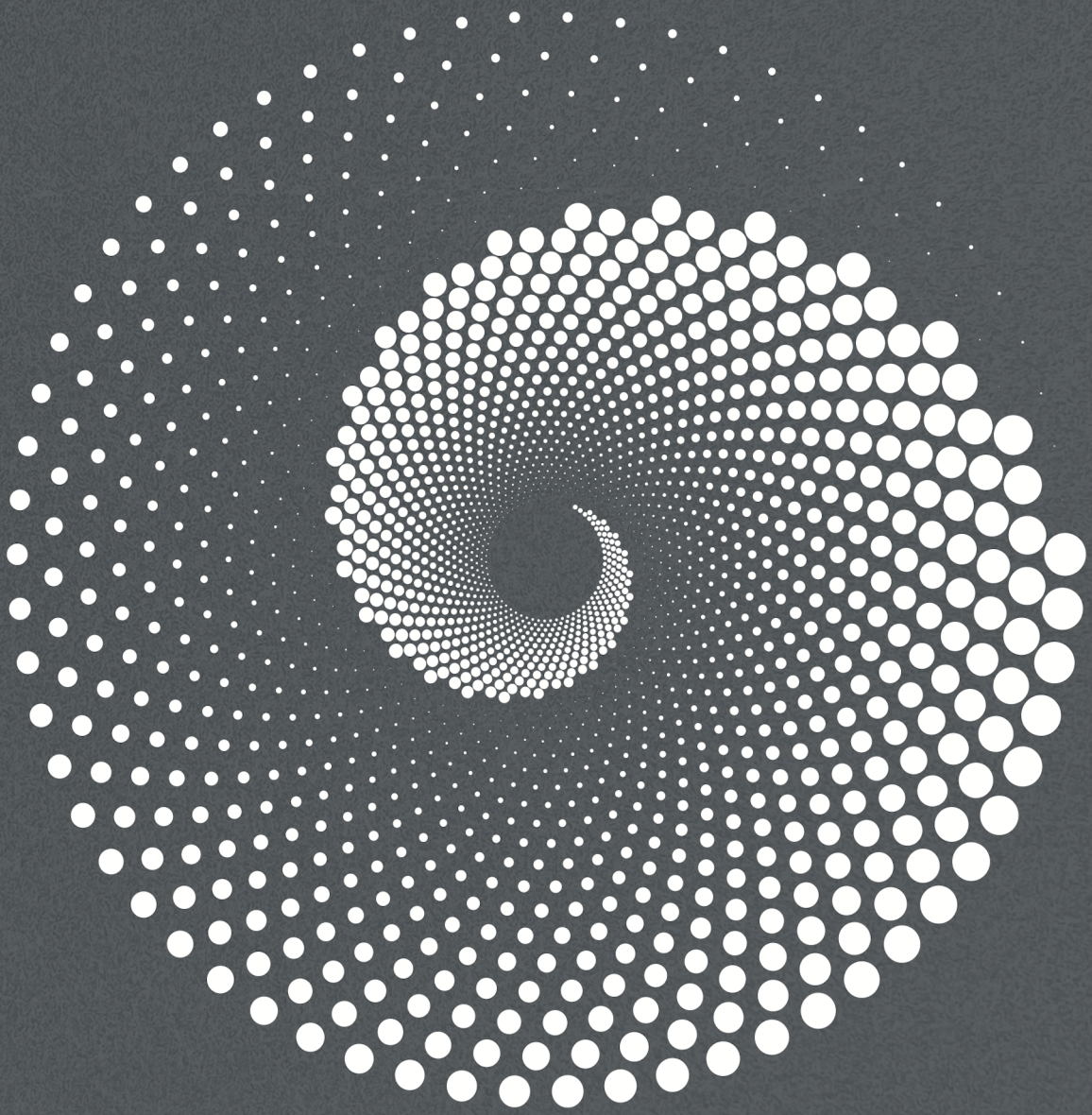
Re-examining assumptions in fair and unbiased learning to rank

Vardasbi, A.

Publication date
2023
Document Version
Final published version

Re-Examining Assumptions in Fair and Unbiased Learning to Rank



Ali Vardasbi

Re-Examining Assumptions in Fair and Unbiased Learning to Rank

Ali Vardasbi

Re-Examining Assumptions in Fair and Unbiased Learning to Rank

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties
ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op woensdag 6 december 2023, te 10:00 uur

door

Ali Vardasbi

geboren te Joybar

Promotiecommissie

Promotor:	Prof. dr. Maarten de Rijke	Universiteit van Amsterdam
Co-promotor:	Dr. Mostafa Dehghani	Google Brain
Overige leden:	Prof. dr. Asia Biega	Max Planck Institute
	Prof. dr. Thorsten Joachims	Cornell University
	Prof. dr. Evangelos Kanoulas	Universiteit van Amsterdam
	Dr. Sara Magliacane	Universiteit van Amsterdam
	Prof. dr. Cees Snoek	Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The research was supported by Elsevier and the Netherlands Organisation for Scientific Research (NWO) under project number 612.001.551.

Cover by Poshstocker – shutterstock.com
Printed by Ridderprint, Amsterdam

ISBN: 978-94-6483-521-2

Acknowledgements

First I want to thank my supervisor Maarten de Rijke whose guidance, expertise, and belief in my abilities kept me focused and motivated throughout this thesis. I truly appreciate the time and effort you generously dedicated to shaping this research and pushing me to reach new heights. I am thrilled to have had the opportunity to work with you, learn from you, and benefit from your advice and support over the past 55 months.

Second I would like to thank my co-promotor for the first two years of my PhD, Ilya Markov. During our collaboration on writing two papers, you have helped me in enhancing the clarity and expression of my ideas within the paper. Thank you Ilya!

Mostafa, my dear friend, and co-promotor during the final two years of my PhD, collaborating with you was always a delightful experience — from the initial brainstorming to conducting experiments and writing the paper. Your insights and expertise were invaluable at every step of our collaboration, and your friendship made the journey even more enjoyable.

Next, I would like to thank all my committee members: Asia, Thorsten, Evangelos, Sara, and Cees. I am honored that you are all part of my PhD committee.

During my four years at IRLab (previously known as ILPS) I enjoyed meeting many people whose support and collaborative efforts have helped me a lot in the successful completion of this thesis. I would like to thank: Ming, Jin, Ana, Maartje, Pengjie, Vera, Shashank, Philipp, Mozhdeh, Ali A, Sam, Mahsa, Maria, Georgios, Julien, Petra, Yangjun, Ziming, Jie, Mariya, Chuan, Svitlana, Olivier, Yifan, Chang, David, Sami, Shaojie, Sebastian, Fei, Christof, Nikos, Maurits, Mohammad, Spyretta, Dan, Peilei, Anna, Xinyi, Arezoo, Wanyu, Harrie, Maarten M, Jiahuan, Hamid, Andrew, Songgaojun, Panagiotis, Jia-Hong, Ruqing, Zahra, Gabriel, Romain, Amir, Ruben, Jingwei, Barrie, Pooya, Yibin, Antonios, Yuanna, Thong, Vaishali, Jingfen, Thilina, Clara, Clemencia, Zihan, Weijia, Yuyue, Arian, Amin, Ivana, Gabrielle, and Negin.

