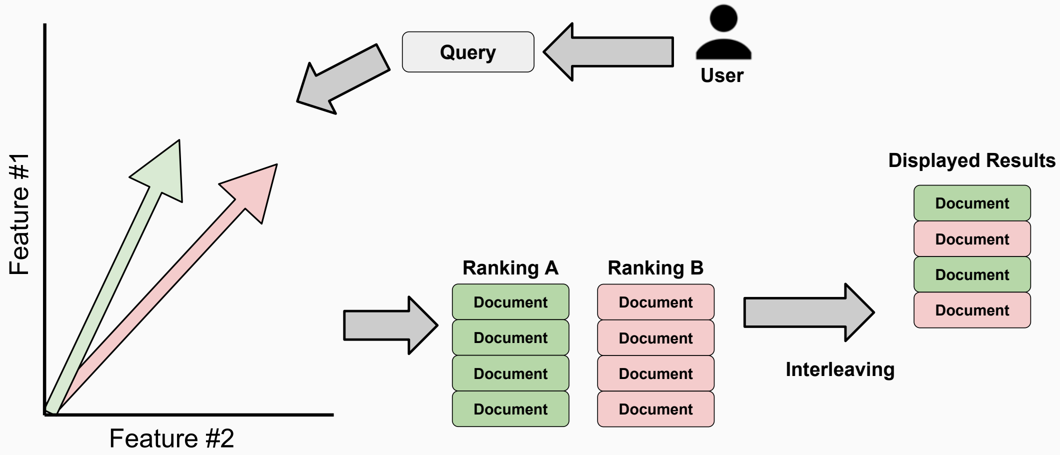
Dueling Bandit Gradient Descent: Visualization



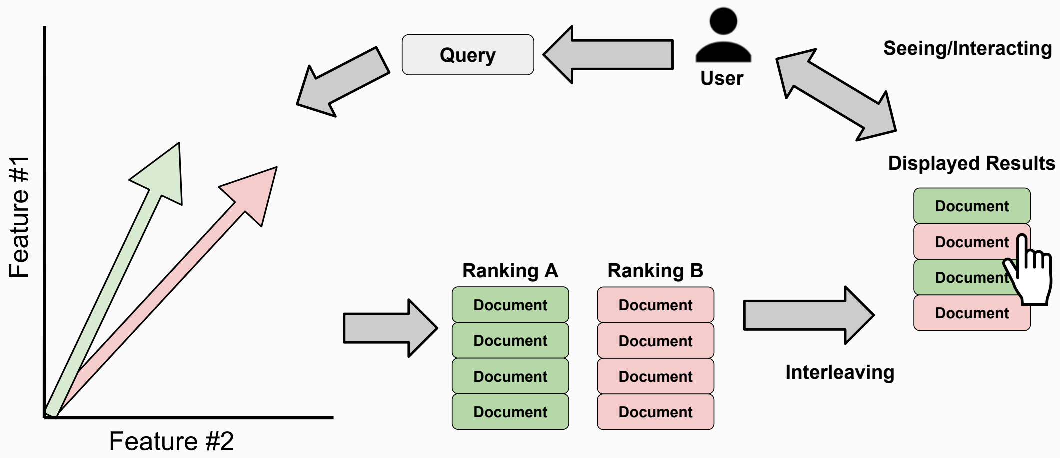
Dueling Bandit Gradient Descent: Visualization



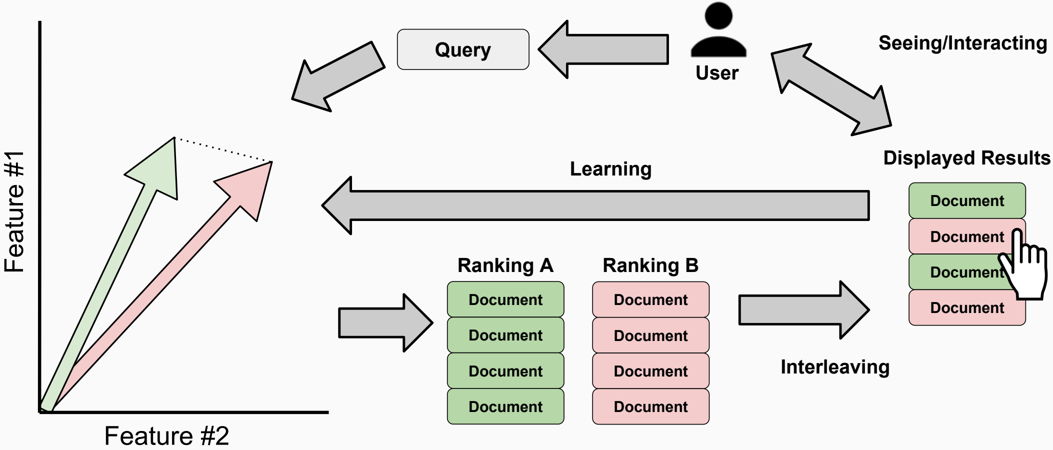
Dueling Bandit Gradient Descent: Visualization



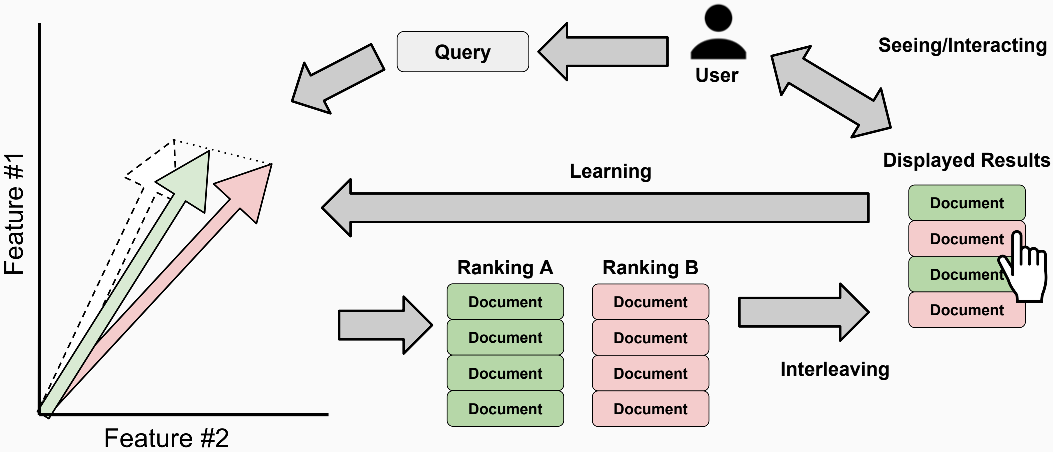
Dueling Bandit Gradient Descent: Visualization



Dueling Bandit Gradient Descent: Visualization



Dueling Bandit Gradient Descent: Visualization



Dueling Bandit Gradient Descent: Properties

Yue and Joachims (2009) prove that under the **assumptions**:

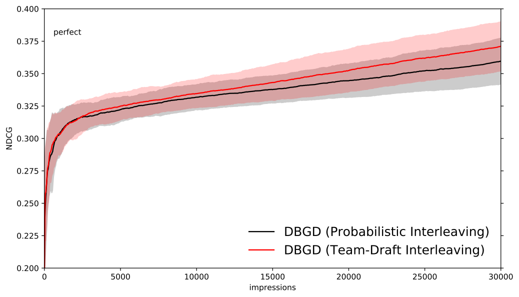
- There is a **single optimal** set of parameters: θ^*.
- The **utility space** w.r.t. θ is **smooth**,
  i.e. small changes in θ lead to small changes in user experience.

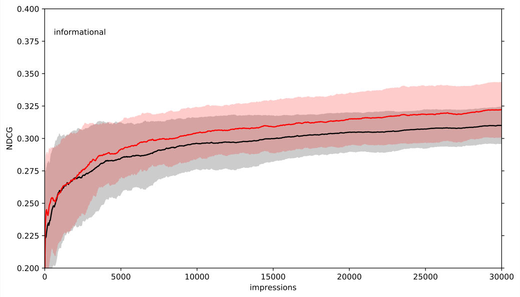
Then Dueling Bandit Gradient Descent is **guaranteed** to have a **sublinear regret**:

- The algorithm will **eventually** approximate the ideal model.
- The duration of time is effected by the number of parameters of the model, the smoothness of the space, the unit chosen, etc.

Dueling Bandit Gradient Descent: Visualization

Simulations based on offline datasets: **user behaviour** is based on the **annotations**.
As a result, we can **measure** how close the **model** is getting to their **satisfaction**.





Simulated results on the MSLR-WEB10k dataset,
a perfect user (left) and an informational user (right).

The Contextual Bandit Problem, and Online Learning to Rank

Online Learning to Rank, and the Contextual Bandit Problem

Online Learning to Rank is related to **Reinforcement Learning** (Sutton and Barto, 1998) and the **Contextual Bandit Problem** (Langford and Zhang, 2008).

Roughly speaking in a contextual bandit problem:

1. The agent receives **contextual information**.
2. The agent **chooses an action** out of a set of available actions.
3. The action is performed.
4. A **reward** for the performed action is **observed**.

Differences between CBP and OLTR

In a contextual bandit problem:

1. The agent receives **contextual information**.
2. The agent **chooses an action** out of a set of available actions.
3. The action is **performed**.
4. A **reward** for the performed action is **observed**.

In online learning to rank:

1. The system receives a **query** from the user.
2. The system **constructs a ranking** out of the set of **available documents**.
3. The ranking is **displayed** to the user.
4. **User interactions** with the ranking are observed.

Differences between CBP and OLTR

Why don't we use **contextual bandit algorithms** (CBP) for the online learning to rank (OLTR) problem?

Differences between CBP and OLTR

Why don't we use **contextual bandit algorithms** (CBP) for the online learning to rank (OLTR) problem?

- In OLTR **rewards** are **not observed directly**.

Differences between CBP and OLTR

Why don't we use **contextual bandit algorithms** (CBP) for the online learning to rank (OLTR) problem?

- In OLTR **rewards** are **not observed directly**.
  - A **reward** can be **inferred from interactions**,
    e.g. the number of clicks is the reward.

Differences between CBP and OLTR

Why don't we use **contextual bandit algorithms** (CBP) for the online learning to rank (OLTR) problem?

- In OLTR **rewards** are **not observed directly**.
  - A **reward** can be **inferred from interactions**, e.g. the number of clicks is the reward.
  - This is **very unsafe**, you risk optimizing the **wrong objective**.

Differences between CBP and OLTR

Why don't we use **contextual bandit algorithms** (CBP) for the online learning to rank (OLTR) problem?

- In OLTR **rewards** are **not observed directly**.
  - A **reward** can be **inferred from interactions**, e.g. the number of clicks is the reward.
  - This is **very unsafe**, you risk optimizing the **wrong objective**.
- In OLTR the **action space is immense**: all possible rankings, CBP algorithms don't work well with large action spaces.