I had the opportunity to complete two wonderful internships during my PhD. Telmo, Robin, Stephen, and everyone else at Apple, thank you for all the engineering in Aachen, I learned a lot from you! Enrico, José, Gustavo, Hugues, Alice, Maryam, Claudia, Paul and everyone else at Spotify, I really enjoyed being part of your amazing band.

I want to express my deepest gratitude to my parents, whose unwavering love and encouragement have been the foundation of my success. Your endless support, both emotionally and financially, has enabled me to pursue my dreams and reach this significant milestone. I am forever grateful for everything you have done for me.

I am profoundly thankful to my wife, Atieh, whose love, patience, and support sustained me throughout this demanding academic journey. Your belief in me, your understanding during long nights near the deadlines, and your constant encouragement were the cornerstones of my perseverance. Thank you for being the source of my strength and happiness.

Ali Vardasbi
Amsterdam, October 2023

Contents

1 Introduction

One of the core problems in information retrieval (IR) concerns ranking a collection of data entities, such as documents, multi-media or commercial products, according to their relevance to a particular information need, such as a textual query or a user's purchase history [9]. Due to the rapid expansion of online information and the challenges associated with finding desired information, the need for effective information retrieval systems has increased significantly [77]. Consequently, online search engines and recommendation systems have become means to help satisfy users' information needs. As a central component of search engines, a ranker is responsible for sorting documents based on their degree of relevance to the queries. In the classic literature on IR, numerous heuristic ranking models have been proposed to address this challenge [113]. With the advent of abundant training data as well as machine learning successes in other fields, researchers in the IR community have focused on leveraging machine learning technologies to construct powerful ranking models. These models are referred to as *learning to rank* (LTR) methods.

Traditional LTR models require explicit relevance labels as their supervision signal, produced by expert annotators [79]. However, obtaining such annotations in great quantities is both expensive and time-consuming [25, 102]. Furthermore, both the information, as well as user taste, are continuously changing, and the preferences of online users may not be completely aligned with that of expert annotators [72, 109]. The ever-expanding application of online search engines provides huge quantities of user interactions to system designers, at no extra cost. Furthermore, unlike expert annotations, interactions from users are an indication of the actual individual user preferences. Research on using user interactions for learning and evaluation of ranking is divided into two families: the online family where the optimal ranker is learned through adaptive interventions with the search results as they are shown to the user [46, 73, 76, 95, 144]; and the counterfactual family that only uses historical logs as their supervision source [1, 58, 131]. In this thesis, we focus on the second family, i.e., *counterfactual learning to rank* (CLTR).

The great advantages of user interactions over manually labeled data for LTR in terms of cost and reflecting actual user preferences, come with several issues. Most importantly, as a form of implicit feedback, user interactions suffer from noise and bias. Consider, for instance, clicks on the items in a search engine result page (SERP) as user interactions. Clicks are noisy in the sense that a non-relevant item may receive a click

or a relevant item may be skipped. As long as the signal-to-noise ratio is acceptable, the effect of noise is mitigated by averaging over a large number of clicks. However, things are different for bias. Clicks are biased in the sense that different items are treated differently by the user, based on their position, presentation, or other contextual factors. For example, *position bias*, a well-known type of bias [30], occurs because users are more likely to examine results at higher ranks. As a consequence, an item may receive more clicks simply because it was displayed at a high rank, not because it was more relevant to the user's information need. Other types of bias include *item-selection bias*: not all items can be displayed at once [96, 99]; *presentation bias*: items are presented in different manners which affects how users click on them [138]; and *trust bias*: users are more likely to click incorrectly on higher ranked items [2, 57]. Since user interactions come with bias, an important focus of CLTR research lies in developing methods to *correct* for the bias of interactions [58, 130].

Learning an unbiased ranker from user interactions helps to build a more effective search engine. However, as one of the important sources for seeking information and obtaining knowledge, search engines can shape the knowledge of crowds by the way in which they represent the results. For example, through a user study, Vlasceanu and Amodio [129] show that exposure of users to biased results of a search engine shapes their cognitive concepts and decisions. Consequently, due to their significant impact on different aspects of everyday life, it is important to make sure that search engines meet fairness concerns such as allocating fair resources, such as exposure, to different stakeholders. In particular, it is important that sensitive groups and individuals are not hurt by the way search engine and recommendation results are represented and that existing discrimination is not reinforced by ranking algorithms.

Compared to traditional LTR that learns from explicit feedback of expert annotators and the goal is to maximize utility only, research on both unbiased and fair LTR is relatively new and it heavily relies on simplifying *assumptions*. For example, the well-known inverse propensity scoring (IPS) method for bias correction is based on the assumption that the clicks only have position bias, but not trust bias [2]. Or the treatment-based fairness of exposure paradigm implicitly assumes that making the exposure distribution of items fair helps to mitigate the discrimination reinforcement in ranking systems [114]. In this thesis, we re-examine a number of such assumptions in the literature on unbiased and fair LTR. We propose several methods for unbiased and fair LTR that are more general than the existing ones, in the sense that they rely on fewer assumptions, or that they are applicable in more situations.

1.1 Research Outline and Questions

The interest on counterfactual learning to rank (CLTR) has peaked with the introduction of inverse propensity scoring (IPS) to the context of LTR [58, 131]. Even though IPS does not rely on any specific click model in theory [58], most IPS-based CLTR experiments rely on the position-based model (PBM), where the probability of examining a result depends on the result's rank only [4, 58, 130, 131]. Consequently, PBM fails to represent the cascade user click behavior, where a user scans a search engine result page (SERP) from top to bottom and each next click depends on the previous clicks [30].

In Chapter 2 we re-examine the widespread assumption of PBM in IPS and deal with the following question:

RQ1 How to go beyond PBM and consider user cascading behavior when using IPS for position bias correction?

We first show by semi-synthetic experiments that when the user behavior follows a cascade model, using PBM-IPS to correct for position bias is not effective, in that training with more clicks does not help to fill the gap towards the ranking quality of a system trained on the explicit feedback of expert annotators, i.e., the full information case. Then, we derive closed-form formulas for click propensities in three widely used cascade-based click models, namely the dependent click model (DCM) [50], dynamic Bayesian network (DBN) [26], and click chain model (CCM) [49] to fill in the gaps of the PBM-based CLTR performance on cascade-based clicks.

Continuing to explore the assumptions underlying IPS, we notice a fundamental assumption in modeling the relation between click probability and relevance: IPS is based on the assumption that click probabilities can be modeled by a position-dependent scaling transformation on relevance. However, user studies show there are other types of bias, such as trust bias, where users are more likely to click incorrectly on higher-ranked items, mainly because of their trust in the search engines [57]. With trust bias, as a result of the noticeable amount of incorrect clicks on higher ranks, a simple scaling transformation is not enough to describe the relation between click probability and relevance. This leads to our next research question:

RQ2 How to effectively correct for trust bias in user click data?

In Chapter 3 we address this question by first proving that no IPS method is able to correct for trust bias and then introducing a novel correction method, called affine correction (AC). We show that our affine correction is a generalization to IPS, meaning that without trust bias, they lead to the same estimator, but with trust bias only IPS is biased. Our extensive semi-synthetic experiments show the effectiveness of our AC method for correcting position and trust bias.

The proofs of the unbiasedness of IPS and AC depend on accurate estimations of the bias parameters. This, in turn, depends on obtaining accurate relevance estimations, which is as hard as the LTR problem itself: LTR is the problem of ranking items based on their relevance to the query [79]. In the literature, this cyclic dependency is solved by a regression-based EM (rbEM) algorithm that simultaneously learns the ranker as well as the bias parameters [2, 122, 131]. However, integration of a regression function into the standard expectation maximization (EM) leads to a number of practical limitations, the most important of which is a lack of guarantees that rbEM converges to a zero gradient. In Chapter 4 we address this problem by asking the following question:

RQ3 Is it possible to break the cyclic dependency between relevance and bias parameters when correcting for position and trust bias?

We answer this question positively by proposing a novel correction method, mixture-based correction (MBC). In MBC, we assume that the probability of seeing a specific click-through rate (CTR) for an item at a position in a ranking is a mixture of CTR

probabilities for relevant and non-relevant items appearing on that position. Our unbiasedness proof of MBC, unlike IPS and AC, does not rely on prior knowledge of the bias parameter values. We also go back to RQ1 and show that when clicks adhere to cascading models, while PBM is assumed by the correction methods, both MBC and AC will remain biased, but MBC is more robust, i.e., its ranking performance is affected less compared to AC.

In the first three chapters of this thesis, we re-examined prevalent assumptions for *bias correction* in CLTR. As a closely related subject, in the last two chapters, we focus on assumptions for *fairness optimization* in ranking.

Our concern in fairness of ranking is to ensure equitable and indiscriminate representation or visibility of items, individuals, or groups in a given context, such as expert search, job application, or news recommendation. When optimizing for fairness, the best a ranking system can do is to arrange the exposure such that items with similar utility receive comparable exposure by users. However, this alone is insufficient. Studies show that users' perceptions of an item's group membership affect their judgments about the utility of items [61, 67, 118]. Therefore, even with equal exposure, users may judge equally-relevant items from two different groups differently and one group may receive fewer clicks than the other. We refer to this behavior as *group membership bias* and study it in Chapter 5. Specifically, we focus on answering the following question:

RQ4 What is the impact of group bias on the quality and fairness metrics in a ranking and how to correct for this bias, without substituting equality for equity?

To answer the first part of this question, we provide three theorems that show the impact of group bias on normalized discounted cumulative gain (NDCG), a metric for ranking quality, as well as disparate treatment ratio (DTR) and expected exposure loss (EEL), two metrics for fairness in ranking. We show that group bias has a negative impact on both ranking quality and fairness metrics. Both DTR and EEL are merit-based fairness metrics, where the goal is to give each group an amount of exposure according to its merit. This is referred to *equity* and is in contrast to *equality*, since the implicit assumption in merit-based fairness metrics is that different groups may not necessarily have the same utility and should not necessarily get equal exposure. Noting this distinction, we argue that naive correction of group bias may lead to equality-instead-of-equity, and deal with the second part of RQ4 by proposing an amortized correction method that preserves equity.

Similar to other types of bias such as position and trust bias (partially dealt with in Chapters 2, 3, and 4 of this thesis), group bias degrades the ranking quality of LTR models. However, unlike those types of bias, it also directly impacts the fairness measurement of a ranking in the sense that with group bias, a supposedly fair ranking is not truly fair.

Our final research question centers around a general framework for optimizing different fairness metrics for ranking. Different fairness measures have been proposed in the literature based on different definitions of fairness and aimed at different environments [e.g., 32, 114]. Using the REINFORCE algorithm [133], and sampling from a Plackett-Luce (PL) distribution, it is possible to optimize any fairness objective function on permutations [115]. This approach works well for optimizing stochastic ranking evaluation metrics such as EEL, provided that a large number of repeating sessions

are available for a given query. However, when the number of repeating sessions for a query is very small, the high variance of PL [43] leads to sub-optimal results. In the extreme case, we can think of a deterministic ranker where the ranking is fixed through different sessions of a query. We re-examine the assumption for fairness optimization that a stochastic policy is available with a large number of repeating sessions per query and seek to answer the following question:

RQ5 Is it possible to have a single general fairness optimization method that performs well for both stochastic and deterministic rankings?

In Chapter 6 we introduce a new representation for the distribution of permutations, called probabilistic permutation graph (PPG), constructed by pairwise inversion probabilities instead of the pointwise logits in the PL representation. Compared to PL, using the PPG representation for fairness optimization leads to more robust results in finding a deterministic fair permutation for one session, while having comparable performance for expected fairness over larger numbers of sessions. Furthermore, our experiments show that in scenarios such as tabular search, where high-quality estimates of utility are available, PPG performs outstandingly well, with a considerable gap to PL-based fairness optimization.

1.2 Main Contributions

This section describes a list of the main contributions in this thesis.

1.2.1 Algorithmic Contributions

1. Closed-form formulas for using three cascade-based models with inverse propensity scoring (IPS); see Chapter 2.
2. Affine correction method to correct for position and trust bias; see Chapter 3.
3. Mixture-based correction method to correct for position and trust bias, while breaking the cyclic dependency between bias parameters and relevance estimation; see Chapter 4.
4. Amortized correction for group membership bias; see Chapter 5.
5. Sampling from a probabilistic permutation graph (PPG) distribution over permutations; see Chapter 6.
6. PPG search for general fairness optimization in ranking; see Chapter 6.

1.2.2 Theoretical Contributions

1. A proof that in cascade-based models, when using IPS, query-dependent relevance probabilities can be replaced with session-dependent clicks; see Equation (2.4).
2. A proof that no IPS method can correct for trust bias; see Theorem 3.1.

3. A proof that the affine estimator is unbiased w.r.t. trust bias; see Theorem 3.2.
4. A proof that the mixture-based correction (MBC) can fully recover the relevance signal from the click-through rates; see Theorem 4.1.
5. Proofs for the impact of group bias on the ranking quality and two merit-based fairness metrics; see Section 5.4.
6. An approximation for efficiently computing the log derivative of the probability distribution of PPG; see Section 6.4.

1.2.3 Empirical Contributions

1. Validating the effectiveness of cascade model IPS for correcting the position bias when clicks adhere to cascade-based models; see Chapter 2.
2. Validating the effectiveness of the AC method for correcting position and trust bias; see Chapter 3.
3. A comparison between MBC and AC to correct for position and trust bias in different click models; see Chapter 4.
4. Experiments to show the impact of group membership bias on the quality and fairness of rankings, both in tabular search and general LTR model regimes; see Chapter 5.
5. Validating the effectiveness of amortized correction method for group membership bias; see Chapter 5.
6. A comparison between PPG and PL for optimizing two fairness metrics in deterministic and stochastic ranking systems, and both in tabular search and general LTR model regimes; see Chapter 6.

1.3 Thesis Overview

The thesis starts with the current chapter. In this chapter we introduce the main subject of the thesis, which is re-examining assumptions in unbiased and fair counterfactual learning to rank (CLTR). Next, in Chapter 2, we re-examine the position-based model (PBM) assumption in inverse propensity scoring (IPS) for position bias correction and propose a formulation for IPS under the cascade-based models (CBM) assumption. After that, in Chapter 3, we address trust bias and propose a novel correction method, called affine correction (AC). This is followed by Chapter 4, where we analyze the dependency of AC and IPS to accurate bias parameter estimation and propose a correction method that does not rely on bias parameter estimation the way AC and IPS do. These three chapters constitute the first part of this thesis, which is the bias part. In the bias part we re-examine three assumptions in the unbiased CLTR literature.