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,
  - we expect this action to lead to the **most immediate reward**!

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,
    - we expect this action to lead to the **most immediate reward**!
    - but we **risk** that there is a **better action** we **don't know about yet**.

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,
  - we expect this action to lead to the **most immediate reward**!
  - but we **risk** that there is a **better action** we **don't know about yet**.
- **Exploration:** **Trying** an action that we **don't think is the best**,

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,
  - we expect this action to lead to the **most immediate reward**!
  - but we **risk** that there is a **better action** we **don't know about yet**.
- **Exploration:** **Trying** an action that we **don't think is the best**,
  - we expect this leads to **immediate suboptimal reward**!

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,
  - we expect this action to lead to the **most immediate reward**!
  - but we **risk** that there is a **better action** we **don't know about yet**.
- **Exploration:** **Trying** an action that we **don't think is the best**,
  - we expect this leads to **immediate suboptimal reward**!
  - but we may find an action is **better than we thought**.

Lessons from CBP for OLTR

The **exploitation/exploration tradeoff** is well studied in CBP and RL,
Hofmann et al. (2013b) showed that (unsurprisingly) it is also **important in OLTR**.

The two sides of the tradeoff:

- **Exploitation:** Performing the action that we **currently think is best**,
  - we expect this action to lead to the **most immediate reward**!
  - but we **risk** that there is a **better action** we **don't know about yet**.
- **Exploration:** **Trying** an action that we **don't think is the best**,
  - we expect this leads to **immediate suboptimal reward**!
  - but we may find an action is **better than we thought**.

A **mix of exploitation and exploration** leads to the **best long-term** performance.

Reusing Historical Interactions

Reusing Historical Interactions

Hofmann et al. (2013a) introduced the idea of **guiding exploration** by **reusing previous interactions**.

Dueling Bandit Gradient Descent tries out a different potential gradient direction at each step.

Intuition: if **previous interactions** showed that a **direction is unfruitful** then we should **avoid it in the future**.

Candidate Pre-Selection

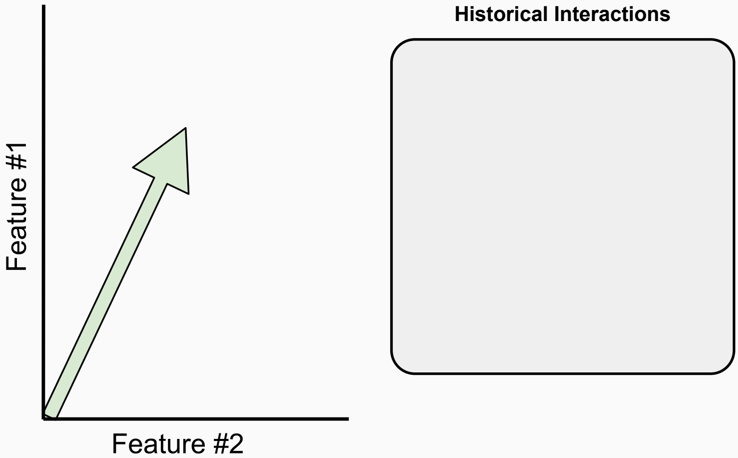
Remember the last n interactions in $\mathbf{h}$.

1. Sample m pre-candidates: $\mathbf{e} \leftarrow \{\theta^c_1, \ldots, \theta^c_m\}$
2. Repeat until $|\mathbf{e}| = 1$:
   1. Sample two candidates from $\mathbf{e}$: θ^c_l, θ^c_r
   2. Sample a **historical user interaction event** from $\mathbf{h}$: h'
   3. Compare candidates using **estimating probabilistic interleaving results**:
      if $o(\theta^c_l, \theta^c_r, h') > 0$
      - remove θ^c_r from $\mathbf{e}$.

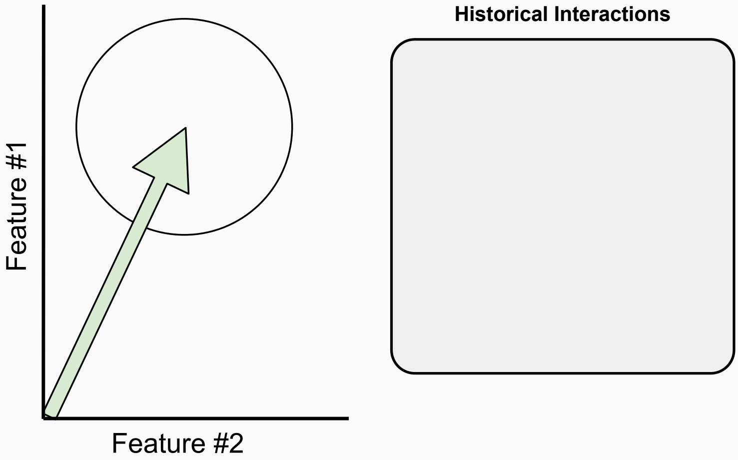
      else if $o(\theta^c_l, \theta^c_r, h') < 0$
      - remove θ^c_l from $\mathbf{e}$.

      else
      - Sample θ^c_x from $\{\theta^c_l, \theta^c_r\}$.
      - Remove θ^c_x from $\mathbf{e}$.
3. The **last remaining candidate** from $\mathbf{e}$ is **pre-selected** for DBGD.