Moving to the second part of the thesis, we draw our attention to fairness in ranking. In Chapter 5 we introduce group membership bias, a new type of bias that directly

impacts fairness metrics. We give theoretical and empirical analysis of the impact of this type of bias on the quality and fairness of ranking and propose a correction method for it. Finally, in Chapter 6, we propose a new representation for the distribution of permutations to replace the widely used Plackett-Luce (PL) distribution for fairness optimization. With the help of this new distribution, we build up a general fairness optimization method that is effective in both deterministic and stochastic ranking systems.

All chapters are based on separate articles. We aim to keep the articles in their original state as much as possible. Because of this, it is unavoidable to have some overlap in the description of some baseline methods or core notation.

1.4 Origins

In this section we list the publications that form the basis for each chapter:

Chapter 2 is based on the conference paper:

- A. Vardasbi, M. de Rijke, and I. Markov. Cascade model-based propensity estimation for counterfactual learning to rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pages 2089–2092, 2020.

The method and proofs were developed by Vardasbi with help of Markov. The experiments were designed and run by Vardasbi. All authors contributed to the text. Vardasbi did most of the writing.

Chapter 3 is based on the conference paper:

- A. Vardasbi, H. Oosterhuis, and M. de Rijke. When inverse propensity scoring does not work: Affine corrections for unbiased learning to rank. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, pages 1475–1484, 2020.

The method and proofs were equally developed by Vardasbi and Oosterhuis. The experiments were designed and run by Vardasbi. All authors contributed equally to the text.

Chapter 4 is based on the conference paper:

- A. Vardasbi, M. de Rijke, and I. Markov. Mixture-based correction for position and trust bias in counterfactual learning to rank. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 1869–1878, 2021.

The method and proofs were developed by Vardasbi. The experiments were designed and run by Vardasbi. All authors contributed to the text. Vardasbi did most of the writing.

Chapter 5 is based on the paper:

- A. Vardasbi, M. de Rijke, F. Diaz, and M. Dehghani. Group membership bias. *arXiv preprint arXiv:2308.02887*, 2023.

The method and proofs were developed by Vardasbi. The experiments were designed and run by Vardasbi. All authors contributed to the text. Vardasbi and De Rijke did most of the writing.

Chapter 6 is based on the conference paper:

- A. Vardasbi, F. Sarvi, and M. de Rijke. Probabilistic permutation graph search: Black-box optimization for fairness in ranking. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 715–725, 2022.

The method and proofs were developed by Vardasbi. Most of the experiments were designed and run by Vardasbi. Experiments on the DTR metric were run by Sarvi. All authors contributed to the text. Vardasbi did most of the writing.

The writing of the thesis also benefited indirectly from work on the following publications:

- J. Ehrhardt, T. Spinde, A. Vardasbi, and F. Hamborg. Omission of information: Identifying political slant via an analysis of co-occurring entities. *Schmidt, Wolff (Eds.): Information between Data and Knowledge*, pages 80–93, 2021.

- H. Bonab, M. Aliannejadi, A. Vardasbi, E. Kanoulas, and J. Allan. Cross-market product recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 110–119, 2021.

- A. Vardasbi, M. de Rijke, and M. Dehghani. Intersection of parallels as an early stopping criterion. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, pages 1965–1974, 2022.

- M. Li, A. Vardasbi, A. Yates, and M. de Rijke. Repetition and exploration in sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2532–2541, 2023.

- A. Vardasbi, T. P. Pires, R. M. Schmidt, and S. Peitz. State spaces aren't enough: Machine translation needs attention. In *The 24th Annual Conference of the European Association for Machine Translation*, EAMT '23, 2023.

- F. Sarvi, A. Vardasbi, M. Aliannejadi, S. Schelter, and M. de Rijke. On the impact of outlier bias on user clicks. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 18–27, 2023.

Part I

Bias

2

Click Model Assumptions Matters: Cascade Model-based Inverse Propensity Scoring

Unbiased counterfactual learning to rank (CLTR) requires click propensities to compensate for the difference between user clicks and true relevance of search results via inverse propensity scoring (IPS). Current propensity estimation methods assume that user click behavior follows the position-based model (PBM) and estimate click propensities based on this assumption. However, in reality, user clicks often follow the cascade model (CM), where users scan search results from top to bottom and where each next click depends on the previous one. In this cascade scenario, PBM-based estimates of propensities are not accurate, which, in turn, hurts CLTR performance. Concerning RQ1, in this chapter, we propose a propensity estimation method for the cascade scenario, called *cascade model-based inverse propensity scoring* (CM-IPS). We show that CM-IPS keeps CLTR performance close to the full-information performance in case the user clicks follow the CM, while PBM-based CLTR has a significant gap towards the full-information. The opposite is true if the user clicks follow the PBM instead of the CM. Finally, we suggest a way to select between CM- and PBM-based propensity estimation methods based on historical user clicks.

2.1 Introduction

Traditional learning to rank (LTR) and online LTR require explicit relevance labels and intervention through search engine results, respectively [79, 95]. In contrast, CLTR only requires historical click logs for learning. Obtaining and using historical click logs incurs no extra cost and does not impose any risk of reduced user satisfaction. More importantly, such benefits come without any significant reduction in LTR performance [4, 58, 130, 131]. However, user clicks are known to suffer from different types of bias, such as position bias, selection bias, trust bias, etc. [29]. Due to these types of

This chapter was published as: A. Vardasbi, M. de Rijke, and I. Markov. Cascade model-based propensity estimation for counterfactual learning to rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pages 2089–2092, 2020.

bias, each result on a search engine result page (SERP) has a different *propensity* of being clicked. Since CLTR learns from user clicks, it should take those propensities into account. To make CLTR unbiased, the IPS method has been introduced in [58, 130].

In IPS-based CLTR, click models are used to estimate propensities [58]. Even though the theoretical IPS method does not rely on any specific click model [58], current IPS-based CLTR experiments rely on the PBM [4, 58, 130, 131]. In PBM, the probability of examining a result depends on the result's rank only and not on any other context, such as clicks on other items.

Although PBM is a well-performing click model [29], it does not always approximate user clicks well [47]. Importantly, PBM fails to represent the cascade user click behavior, where a user scans a SERP from top to bottom and where each next click depends on the previous click [30] – such behavior is often observed in practice [29, 47]. In this case, PBM-IPS estimators are not accurate and CLTR performance drops considerably. Look, for example, at Figure 2.1. There, the PBM-IPS CLTR is trained over two different simulated click logs: one following PBM and the other following dependent click model (DCM) [50] one of the popular cascade-based models. When trained on the same number of clicks and the same queries, PBM-IPS performs significantly better on the PBM than DCM simulated clicks. The "Full Info" legend in this plot shows the performance of LTR trained on real relevance tags instead of simulated clicks. PBM-IPS performs close to the full-info only when trained on PBM simulated clicks.

Based on the above observation, in this chapter, we address the following research question:

RQ1 How to go beyond PBM and consider user cascading behavior when using IPS for position bias correction?

We first experimentally validate this observation for different parameter settings. We apply PBM-IPS unbiased CLTR on various sets of simulated clicks and show that PBM-IPS CLTR only performs well when the simulated clicks are drawn based on the PBM. The significance of these results is also in noticing that the current IPS unbiased CLTR papers all use PBM-IPS estimation in their experiments [4, 58, 131]. To fill in the gaps of the PBM-based CLTR performance on cascade-based clicks, we provide CM-IPS and derive closed-form formulas for click propensities in three widely used cascade-based click models, DCM [50], dynamic Bayesian network (DBN) [26], and

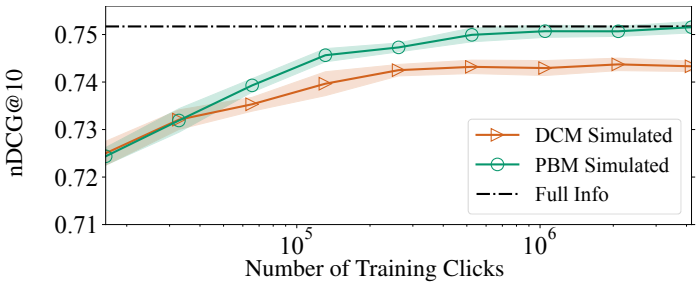


Figure 2.1: Performance of PBM-IPS CLTR on different sets of simulated clicks.

Table 2.1: Notation.

Parameter	Description
x_j	representation of query, document pair at position j
$c_j \in \{0, 1\}$	click on result x_j at position j
$r_j \in \{0, 1\}$	relevance of result x_j at position j
$e_j \in \{0, 1\}$	examination of result x_j at position j
$\mathcal{X}_q$	ordered set of the results corresponding query q
q_S	the query of session S

click chain model (CCM) [49]. We experimentally show the effectiveness of our derived propensity formulas for DCM.

To sum up, in this chapter we are interested in the following research questions:

RQ1.1 Can a PBM-IPS CLTR effectively learn from clicks when the user click behavior is closer to cascade-based models?

RQ1.2 What are the cascade model-based propensity alternatives that are more suitable for IPS CLTR in the presence of cascade user click behavior?

Table 2.1 summarizes the notation we use in this chapter.

2.2 Related Work

Click models. Click models model user behavior. Most click models factorize the click probability into two independent probabilities: the probability of examination and the probability of attractiveness (or relevance) [29]. In order to predict the examination probability, various probabilistic click models with different assumptions have been proposed. The *position-based model* (PBM) assumes that the examination probability of a result only depends on its rank in the result list. There are several cascade-based models that assume that a user examines the results on the SERP linearly, from top to bottom, until she is satisfied with a result and abandons the session. See Section 2.3 for details. The true click model of a given (set of) click log(s) is not known, but unbiased CLTR requires the knowledge of the click propensities. Consequently, a click model is usually assumed for the click logs and the click propensities are estimated based on that assumed click model [58, 131].

Learning PBM propensities. In LTR, it is common to optimize the sum of some performance metric only over relevant training documents [2, 4, 58]. However, in the click logs, the r_x are unknown. What is observed is c_x. According to the examination hypothesis, clicks appear on the relevant results that are also *examined*. Hence, the click signals are biased by the examination probability. To debias these signals, Joachims et al. [58] propose to use the so-called *inverse propensity scoring* (IPS) method:

$$\hat{\mathscr{L}}_S = \sum_{x_j \in \mathcal{X}_{q_S}} \frac{c_j^{(S)} \cdot \mathcal{L}_{x_j}}{P\left(E_j = 1 \mid q_S\right)}, \tag{2.1}$$

where $P\left(E_j = 1 \mid q_S\right)$ is the marginalized examination probability over all the sessions with the same query:

$$P(E_j = 1 \mid q) = E_{S|q_S=q}\left[P\left(E_j = 1 \mid S\right)\right]. \tag{2.2}$$

So far, LTR models dealing with click signals assume PBM for the clicks (in practice) and estimate the propensities based on this assumption [4, 58]. In PBM one can write:

$$P_{\text{PBM}}(E_j = 1 \mid q) = P_{\text{PBM}}(E_j = 1) = \theta_j \tag{2.3}$$

Existing CLTR work builds on the PBM assumption [4, 131]. But PBM is not necessarily the best fitting model in all situations.

Cascade bias. Chandar and Carterette [24] discuss the idea that the existing CLTR methods do not consider the cascade bias, i.e. higher ranked relevancy dependent examination of items. They focus on counterfactual evaluation of rankers and show that, in presence of cascade bias, a *Context-Aware* IPS ranker has a higher Kendall's tau correlation with the full information ranker than that of a simple IPS. Though the basic ideas of [24] are the same as the current chapter, there at least four important differences: (i) We propose closed-form propensity formulas for cascade models, while they directly estimate the propensities using result randomization. (ii) We employ CM-IPS in CLTR to learn the ranker, as opposed to evaluating the rankers. (iii) Unlike them, we prove that the hidden click probabilities can be replaced with observed clicks without violating the unbiasedness. (iv) We use real query-document features for training our CLTR, whereas they only use fully simulated features in their experiments.

2.3 Cascade Model-based Propensity Estimation

We derive recursive formulas for propensity estimation in popular cascade models based on clicks on a query session. We use CM-IPS to refer to an IPS method that uses these formulas. For each of DCM, DBN and CCM, we derive the examination probability at a position, based on the model parameters and the clicks over the previous positions. This exercise is not necessary for PBM since the propensities are the parameters themselves and examination at a position is independent of user behavior on other positions.

Before proceeding to specific propensity formulas for each click model, we need to rewrite the original IPS method proposed in [58] to make it more suitable for cascade-based models (CBM). Let us define the IPS per query loss as $\hat{\mathscr{L}}_q = \sum_{S|q_S=q} \hat{\mathscr{L}}_S$. In what follows we show that, in CBM, if the marginalized $P(E_j = 1 \mid q)$ in (2.1) is replaced with the session dependent probabilities $P(E_j = 1 \mid C_{<j})$, the per query loss will remain asymptotically unchanged. For brevity, we will drop the summation over positions as well as the $q_S = q$ condition.

$$\begin{aligned}
\hat{\mathcal{L}}_q[j] =& \sum_S \frac{c_j^{(S)} \cdot \mathcal{L}_{x_j}}{P\left(E_j = 1 \mid q\right)} = \frac{\mathcal{L}_{x_j}}{P\left(E_j = 1 \mid q\right)} \sum_S c_j^{(S)} \\
=& N_q \cdot \frac{\mathcal{L}_{x_j} \cdot P(C_j = 1 \mid q)}{P\left(E_j = 1 \mid q\right)} = N_q \cdot P(R_j = 1) \cdot \mathcal{L}_{x_j} \\
=& \sum_S \frac{P(C_j = 1 \mid c_{<j}^{(S)})}{P(E_j = 1 \mid c_{<j}^{(S)})} \cdot \mathcal{L}_{x_j} \overset{(2.5)}{=} \sum_S \frac{c_j^{(S)} \cdot \mathcal{L}_{x_j}}{P(E_j = 1 \mid c_{<j}^{(S)})}
\end{aligned} \tag{2.4}$$

where the last equality is empirically valid based on Eq. (2.5) below. In CBM, we can write for a general function g depending on the clicks before, and including, position j:

$$\begin{aligned}
&\sum_{S} P(C_j = 1 \mid c_{<j}) \cdot g(c_{\leq j}) \\
&= \sum_{c_{<j} \in \{0,1\}^{j-1}} |S_{c_{<j}}| \cdot P(C_j = 1 \mid c_{<j}) \cdot g(c_{\leq j}) \\
&\simeq \sum_{c_{<j} \in \{0,1\}^{j-1}} \sum_{S \in S_{c_{<j}}} c_j^{(S)} \cdot g(c_{\leq j}) = \sum_{S} c_j^{(S)} \cdot g(c_{\leq j})
\end{aligned} \tag{2.5}$$

where $S_{c_{<j}} = \{S \mid c_{<j}^{(S)} = c_{<j}\}$ and the third line is the empirical estimation of the second line. For CBM, the marginalized $P(E_j = 1)$ depends on the relevance probabilities of higher ranked results. But relevance is unknown during the CLTR and is yet to be learned. Instead of using EM algorithms to estimate relevance and marginalized examination probabilities, we propose to simply use the $P(E_j = 1 \mid C_{<j})$ which has been shown here to be empirically equivalent to the original loss.