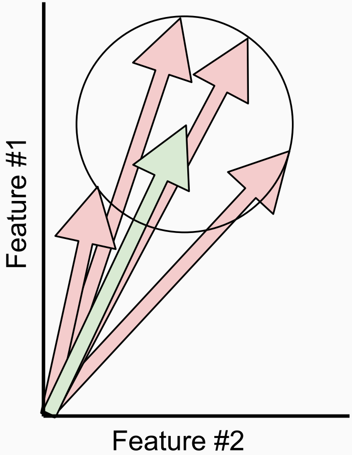
Candidate Pre-Selection: Visualization

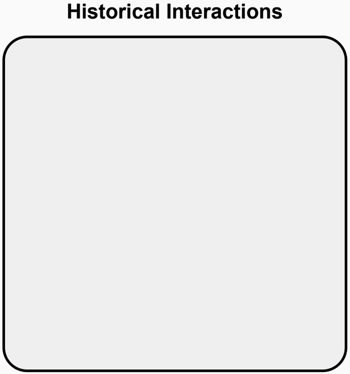


Candidate Pre-Selection: Visualization

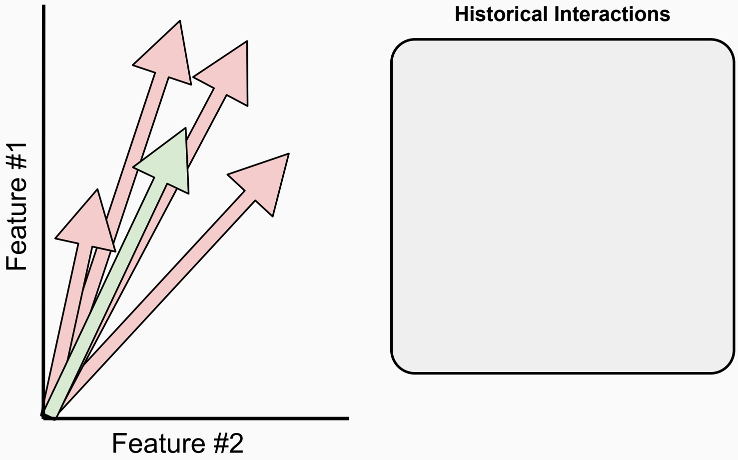


Candidate Pre-Selection: Visualization

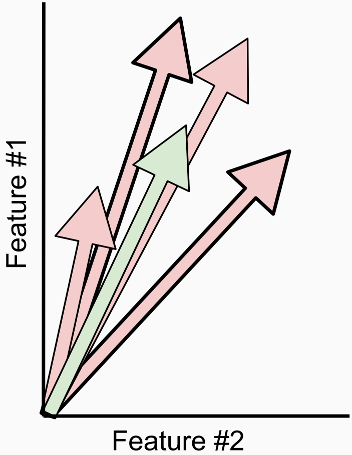


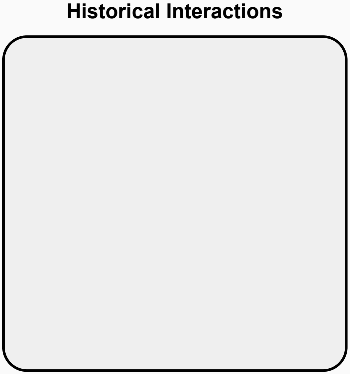


Candidate Pre-Selection: Visualization

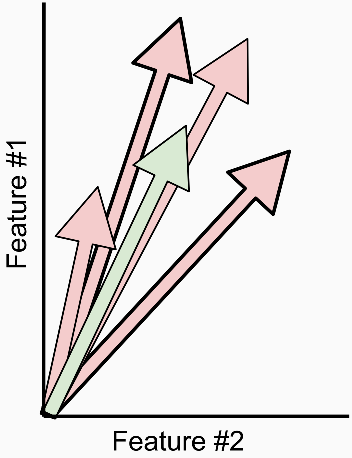


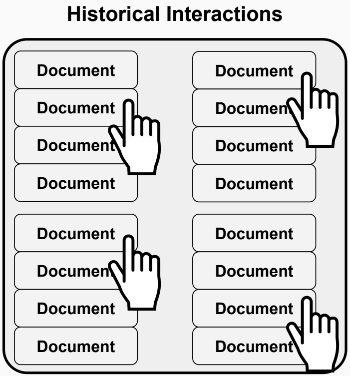
Candidate Pre-Selection: Visualization



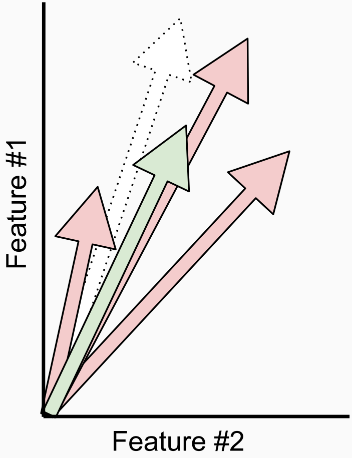


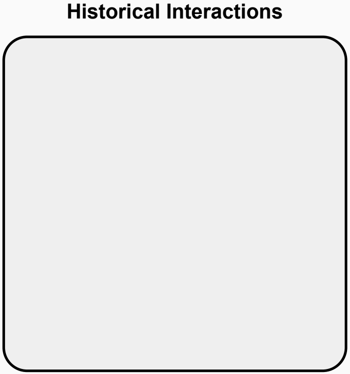
Candidate Pre-Selection: Visualization



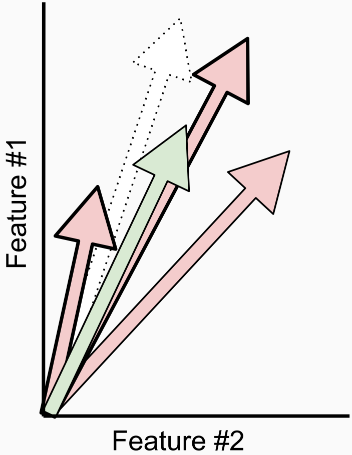


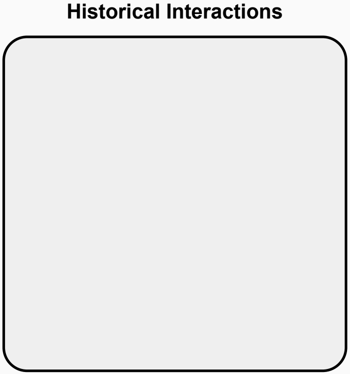
Candidate Pre-Selection: Visualization



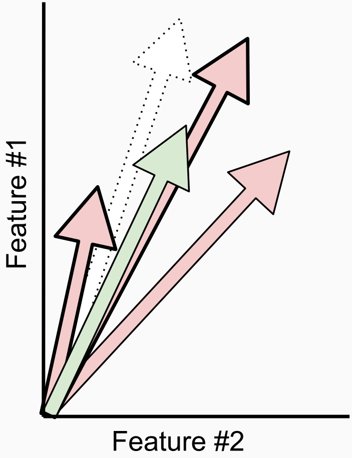


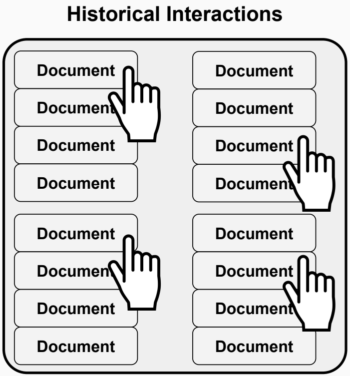
Candidate Pre-Selection: Visualization



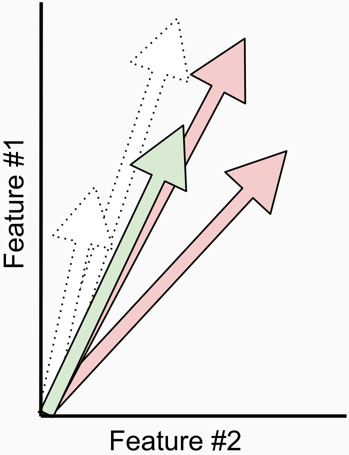


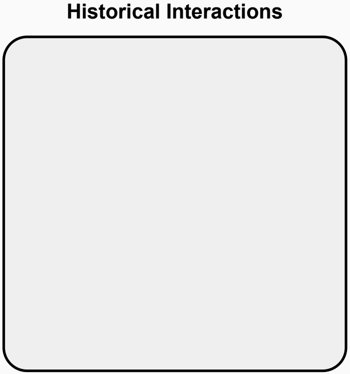
Candidate Pre-Selection: Visualization



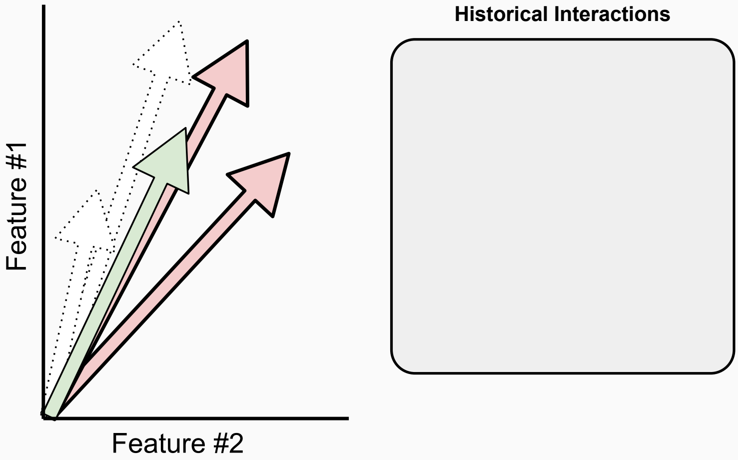


Candidate Pre-Selection: Visualization

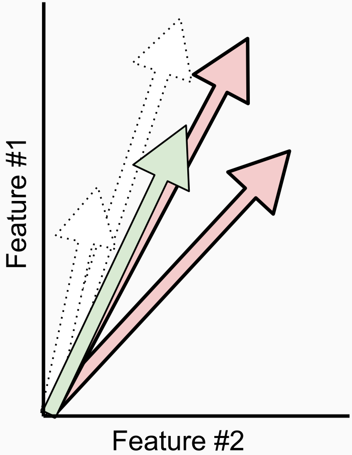


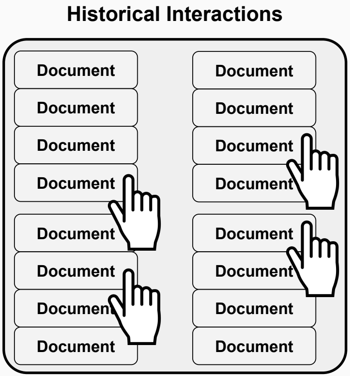


Candidate Pre-Selection: Visualization

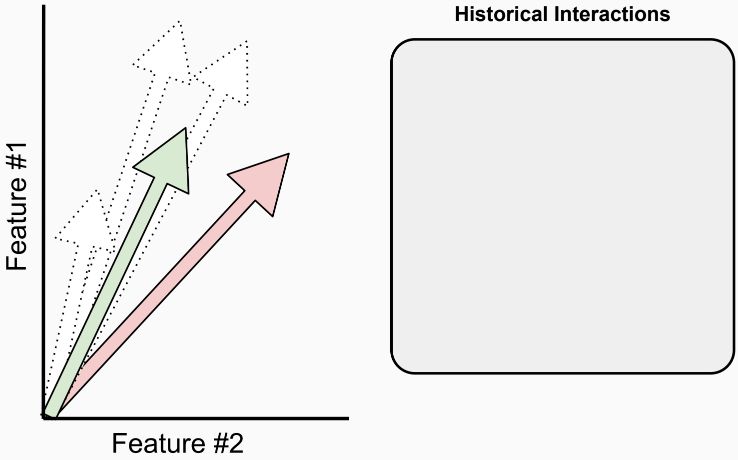


Candidate Pre-Selection: Visualization

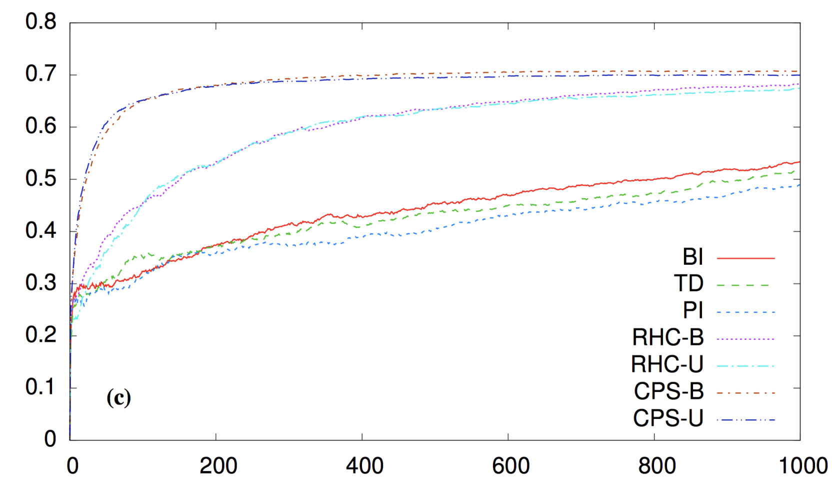




Candidate Pre-Selection: Visualization



Reusing Historical Interactions: Performance



Simulated results on the NP2003 dataset, graph from (Hofmann et al., 2013a).

Reusing Historical Interactions: Caveat

Candidate pre-selection achieves a **faster learning rate** by guiding exploration: **discarding unpromising candidate rankers**.

What is a **potential danger** with this approach?

Reusing Historical Interactions: Caveat

Candidate pre-selection achieves a **faster learning rate** by guiding exploration: **discarding unpromising candidate rankers**.

What is a **potential danger** with this approach?

Remember:

- **Exploration** is used to **discover** actions that perform **better than expected**, this ultimately leads to the **best long-term performance**.

Reusing Historical Interactions: Caveat

Candidate pre-selection achieves a **faster learning rate** by guiding exploration: **discarding unpromising candidate rankers**.

What is a **potential danger** with this approach?

Remember:

- **Exploration** is used to **discover** actions that perform **better than expected**, this ultimately leads to the **best long-term performance**.
- **Candidate Pre-Selection** uses **expectations from history** to **exclude candidate rankers** from being explored.

Reusing Historical Interactions: Caveat

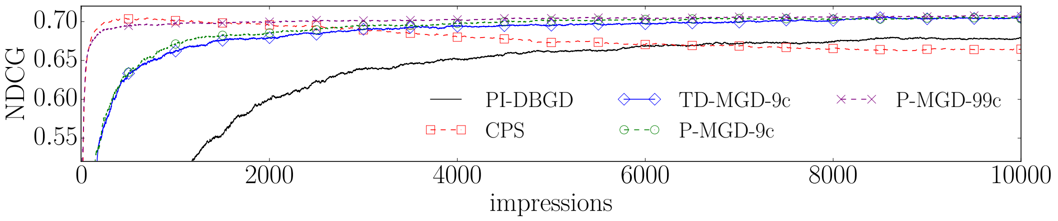
Candidate pre-selection achieves a **faster learning rate** by guiding exploration: **discarding unpromising candidate rankers**.

What is a **potential danger** with this approach?

Remember:

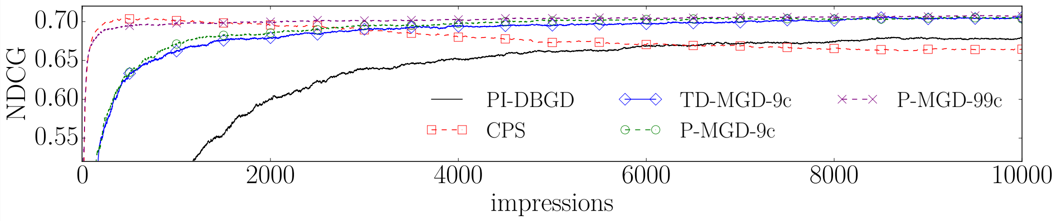
- **Exploration** is used to **discover** actions that perform **better than expected**, this ultimately leads to the **best long-term performance**.
- **Candidate Pre-Selection** uses **expectations from history** to **exclude candidate rankers** from being explored.
- This is dangerously close to a **self-confirming loop**.