Next we will derive separate formulas for $P(E_j = 1 \mid C_{<j})$ in DCM, DBN and CCM models.

DCM. In DCM, the user examines the results from top to bottom until she finds an attractive result, $P(E_{j+1} = 1 \mid E_j = 1, C_j = 0) = 1$. After each click, there is a position dependent chance that the user is not satisfied, $P(E_{j+1} = 1 \mid C_j = 1) = \lambda_j$. Therefore:

$$P_{\text{DCM}}(E_j = 1 \mid c_{<j}) = \prod_{i<j} (1 - c_i(1 - \lambda_i)) \tag{2.6}$$

DBN. In DBN, there is another binary variable to model the user's satisfaction after a click. A satisfied user abandons the session, $P(E_{i+1} = 1 \mid S_i = 1) = 0$. An unsatisfied user may also abandon the session with a constant probability γ. Finally, after a click, the satisfaction probability depends on the document, $P(S_i = 1 \mid C_i = 1) = s_{x_i}$. Thanks to Eq. (2.4), we only need the session specific examination probability, which can be derived as follows:

$$P_{\text{DBN}}(E_j = 1 \mid c_{<j}) = \prod_{i<j} \gamma \cdot (1 - c_i \cdot s_{x_i}) \tag{2.7}$$

CCM. The CCM is a generalization of DCM where continuing to examine the results before a click is not deterministic, $P(E_{j+1} = 1 \mid E_j = 1, C_j = 0) = \alpha_1$. The probability of continuing after a click is not position dependent, but relevance dependent, $P(E_{j+1} \mid C_j = 1) = \alpha_2(1 - R_i) + \alpha_3 R_i$. Similar to DCM we have:

$$P_{\text{CCM}}(E_j = 1 \mid c_{<j}) = \prod_{i<j} (\alpha_1 - c_i(\alpha_1 - \alpha_2(1 - R_i) - \alpha_3 R_i)). \tag{2.8}$$

Parameter estimation. In click model studies, parameter estimation is performed for each query over the sessions initiated by that query [29]: a low variance estimation requires a great number of sessions for each query. In CLTR studies, on the other hand, one uses features of query-document pairs in order to generalize well to tail queries [131]. We leave the feature-based parameter estimation of CBM as future work.

2.4 Experimental Setup

Dataset. We use the Yahoo! Webscope [25] dataset for LTR with synthetic clicks. Our methodology follows previous unbiased LTR papers [4, 58]. We use binary relevance, considering the two most relevant levels as $r = 1$. We randomly select 50 queries from the training set and train a LambdaMART model over them to act as the initial ranker. The documents of all the queries are ranked using this initial ranker and the top 20 documents are shown to the virtual user. We remove all the queries which have no relevant documents in their top 20 documents. Consequently, the train and test sets have 11,474 and 4,085 queries, respectively. User behavior is modeled by PBM or DCM with various parameter assignments (see below). Sessions with at least one click are kept in the training set.

The reported results use $4M$ clicks for training, where the performance of CLTR is converged.

Click simulation. We use PBM and DCM for generating click data. For PBM, we use the widely used reciprocal formula for the examination probability [4, 58] (see Eq. (2.3)):

$$P_{\text{PBM}}(E_j = 1) = \theta_j = \left(\frac{1}{j}\right)^{\eta}, \tag{2.9}$$

with $\eta \in \{0.5, 1, 2\}$.

For DCM, we use a similar formula for λ (see Eq. (2.6)):

$$P_{\text{DCM}}(E_{j+1} = 1 \mid C_j = 1) = \lambda_j = \beta \left(\frac{1}{j}\right)^{\eta}, \tag{2.10}$$

where β and η are tuning parameters. We use $\beta \in \{0.6, 1\}$ and $\eta \in \{0.5, 1, 2\}$.

In both PBM and DCM cases, we used a noise (i.e. click on examined non-relevant items) with probability 0.05.

Experimental protocol. To investigate the effectiveness of PBM-IPS as well as CM-IPS, we try to train a CLTR over different sets of simulated click logs as explained above. We use DLA [4] to learn the click propensities based on the PBM assumption and MLE [50] to estimate λ's for DCM. Similar to other works on CLTR, we evaluate the rankings using explicit relevance judgements in the test set. We use nDCG at 10 to compare the rankings. We also report full-information results where the true relevance labels are used for training, i.e., the highest possible performance (skyline).

LTR implementation. Different LTR algorithms have been used for CLTR, including SVMRank [58], neural networks (NNs) [4, 5], and LambdaMART [131]. The differences are minimal [4]. We follow [4] and model the score function by a DNN, with the loss being softmax cross entropy. We use three layers with sizes $\{512, 256, 128\}$ and elu activation; the last two layers use dropout with a dropping probability of 0.1. Based on [119] we use a propensity clipping constant of 100 to avoid exploding variance.

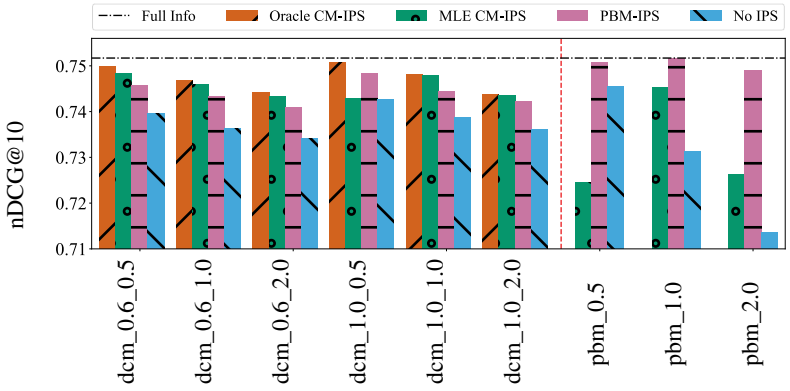


Figure 2.2: Performance of PBM-IPS and CM-IPS CLTR on different sets of simulated clicks. We repeated each experiment 15 times and report the mean value.

2.5 Results

CM-IPS effectiveness. In order to analyze the effectiveness of CM-IPS, we follow the protocol described in Section 2.4. Figure 2.2 shows the performance of CM-IPS compared to PBM-IPS CLTR on numerous simulated click sets.