Reusing Historical Interactions: Long Term Performance



Simulated results on the NP2003 dataset, graph from (Oosterhuis et al., 2016).

Reusing Historical Interactions: Long Term Performance



Simulated results on the NP2003 dataset, graph from (Oosterhuis et al., 2016).

Remember, in the online setting the **performance cannot be measured**, thus **early-stopping is impossible**.

Reusing Historical Interactions: Other Work

Besides Hofmann et al. (2013a) **other work** has also tried **reusing historical interactions** for online learning to rank: (Zhao and King, 2016; Wang et al., 2018a).

The problem with these works is that:

- they **don't consider the long-term convergence**.
- they were **not evaluated** on the **largest available industry datasets**.

As a result, it is **still unclear** whether we can **reliably reuse historical interactions** during online learning.

Multileave Gradient Descent

Multileave Gradient Descent

The introduction of **multileaving** in online evaluation allowed for **multiple rankers being compared simultaneously** from a single interaction.

A **natural extension** of Dueling Bandit Gradient Descent is to combine it with multileaving, resulting in **Multileave Gradient Descent** (Schuth et al., 2016).

Multileaving allows comparisons with **multiple candidate rankers**, **increasing** the **chance of finding an improvement**.

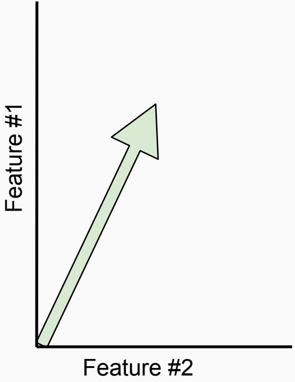
Multileave Gradient Descent: Method

Start with the current ranking model parameters: θ_b.

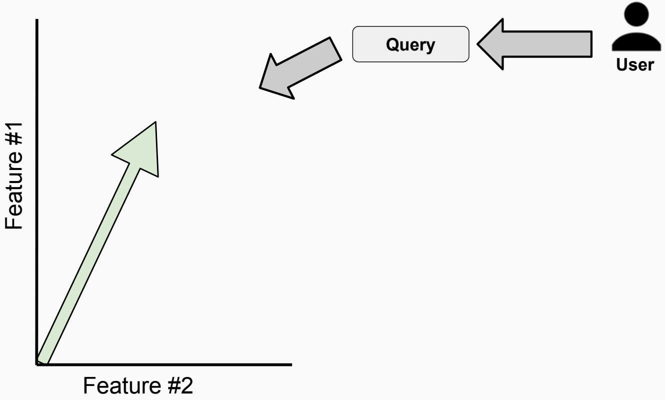
Then indefinitely:

1. Start with an empty candidate set: $\zeta \leftarrow \{\}$.
2. Then for n candidates:
   1. **Sample a random direction** from the unit sphere: u, (thus $|u| = 1$).
   2. Compute the **candidate ranking model** $\theta_c = \theta_b + u$, (thus $|\theta_b - \theta_c| = 1$).
   3. Add **candidate** θ_c to set: $\zeta \leftarrow \zeta \cup \{\theta_c\}$.
3. Compare θ_b and ζ using **multileaving to get the preferences**: $\mathcal{P}$.
4. Determine the **winning set**: $\omega \leftarrow \{\theta_c | \theta_c \in \mathcal{P} \wedge \theta_c >_{\mathcal{P}} \theta_b\}$
5. **Update** current model $\theta_b \leftarrow \theta_b + \frac{1}{|\omega|} \sum_{\theta_c \in \omega} (\theta_c - \theta_b)$

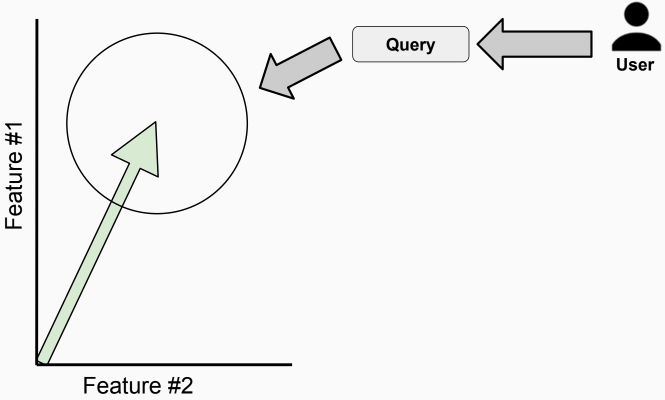
Multileave Gradient Descent: Visualization



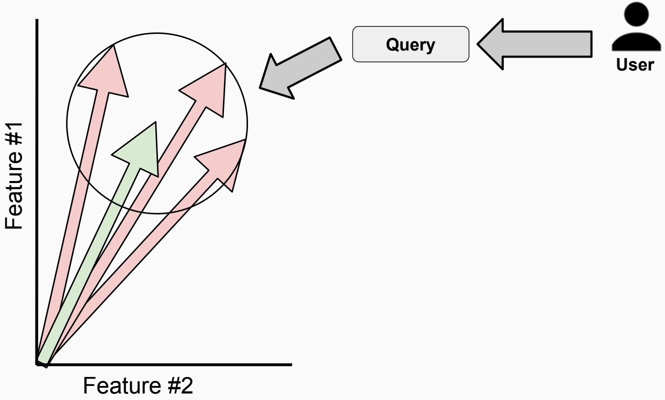
Multileave Gradient Descent: Visualization



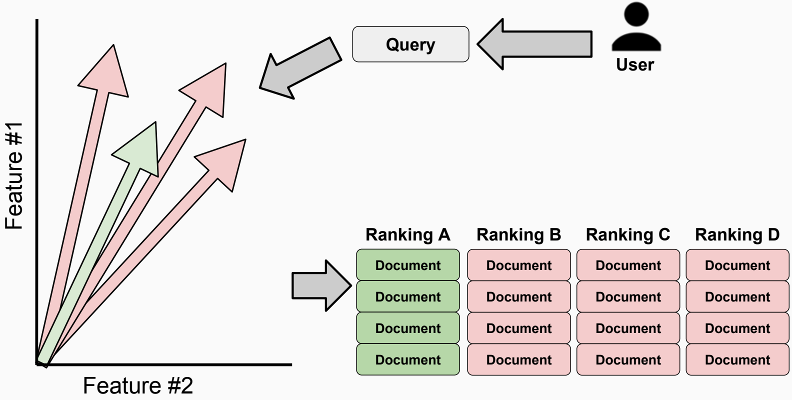
Multileave Gradient Descent: Visualization



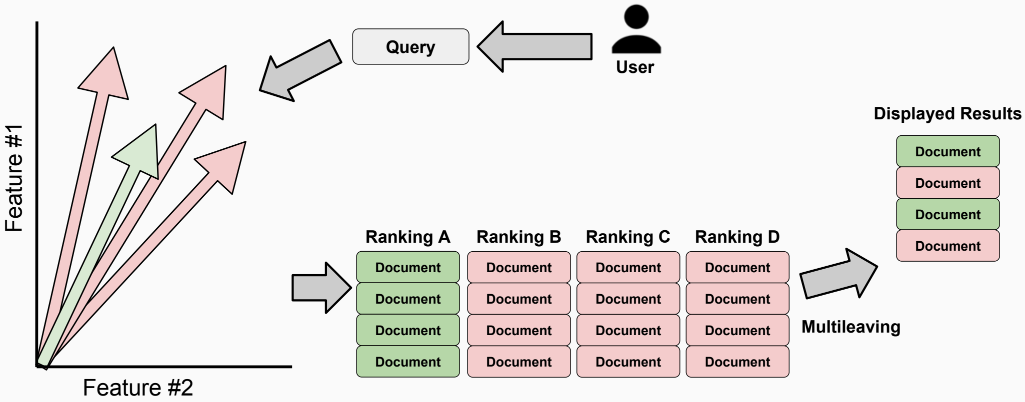
Multileave Gradient Descent: Visualization



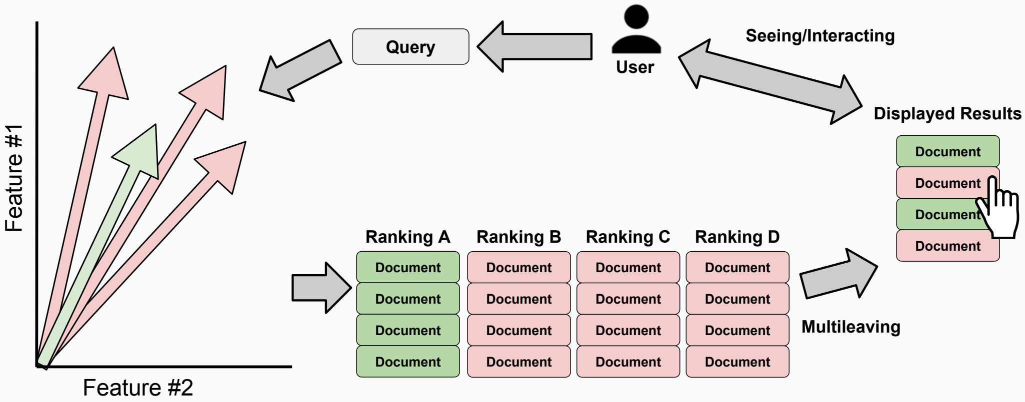
Multileave Gradient Descent: Visualization



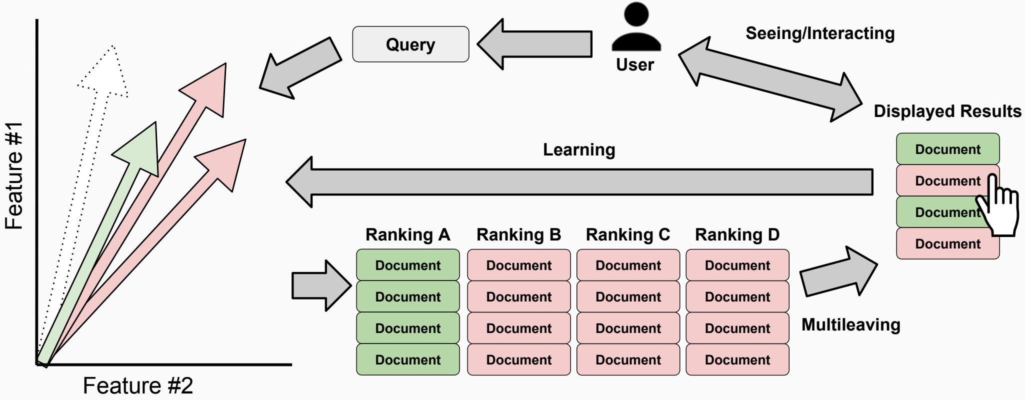
Multileave Gradient Descent: Visualization