The x-axis shows the method used for simulating clicks: either "dcm_β_η" or "pbm_η" as explained in Section 2.4. The y-axis is the ranking performance of CLTR methods in terms of nDCG at 10. We see both PBM-IPS and CM-IPS improve the biased naïve LTR (indicated by "No IPS") in almost all cases (except MLE CM-IPS on pbm_0.5). When using CM-IPS correction with oracle parameters for DCM simulated click sets (on the left of the vertical dashed line), the performance is consistently improved compared to PBM-IPS. All the differences are significant with $p < 0.0001$ except for dcm_1.0_2.0 which has $p < 0.05$. The reverse holds for PBM-IPS: it has a better performance for PBM simulated click sets (on the right of the vertical dashed line), compared to CM-IPS. In these datasets, the performance of PBM-IPS CLTR is very close to the full-information case. These observations suggest that for a great variety of different parameter settings of DCM, the PBM-IPS correction cannot remove the bias, while CM-IPS can. More generally, Figure 2.2 shows that when the click behavior and the correction method agree, the results are consistently better than the other case.

There is one practical issue that we leave as future work. This concerns the parameter estimation of DCM. The above discussions are valid when using the oracle parameter values for λ_j. It is worth mentioning that, unlike PBM, the parameters and the propensities are two different things in CM-IPS. The novelty of CM-IPS lies in computing the propensities given the parameters. We have tested maximum likelihood estimation (MLE) for estimating λ_j's [50]. Though the results with MLE CM-IPS are better than the PBM-IPS in most of the DCM simulated click sets, they are worse for dcm_1.0_0.5 and not significant for dcm_1.0_2.0 (Figure 2.2). As argued in [29], MLE for DCM is based on a simplifying assumption, which is not always true. Our

findings coincide with this fact. Therefore, there is a need for CLTR-based algorithms for parameter estimation for CBM (see Section 2.3).

Method selection. In order to choose between PBM- and CM-IPS for debiasing click logs, a measure that uses historical clicks to validate debiasing models is desired. For that, we use click log-likelihood. Click log-likelihood requires the click probabilities which are computed as the examination probability multiplied by the relevance probability. The examination probabilities are discussed in Section 2.3. For the relevance probabilities we use the output of our ranking function and pass it to different normalizing functions to have a valid probability range.

Our results on the sets presented previously in this section show the followings: (i) $\mathrm{softmax}$ always prefers CM-IPS (wrong selection for clicks close to PBM); (ii) $\mathrm{sigmoid}$ always prefers PBM-IPS (wrong selection for clicks close to CBM); and (iii) exponential $\mathrm{min\text{-}max}$ selects the better performing approach on the test set (correct selection in both cases). We leave more discussions in this regard as future work.

2.6 Conclusion

The position-based model (PBM) is the default assumption in IPS-based CLTR. However, it is unable to properly model the cascade behavior of users. We raised the question of the effectiveness of PBM in IPS-based unbiased CLTR when users click behavior tends to cascade-based models (CBM) (**RQ1.1**). Through a number of experiments, we have answered our **RQ1.1** negatively: PBM-IPS is not helpful in CBM situations and, in our tested cases, there is a gap between its performance and the full information case. This answer leads to a more important question: How to perform IPS correction for clicks close to CBM (**RQ1.2**). We provided CM-IPS, with closed-form formulas for three widely used CBMs, namely dependent click model (DCM), dynamic Bayesian network (DBN) and click chain model (CCM). We have shown the effectiveness of CM-IPS on the special case of DCM. Finally, we have given a short discussion about how to select between PBM- and CM-IPS only by looking at the clicks (and not using the true relevance labels).

In the next chapter, we will re-examine another assumption in IPS in terms of the relation between the click probability and relevance. We will show that in some settings, such as when the clicks suffer from trust bias, the assumption of a scaling transformation between the click probability and relevance does not hold. Accordingly, we will raise and address RQ2 in the following chapter.

3

Inverse Propensity Scoring Is Not Enough: Affine Correction for Trust Bias

In Chapter 2 we only considered position bias in clicks. Besides position bias, which has been well-studied, trust bias is another type of bias prevalent in user interactions with rankings: users are more likely to click incorrectly w.r.t. their preferences on highly ranked items because they trust the ranking system. This type of bias is the focus of RQ2 and will be addressed in this chapter. While previous work has observed this behavior in users, we prove that existing counterfactual learning to rank (CLTR) methods do not remove this bias, including methods specifically designed to mitigate this type of bias. Moreover, we prove that inverse propensity scoring (IPS) is principally unable to correct for trust bias under non-trivial circumstances. Our main contribution is a new estimator based on affine corrections: it both reweights clicks and penalizes items displayed on ranks with high trust bias. Our estimator is the first estimator that is proven to remove the effect of both trust bias and position bias. Furthermore, we show that our estimator is a generalization of the existing CLTR framework: if no trust bias is present, it reduces to the original IPS estimator. Our semi-synthetic experiments indicate that by removing the effect of trust bias in addition to position bias, CLTR can approximate the optimal ranking system even closer than previously possible.

3.1 Introduction

As we discussed in Chapter 2, learning from user interactions has difficulties in terms of noisy and biased implicit feedback. For instance, clicks are noisy in the sense that, often, a non-relevant item receives a click or a relevant item is skipped. The effect of noise is easily mitigated by averaging over a large number of clicks, but this is not true for bias. *Position bias*, a well-known type of bias of interactions through clicks [30], occurs because users are more likely to examine results at higher ranks. As a consequence, an item may receive more clicks because it was displayed at a high rank, not because it was preferred by the user. Other types of bias include *item-selection bias*: not all items can be displayed at once [96, 99]; *presentation bias*: items are presented in different

This chapter was published as: A. Vardasbi, H. Oosterhuis, and M. de Rijke. When inverse propensity scoring does not work: Affine corrections for unbiased learning to rank. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, pages 1475–1484, 2020.

manners [138]; and *trust-bias*: users are more likely to click incorrectly on higher ranked items [57]. In order to infer a user's true preferences from their interactions, the effects of these biases have to be corrected for.

Research into CLTR aims to find methods that learn from user interactions but whose optimization process is unaffected by biases [58]. Early CLTR methods correct for position bias using IPS [58, 130]. IPS estimators weight clicks inversely to the probability of the clicked items being examined during logging. Thus, clicks on items that are less likely to have been examined by users are weighted more heavily. This reweighting compensates for the effect of position bias, allowing CLTR methods to estimate and learn without being affected by position bias in expectation. Later CLTR work has focused on estimating examination probabilities [3, 4, 40, 131], training deep learning models [1], and correcting for more types of bias [2, 96, 99]. In particular, Agarwal et al. [2] have proposed an expansion to IPS to correct for both position bias and trust bias.

In this chapter, we address the following research question:

RQ2 How to effectively correct for trust bias in user click data?

First, we prove that *no IPS estimator is able to correct for trust bias*, under non-trivial circumstances. Since all existing bias mitigation methods are IPS-based approaches, this implies that there is currently no known CLTR method that can deal with trust bias. We identify the root cause to be the fact that IPS only corrects for missing-not-at-random (MNAR) feedback [58]. While position bias prevents clicks from occurring due to a lack of user examination, trust bias adds additional clicks due to user trust [2, 57]. Hence, clicks that are affected by trust bias are not simply a form of MNAR feedback and IPS cannot correct for such biases.