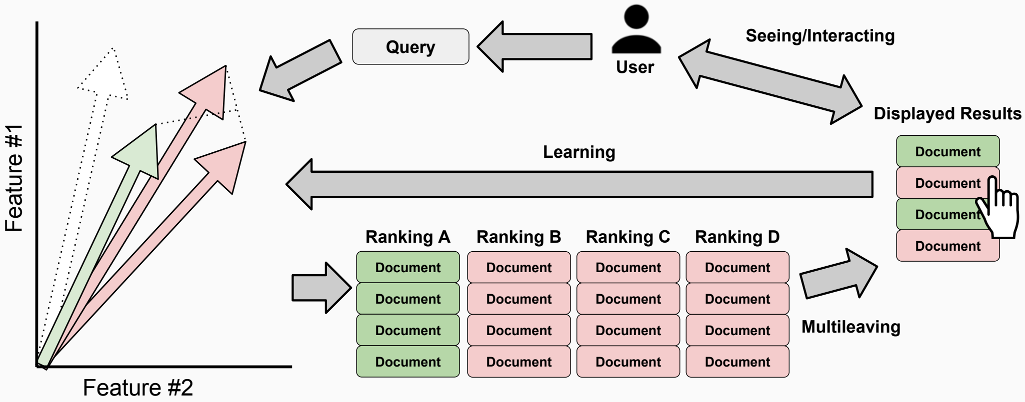
Multileave Gradient Descent: Visualization



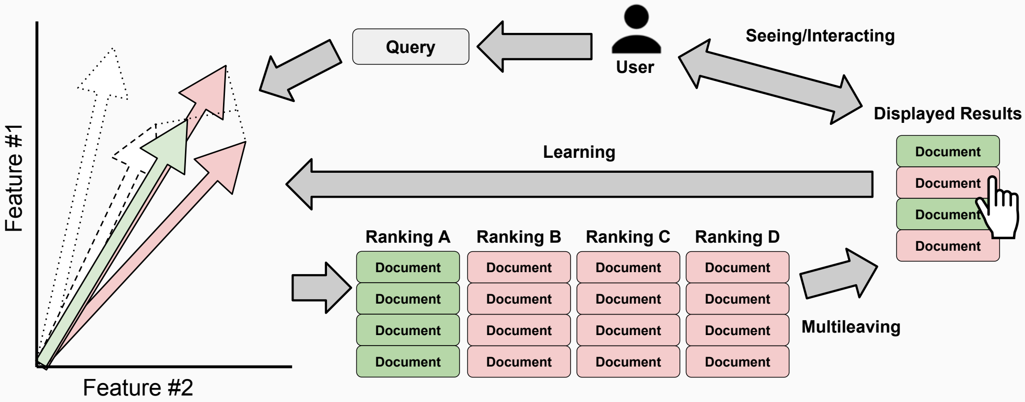
Multileave Gradient Descent: Visualization



Multileave Gradient Descent: Visualization

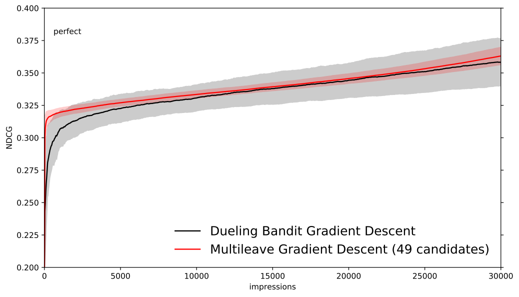


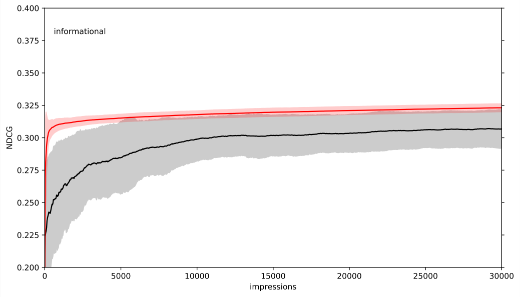
Multileave Gradient Descent: Visualization



Multileave Gradient Descent: Results

Results on the MSRL10k dataset under simulated users:





Multileave Gradient Descent: Conclusion

Properties of Multileave Gradient Descent:

- **Vastly speeds up the learning rate** of Dueling Bandit Gradient Descent.
  - Much better user experience.
- Instead of **limiting (guiding) exploration**, it is done more **efficiently**.

Multileave Gradient Descent: Conclusion

Properties of Multileave Gradient Descent:

- **Vastly speeds up the learning rate** of Dueling Bandit Gradient Descent.
  - Much better user experience.
- Instead of **limiting (guiding) exploration**, it is done more **efficiently**.
- **Huge computational costs**, large number of rankers have to be applied.

Speed-Quality Tradeoff

Speed-Quality Tradeoff

So far we've **only** discussed **different algorithms** for online learning to rank.

We've **not talked** about different ranking **models**.

The **first eights years** of work in the field have only considered **linear models**, this is not a coincidence.

Ranking Models in Online Learning to Rank Research

Recognized by Oosterhuis and de Rijke (2017a) is the **Speed-Quality tradeoff**, that is unique to online learning.

We know from machine learning:

Ranking Models in Online Learning to Rank Research

Recognized by Oosterhuis and de Rijke (2017a) is the **Speed-Quality tradeoff**, that is unique to online learning.

We know from machine learning:

- **Complex** models, e.g. deep learning, are **more expressive**, i.e. fit more patterns, however, they also **require more data** to train.

Ranking Models in Online Learning to Rank Research

Recognized by Oosterhuis and de Rijke (2017a) is the **Speed-Quality tradeoff**, that is unique to online learning.

We know from machine learning:

- **Complex** models, e.g. deep learning, are **more expressive**, i.e. fit more patterns, however, they also **require more data** to train.
- **Simpler** models, e.g. linear models, are **less expressive**, i.e. underfit some patterns, however, they **require much less data** to train.

Speed-Quality Tradeoff

For online learning to rank:

- **More data means more user interactions.**

Thus the **choice of model** balances the **short-term** (speed) and **long-term** (quality) performance:

Speed-Quality Tradeoff

For online learning to rank:

- **More data means more user interactions.**

Thus the **choice of model** balances the **short-term** (speed) and **long-term** (quality) performance:

- **Complex** models have better **convergence** (long-term performance), but **need more user interactions** to reach decent quality (short-term performance).

Speed-Quality Tradeoff

For online learning to rank:

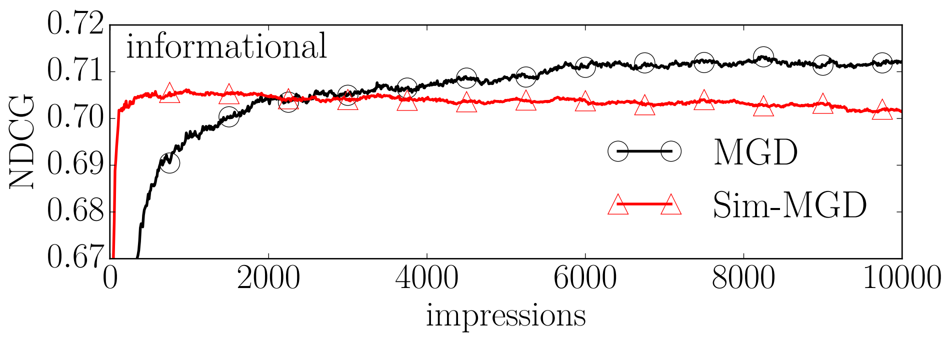
- **More data means more user interactions.**

Thus the **choice of model** balances the **short-term** (speed) and **long-term** (quality) performance:

- **Complex** models have better **convergence** (long-term performance), but **need more user interactions** to reach decent quality (short-term performance).
- **Simpler** models can learn **very fast** (short-term performance), but will **converge on suboptimal quality** (long-term performance).

Speed-Quality Tradeoff

Results for a **linear model** (MGD) and a simpler model with **reduced dimensionality** (Sim-MGD):



Source: (Oosterhuis and de Rijke, 2017a)

Cascading Multiple Models

Introducing a **new model** can thus **never both**:

- **Improve final convergence.**
- **Improve user experience during optimization.**

Explaining the lack of work into models for online learning to rank.

Cascading Multiple Models

Introducing a **new model** can thus **never both**:

- **Improve final convergence.**
- **Improve user experience during optimization.**

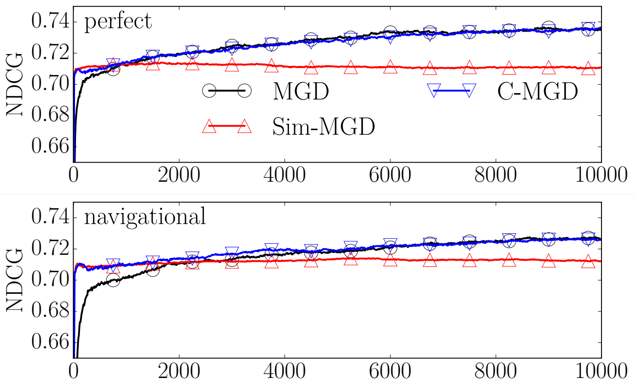
Explaining the lack of work into models for online learning to rank.

As a solution Oosterhuis and de Rijke (2017a) **optimize a cascade of models**:

- **Optimize** a **simple** model until **convergence**.
- **Continue** with **complexer** model.

Cascading Multiple Models: Results

Results for a **linear model** (MGD) and a **simpler model** with **reduced dimensionality** (Sim-MGD) and a **cascade of the two models** (C-MGD):



Source: (Oosterhuis and de Rijke, 2017a)

Problems with Dueling Bandit Gradient Descent

Problems with Dueling Bandit Gradient Descent

A **problem** with Dueling Bandit Gradient Descent and **all its extensions**:

- Their **performance at convergence** is **much worse** than offline approaches, even **under ideal user interactions**.

How is this possible, if it's **guaranteed to find the optimal model in sublinear time**?

Problems with the Dueling Bandit Gradient Descent Bounds

Remember the **regret** of Dueling Bandit Gradient Descent made **two assumptions**:

- There is a **single optimal model**: θ^*.
- The **utility space is smooth** w.r.t. to the model weights θ.

Problems with the Dueling Bandit Gradient Descent Bounds

Remember the **regret** of Dueling Bandit Gradient Descent made **two assumptions**:

- There is a **single optimal model**: θ^*.
- The **utility space is smooth** w.r.t. to the model weights θ.

These **assumptions don't hold** for models that can be **scaled**:

$$\forall \theta, \quad \forall \alpha, \quad \exists \theta', \quad \alpha > 0 \rightarrow f(\mathbf{d}, \theta) = \alpha f(\mathbf{d}, \theta'). \tag{13}$$

Problems with the Dueling Bandit Gradient Descent Bounds

Remember the **regret** of Dueling Bandit Gradient Descent made **two assumptions**:

- There is a **single optimal model**: θ^*.
- The **utility space is smooth** w.r.t. to the model weights θ.

These **assumptions don't hold** for models that can be **scaled**:

$$\forall\theta, \quad \forall\alpha, \quad \exists\theta', \quad \alpha > 0 \rightarrow f(\mathbf{d}, \theta) = \alpha f(\mathbf{d}, \theta'). \tag{13}$$

For instance, multiplying the weights of a linear model with any positive scalar results in the same rankings.

Problems with the Dueling Bandit Gradient Descent Bounds

Remember the **regret** of Dueling Bandit Gradient Descent made **two assumptions**:

- There is a **single optimal model**: θ^*.
- The **utility space is smooth** w.r.t. to the model weights θ.

These **assumptions don't hold** for models that can be **scaled**:

$$\forall\theta, \quad \forall\alpha, \quad \exists\theta', \quad \alpha > 0 \rightarrow f(\mathbf{d}, \theta) = \alpha f(\mathbf{d}, \theta'). \tag{13}$$

For instance, multiplying the weights of a linear model with any positive scalar results in the same rankings.

This is true for **linear models, deep models, regression trees, etc.**
for all these models the **assumptions do not hold**, therefore **neither does the proof**.

Problems with the Dueling Bandit Gradient Descent

Upon closer inspection **Dueling Bandit Gradient Descent** looks more like an **evolutionary algorithm** than **stochastic gradient descent**.

Pairwise Differentiable Gradient Descent

Pairwise Differentiable Gradient Descent

Dueling Bandit Gradient Descent and all its extensions are **based on online evaluation methods**.
(This is all existing work in Online Learning to Rank up to 2018.)

In the upcoming CIKM'18 conference we will present a **novel online learning to rank algorithm** (Oosterhuis and de Rijke, 2018b).

Intuition: A **pairwise** method can be made **unbiased**, while being **differentiable**, without relying on online evaluation method or the sampling of models.

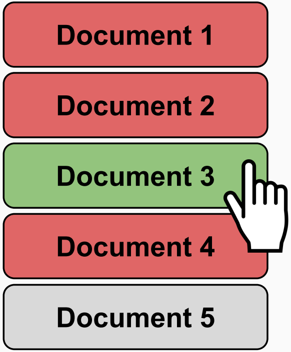
Plackett Luce Model

Pairwise Differentiable Gradient Descent optimizes a **Plackett Luce** ranking model, this models a **probabilistic distribution over documents**:

$$P(d|D, \theta) = \frac{\exp^{f(\mathbf{d}, \theta)}}{\sum_{d' \in D} \exp^{f(\mathbf{d}', \theta)}} \tag{14}$$

Pairwise Preference Inference

Similar to existing pairwise methods (Oosterhuis and de Rijke, 2017b; Joachims, 2002), Pairwise Differentiable Gradient Descent infers **document preferences from user clicks**:



Biased Pairwise Update

The **probability** that a document pair d_i, d_j is sampled **according to the inferred preference** $d_i >_{\mathbf{c}} d_j$ is **increased**:

$$P(d_i \succ d_j | D, \theta) = \frac{P(d_i|D,\theta)}{P(d_i|D,\theta) + P(d_j|D,\theta)} = \frac{\exp^{f(\mathbf{d_i},\theta)}}{\exp^{f(\mathbf{d_i},\theta)} + \exp^{f(\mathbf{d_j},\theta)}} \tag{15}$$

Biased Pairwise Update

The **probability** that a document pair d_i, d_j is sampled **according to the inferred preference** $d_i >_{\mathbf{c}} d_j$ is **increased**:

$$P(d_i \succ d_j | D, \theta) = \frac{P(d_i|D,\theta)}{P(d_i|D,\theta) + P(d_j|D,\theta)} = \frac{\exp^{f(\mathbf{d_i},\theta)}}{\exp^{f(\mathbf{d_i},\theta)} + \exp^{f(\mathbf{d_j},\theta)}} \tag{15}$$

With $>_{\mathbf{c}}$ indicating inferred document preference, this gives the **(estimated) gradient**:

$$\sum_{d_i >_{\mathbf{c}} d_j} \nabla P(d_i \succ d_j | D, \theta) = \sum_{d_i >_{\mathbf{c}} d_j} \frac{\exp^{f(\mathbf{d_i},\theta)} \exp^{f(\mathbf{d_j},\theta)}}{(\exp^{f(\mathbf{d_i},\theta)} + \exp^{f(\mathbf{d_j},\theta)})^2} (f'(\mathbf{d_i},\theta) - f'(\mathbf{d_j},\theta)) \tag{16}$$

Biased Pairwise Update

The **probability** that a document pair d_i, d_j is sampled **according to the inferred preference** $d_i >_{\mathbf{c}} d_j$ is **increased**:

$$P(d_i \succ d_j | D, \theta) = \frac{P(d_i|D,\theta)}{P(d_i|D,\theta) + P(d_j|D,\theta)} = \frac{\exp^{f(\mathbf{d_i},\theta)}}{\exp^{f(\mathbf{d_i},\theta)} + \exp^{f(\mathbf{d_j},\theta)}} \tag{15}$$

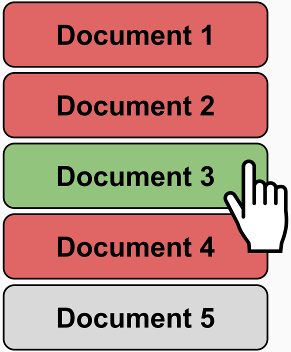
With $>_{\mathbf{c}}$ indicating inferred document preference, this gives the **(estimated) gradient**:

$$\sum_{d_i >_{\mathbf{c}} d_j} \nabla P(d_i \succ d_j | D, \theta) = \sum_{d_i >_{\mathbf{c}} d_j} \frac{\exp^{f(\mathbf{d_i},\theta)} \exp^{f(\mathbf{d_j},\theta)}}{(\exp^{f(\mathbf{d_i},\theta)} + \exp^{f(\mathbf{d_j},\theta)})^2} (f'(\mathbf{d_i},\theta) - f'(\mathbf{d_j},\theta)) \tag{16}$$

What may be a problem with this approach?

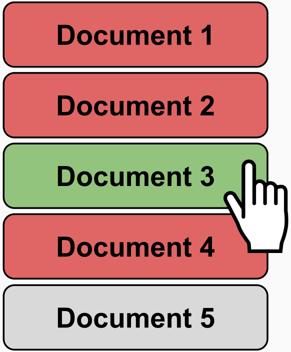
Bias in Pairwise Inference

The pairwise preference approach is **biased**, some preferences are **more likely to be found** that others.



Bias in Pairwise Inference

The pairwise preference approach is **biased**, some preferences are **more likely to be found** that others.

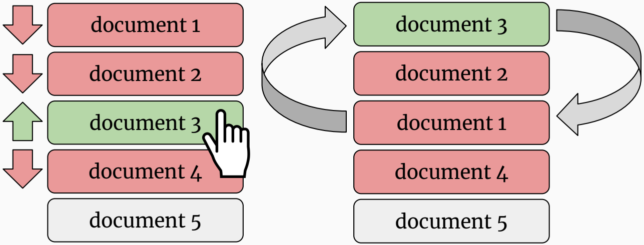


How do we solve this problem?

Reverse Pair Rankings

Assumption:

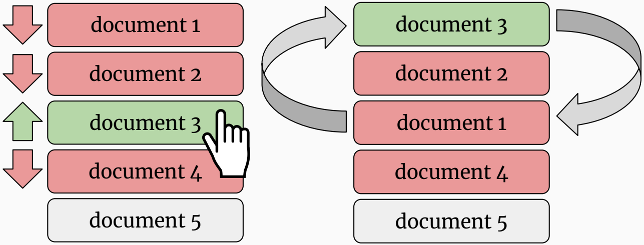
- if d_i and d_j are **equally relevant** then **finding $d_i >_\mathbf{c} d_j$ is equally likely as finding $d_j >_\mathbf{c} d_i$**, after the **documents are swapped** in a ranking.



Reverse Pair Rankings

Assumption:

- if d_i and d_j are **equally relevant** then **finding $d_i >_c d_j$ is equally likely as finding $d_j >_c d_i$**, after the **documents are swapped** in a ranking.



We call the ranking with the swapped pair the **reversed pair ranking**: $R^*(R, d_i, d_j)$.

Unbiasing the Pairwise Update

The **ratio** between the probability of the ranking and the reversed ranking indicates the **bias between the two directions**:

$$\rho(d_i, d_j, R) = \frac{P(R^*(d_i, d_j, R)|f, D)}{P(R|, f, D) + P(R^*(d_i, d_j, R)|f, D)} \tag{17}$$

Pairwise Differentiable Gradient Descent uses this ratio to **unbias the gradient estimation**:

$$\nabla f(\cdot, \theta) \approx \sum_{d_i >_{\mathbf{c}} d_j} \rho(d_i, d_j, R) \frac{\exp^{f(\mathbf{d_i}, \theta)} \exp^{f(\mathbf{d_j}, \theta)}}{(\exp^{f(\mathbf{d_i}, \theta)} + \exp^{f(\mathbf{d_j}, \theta)})^2} (f'(\mathbf{d_i}, \theta) - f'(\mathbf{d_j}, \theta)) \tag{18}$$

Unbiasedness of Pairwise Differentiable Gradient Descent

Under the reversed pair ranking assumption, it is proven that **the expected estimated gradient** can be written as:

$$E[\nabla f(\cdot,\theta)] = \sum_{d_i,d_j} \alpha_{ij}(f'(\mathbf{d_i},\theta) - f'(\mathbf{d_j},\theta)). \tag{19}$$

Unbiasedness of Pairwise Differentiable Gradient Descent

Under the reversed pair ranking assumption, it is proven that **the expected estimated gradient** can be written as:

$$E[\nabla f(\cdot, \theta)] = \sum_{d_i, d_j} \alpha_{ij} (f'(\mathbf{d_i}, \theta) - f'(\mathbf{d_j}, \theta)). \tag{19}$$

Furthermore, the weights α_{ij} will **match the user preferences** in expectation:

$$d_i =_{rel} d_j \Leftrightarrow \alpha_{ij} = 0 \tag{20}$$

$$d_i >_{rel} d_j \Leftrightarrow \alpha_{ij} > 0 \tag{21}$$

$$d_i <_{rel} d_j \Leftrightarrow \alpha_{ij} < 0 \tag{22}$$

Thus the estimated gradient is **unbiased w.r.t. document pair preferences**.

Unbiasedness of Pairwise Differentiable Gradient Descent

Under the reversed pair ranking assumption, it is proven that **the expected estimated gradient** can be written as:

$$E[\nabla f(\cdot, \theta)] = \sum_{d_i, d_j} \alpha_{ij} (f'(\mathbf{d_i}, \theta) - f'(\mathbf{d_j}, \theta)). \tag{19}$$

Furthermore, the weights α_{ij} will **match the user preferences** in expectation:

$$d_i =_{rel} d_j \Leftrightarrow \alpha_{ij} = 0 \tag{20}$$

$$d_i >_{rel} d_j \Leftrightarrow \alpha_{ij} > 0 \tag{21}$$

$$d_i <_{rel} d_j \Leftrightarrow \alpha_{ij} < 0 \tag{22}$$

Thus the estimated gradient is **unbiased w.r.t. document pair preferences**.

However, **we don't know** what the **norms** of the **weights** α should be.

Pairwise Differentiable Gradient Descent: Method

Start with initial model θ_t.

Then indefinitely:

1. **Sample** (without replacement) a **ranking** R from the document distribution:

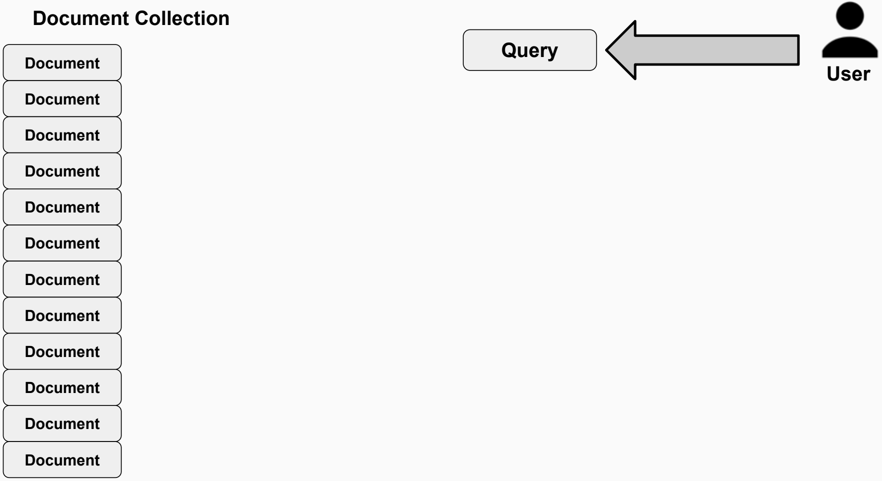
$$P(d|D,\theta_t) = \frac{\exp^{f(\mathbf{d},\theta_t)}}{\sum_{d'\in D}\exp^{f(\mathbf{d}',\theta_t)}}. \tag{23}$$

2. **Display** the ranking R to the user.
3. **Infer document preferences** from the **user clicks** $\mathbf{c}$.
4. **Update** model according to the **estimated (unbiased) gradient**:

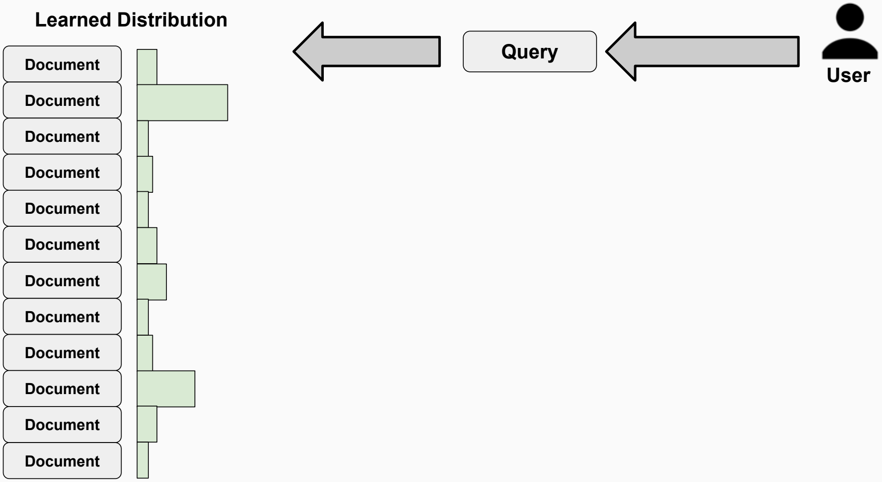
$$\nabla f(\cdot,\theta) \approx \sum_{d_i >_{\mathbf{c}} d_j} \rho(d_i,d_j,R)\nabla P(d_i \succ d_j|D,\theta). \tag{24}$$

Pairwise Differentiable Gradient Descent: Visualization

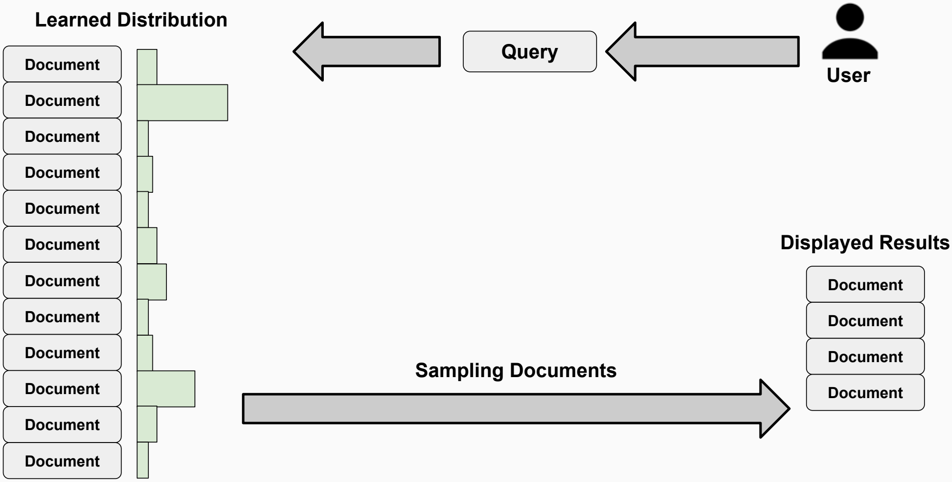
Pairwise Differentiable Gradient Descent: Visualization



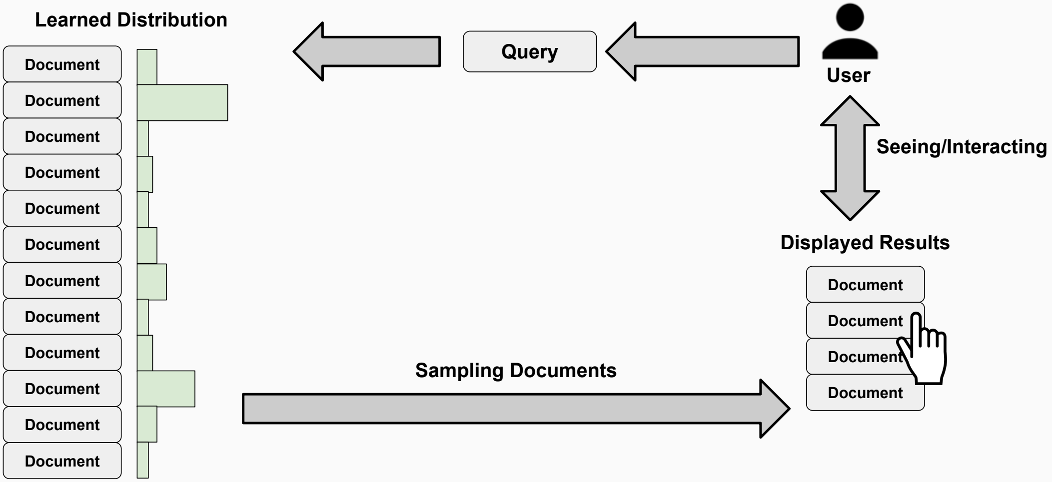
Pairwise Differentiable Gradient Descent: Visualization



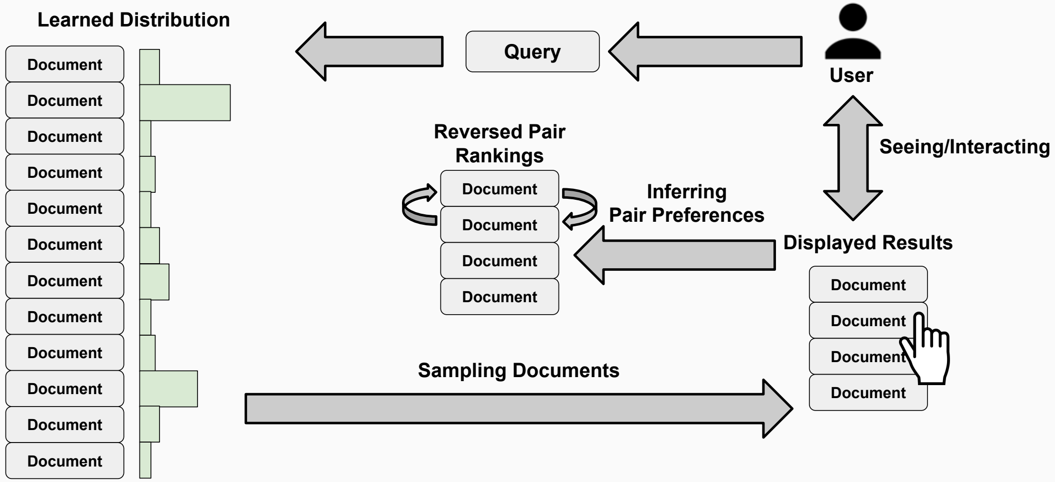
Pairwise Differentiable Gradient Descent: Visualization



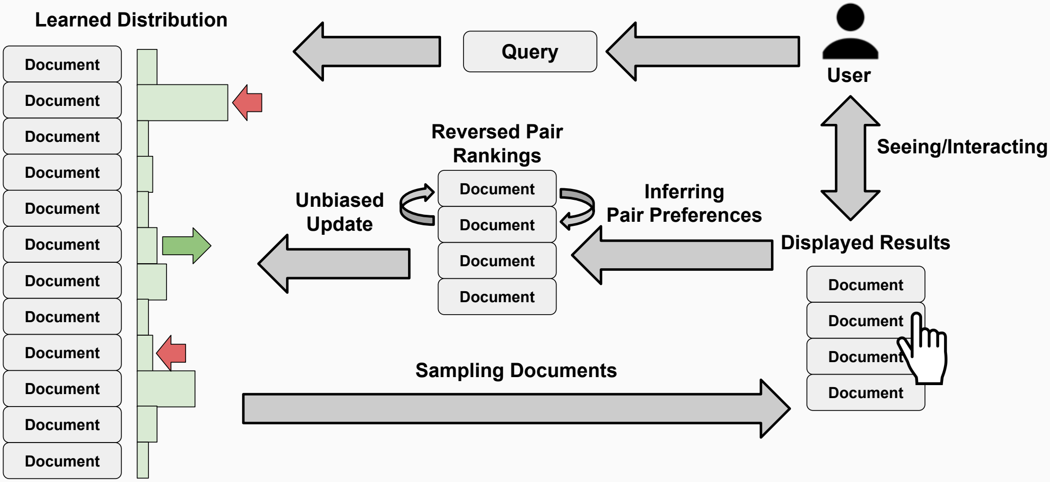
Pairwise Differentiable Gradient Descent: Visualization



Pairwise Differentiable Gradient Descent: Visualization

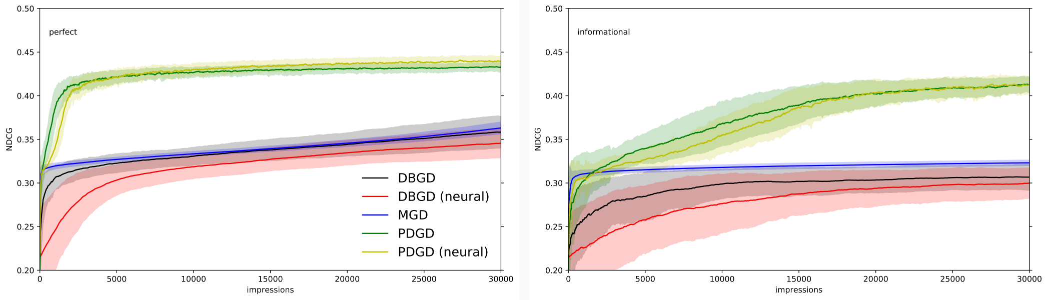


Pairwise Differentiable Gradient Descent: Visualization



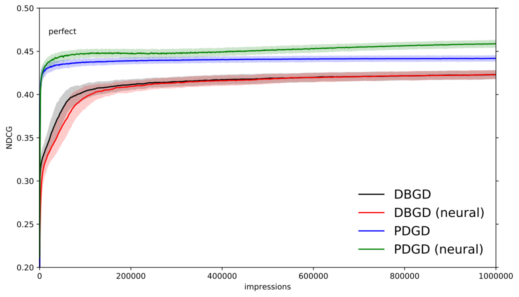
Pairwise Differentiable Gradient Descent: Results Short Term

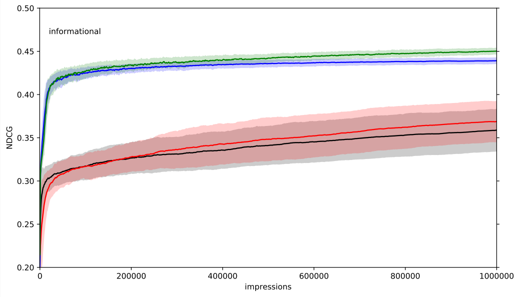
Simulated results on the MSRL-WEB10k dataset:



Pairwise Differentiable Gradient Descent: Results Long Term

Simulated results on the MSRL-WEB10k dataset:





Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

- the **first online method** to convergence **near offline levels of performance**.

Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

- the **first online method** to convergence **near offline levels of performance**.
- **considerably faster learning** (user experience) than before.

Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

- the **first online method** to convergence **near offline levels of performance**.
- **considerably faster learning** (user experience) than before.
- **performance** of an **non-linear model** to **exceed** linear model.

Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

- the **first online method** to convergence **near offline levels of performance**.
- **considerably faster learning** (user experience) than before.
- **performance** of an **non-linear model** to **exceed** linear model.
- **computationally much more efficient** than most previous methods.

Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

- the **first online method** to convergence **near offline levels of performance**.
- **considerably faster learning** (user experience) than before.
- **performance** of an **non-linear model** to **exceed** linear model.
- **computationally much more efficient** than most previous methods.
- currently **no proven regret bounds**.

Pairwise Differentiable Gradient Descent: Conclusion

With the introduction of **Pairwise Differentiable Gradient Descent** we have:

- the **first online method** to convergence **near offline levels of performance**.
- **considerably faster learning** (user experience) than before.
- **performance** of an **non-linear model** to **exceed** linear model.
- **computationally much more efficient** than most previous methods.
- currently **no proven regret bounds**.

So what's left for online learning to rank?

Future Directions

Future Directions for Online Learning to Rank

Now that performance is on the level of offline learning to rank:

- **The time is ripe for real-world experiments.**

Future Directions for Online Learning to Rank

Now that performance is on the level of offline learning to rank:

- **The time is ripe for real-world experiments.**
- How much does online learning to rank **improve over offline/historical approaches**?

Future Directions for Online Learning to Rank

Now that performance is on the level of offline learning to rank:

- **The time is ripe for real-world experiments.**
- How much does online learning to rank **improve over offline/historical approaches**?
- How effective is it for **personalization**?

Future Directions for Online Learning to Rank

Other areas to expand to:

Future Directions for Online Learning to Rank

Other areas to expand to:

- **Beyond clicks**:
  - Can we learn from dwell time, conversion, purchases, watch-time, etc.

Future Directions for Online Learning to Rank

Other areas to expand to:

- **Beyond clicks**:
  - Can we learn from dwell time, conversion, purchases, watch-time, etc.
- **Beyond ten blue links**:
  - Do methods still work in non-vertical displays? (Oosterhuis and de Rijke, 2018a).

Future Directions for Online Learning to Rank

Other areas to expand to:

- **Beyond clicks**:
    - Can we learn from dwell time, conversion, purchases, watch-time, etc.
- **Beyond ten blue links**:
    - Do methods still work in non-vertical displays? (Oosterhuis and de Rijke, 2018a).
- **Beyond relevance**:
    - Can we optimize for aspects beside relevance: e.g. result diversity?

Future Directions for Online Learning to Rank

Other areas to expand to:

- **Beyond clicks**:
  - Can we learn from dwell time, conversion, purchases, watch-time, etc.
- **Beyond ten blue links**:
  - Do methods still work in non-vertical displays? (Oosterhuis and de Rijke, 2018a).
- **Beyond relevance**:
  - Can we optimize for aspects beside relevance: e.g. result diversity?
- **Responsible A.I.**:
  - Can our algorithms guarantee to respect users during exploration?
  - Can they explain and explicitly substantiate their learned behaviour?

Conclusion

Conclusion

Covered in the second part:

- **Learn from user interactions,** not from human annotators.
  - The **true preferences** of users can be **learned** from their **behaviour**.
  - Be careful with **noise** and **bias**, avoid the **self-confirming loop**.

Conclusion

Covered in the second part:

- **Learn from user interactions,** not from human annotators.
    - The **true preferences** of users can be **learned** from their **behaviour**.
    - Be careful with **noise** and **bias**, avoid the **self-confirming loop**.
- Online Learning to Rank methods can **rank and learn simultaneously**.
    - **Interventions** allow for **very fast and efficient learning**.
    - Fast learning **enables responsive behaviour** which leads to **great user experience**.

Conclusion

Covered in the second part:

- **Learn from user interactions,** not from human annotators.
  - The **true preferences** of users can be **learned** from their **behaviour**.
  - Be careful with **noise** and **bias**, avoid the **self-confirming loop**.
- Online Learning to Rank methods can **rank and learn simultaneously**.
  - **Interventions** allow for **very fast and efficient learning**.
  - Fast learning **enables responsive behaviour** which leads to **great user experience**.
- **Recent advances** show there is **much more potential** in online learning to rank.
  - Only just been able to reach the **level of offline performance**.
  - **Many different models, settings, and extensions possible.**

Conclusion

Covered in the second part:

- **Learn from user interactions,** not from human annotators.
  - The **true preferences** of users can be **learned** from their **behaviour**.
  - Be careful with **noise** and **bias**, avoid the **self-confirming loop**.
- Online Learning to Rank methods can **rank and learn simultaneously**.
  - **Interventions** allow for **very fast and efficient learning**.
  - Fast learning **enables responsive behaviour** which leads to **great user experience**.
- **Recent advances** show there is **much more potential** in online learning to rank.
  - Only just been able to reach the **level of offline performance**.
  - **Many different models, settings, and extensions possible.**
- **Continue our work:** `https://github.com/HarrieO/OnlineLearningToRank`.

Q. Ai, K. Bi, C. Luo, J. Guo, and W. B. Croft. Unbiased learning to rank with unbiased propensity estimation. *arXiv preprint arXiv:1804.05938*, 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.

D. Dato, C. Lucchese, F. M. Nardini, S. Orlando, R. Perego, N. Tonellotto, and R. Venturini. Fast ranking with additive ensembles of oblivious and non-oblivious regression trees. *ACM Transactions on Information Systems (TOIS)*, 35(2):15, 2016.

K. Hofmann, A. Schuth, S. Whiteson, and M. de Rijke. Reusing historical interaction data for faster online learning to rank for ir. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 183–192. ACM, 2013a.

K. Hofmann, S. Whiteson, and M. de Rijke. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Information Retrieval*, 16(1):63–90, 2013b.

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, 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.

J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, pages 817–824, 2008.

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. Balancing speed and quality in online learning to rank for information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 277–286. ACM, 2017a.

H. Oosterhuis and M. de Rijke. Sensitive and scalable online evaluation with theoretical guarantees. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 77–86. ACM, 2017b.

H. Oosterhuis and M. de Rijke. Ranking for relevance and display preferences in complex presentation layouts. In *ACM SIGIR Forum*. ACM, 2018a.

H. Oosterhuis and M. de Rijke. Differentiable unbiased online learning to rank. In *Proceedings of the 2018 ACM on Conference on Information and Knowledge Management*. ACM, 2018b.

H. Oosterhuis, A. Schuth, and M. de Rijke. Probabilistic multileave gradient descent. In *European Conference on Information Retrieval*, pages 661–668. Springer, 2016.

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.

A. Schuth, H. Oosterhuis, S. Whiteson, and M. de Rijke. Multileave gradient descent for fast online learning to rank. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 457–466. ACM, 2016.

R. S. Sutton and A. G. Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

H. Wang, R. Langley, S. Kim, E. McCord-Snook, and H. Wang. Efficient exploration of gradient space for online learning to rank. *arXiv preprint arXiv:1805.07317*, 2018a.

X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to rank with selection bias in personal search. In *SIGIR*, 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, 2018b.

Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1201–1208. ACM, 2009.

T. Zhao and I. King. Constructing reliable gradient exploration for online learning to rank. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1643–1652. ACM, 2016.

Acknowledgments

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