We introduce a novel estimator for CLTR that makes use of *affine* corrections, as opposed to the *linear* corrections of IPS. Our novel affine estimator both reweights clicks based on examination probabilities and penalizes items for being displayed on ranks where many incorrect clicks take place. We prove that the affine estimator is the first method that can correct for both position bias and trust bias. Furthermore, we show that it is an extension of the existing CLTR framework: when no trust bias is present the affine estimator naturally reduces to an IPS estimator. The results of our semi-synthetic experiments show that while existing CLTR methods are negatively affected by trust bias, our affine approach approximates the optimal ranking model under varying degrees of position bias and trust bias.

Answering RQ2, the main contributions of this chapter are:

1. The first CLTR estimator that is proven to be unbiased w.r.t. both position bias and trust bias.
2. A theoretical analysis that shows that IPS estimators cannot correct for trust bias.
3. An empirical analysis based on semi-synthetic experiments that reveal our affine estimator bridges the gap between existing CLTR methods and the optimal model when trust bias is present.

Table 3.1 summarizes the notation we use in this chapter.

Table 3.1: Notation used in this chapter.

Symbol	Description
q	a query
d	an item (to be ranked)
D_q	set of items to be ranked for query q
λ	metric function that assigns a weight per rank
f	a ranker, or ranking function, that scores items
y_i	a ranking displayed at interaction i
C	a click on an item
E	user examination of an item
R	the relevance of an item
$\tilde{R}$	the perceived relevance of an item
θ_k	examination probability at rank k: $P(E = 1 \mid k)$
$\gamma_{q,d}$	relevance probability: $P(R = 1 \mid q, d)$
ϵ_k^+	perceived relevance probability at rank k of an examined relevant item: $P(\tilde{R} = 1 \mid E = 1, R = 1, k)$
ϵ_k^-	perceived relevance probability at rank k of an examined non-relevant item: $P(\tilde{R} = 1 \mid E = 1, R = 0, k)$
α_k	first weight of the affine transformation of trust bias: $\alpha_k = \theta_k(\epsilon_k^+ - \epsilon_k^-)$
β_k	second weight of the affine transformation of trust bias: $\beta_k = \theta_k \epsilon_k^-$

3.2 Background and Related Work

This section covers supervised learning to rank (LTR) and the original IPS method for CLTR with position bias correction.

3.2.1 Learning to Rank

In general, the goal of LTR methods is to find the optimal ranking function f, in order to sort items for user-issued queries. For this work, we will use f to sort in ascending order. Let q indicate a query, d an item, and $\text{rank}(d \mid q, f)$ the rank of item d in the ranking produced by f for q. Then:

$$f(d_i \mid q) > f(d_j \mid q) \Rightarrow \text{rank}(d_i \mid q, f) \succ \text{rank}(d_j \mid q, f). \tag{3.1}$$

Commonly, f is considered optimal if it maximizes some linearly decomposable metric. Let $P(q)$ be the distribution of queries, D_q the set of items to be ranked for query q, and $P(R = 1 \mid q, d)$ the probability that an item d is considered relevant by the user. Then, with some weighting function λ, a linearly decomposable metric has the form:

$$\Delta(f) = \sum_q P(q) \sum_{d \in D_q} P(R = 1 \mid q, d) \cdot \lambda(d \mid q, f). \tag{3.2}$$

Generally, λ is based on the rank of d for q according to f. For instance, it can be chosen to match the well-known discounted cumulative gain (DCG) metric:

$$\lambda_{\text{DCG}}(d \mid q, f) = \Big(\log_2 \big(\text{rank}(d \mid D_q, q, f) + 1\big) \Big)^{-1}. \tag{3.3}$$

If the relevance probabilities $P(R = 1 \mid q, d)$ are known, finding the optimal f can be done through traditional supervised LTR methods [79].

3.2.2 Counterfactual LTR for Position Bias Correction

In practice the relevance probabilities $P(R = 1 \mid q, d)$ are not known and are costly to estimate through human labelling [25, 102]. Moreover, often the annotations obtained through manual labelling are not aligned with the actual preferences of the users [109].

As an alternative, CLTR methods use click logs to base their optimization and evaluation on. Clicks can be seen as a form of implicit feedback, which is indicative of the users' preferences but also a very noisy and biased signal. One of the most prevalent biases in clicks on items included in a ranking is *position bias*: users are less likely to examine – and therefore click – items on lower ranks. Position bias is formally modeled through the examination hypothesis, which states that a clicked item ($C \in \{0, 1\}$) must be examined ($E \in \{0, 1\}$) and considered relevant ($R \in \{0, 1\}$): $C = 1 \leftrightarrow E = 1 \wedge R = 1$. Position bias is often assumed to depend on the rank at which an item is displayed, while the relevance of an item is assumed to be independent of where it is displayed [30]. Thus, if k is the rank at which d is displayed, the probability of a click is:

$$P\big(C = 1 \mid q, d, k\big) = P\big(E = 1 \mid k\big) \cdot P\big(R = 1 \mid q, d\big). \tag{3.4}$$

The click probability (Eq. (3.4)) shows us that the position bias, modeled by $P(E = 1 \mid k)$, gives an unfair advantage to documents in positions that are more likely to be examined.

Let $\mathcal{D}$ be the set of logged interactions, containing N tuples each consisting of a user-issued query q_i, a displayed ranking y_i, and the observed clicks c_i where $c_i(d) \in \{0, 1\}$:

$$\mathcal{D} = \{(q_i, y_i, c_i)\}_{i=1}^{N}. \tag{3.5}$$

For brevity we will use the sum $\sum_{(d,k)\in y_i}$, which sums over the items d and their associated ranks in y_i:

$$\forall (d, k) \in y_i,\ k = \text{rank}(d \mid i). \tag{3.6}$$

Furthermore, we use $P(E = 1 \mid k) = \theta_k$. Thus, the probability of item d being examined at interaction i depends on the rank it was displayed at: $P(E = 1 \mid d, i) = \theta_{\text{rank}(d|i)}$. The first published CLTR methods correct for position bias using an IPS estimator [58, 130]. This IPS estimator weights each click inversely to the probability that the clicked item was examined:

$$\hat{\Delta}_{IPS}(f) = \frac{1}{N} \sum_{i=1}^{N} \sum_{(d,k)\in y_i} \frac{c_i(d)}{\theta_k} \cdot \lambda(d \mid q_i, f). \tag{3.7}$$

The result is an unbiased estimator, since in expectation it correctly estimates Δ:

$$\mathbb{E}_{q,y,c}[\hat{\Delta}_{IPS}(f)] = \Delta(f) \quad \text{(under the click model in Eq. (3.4))}. \tag{3.8}$$

For a proof of unbiasedness we refer to the work by Joachims et al. [58], who prove that even with click noise $\hat{\Delta}_{IPS}$ can be used for unbiased CLTR optimization. However, we note that this proof relies on (at least) three important assumptions: (i) the click model as described in Eq. (3.4) is true, (ii) the propensities θ are known, and (iii) all propensities are positive: $\forall k\ \theta_k > 0$.

A lot of related work has considered the estimation of the position bias parameters θ, using randomization [3, 40, 58, 131], or by jointly estimating relevance and position bias [4, 130]. Recently, both Ovaisi et al. [99] and Oosterhuis and de Rijke [96] have proposed using different propensities when not all items can be displayed at once (i.e., in case $\exists k\, \theta_k = 0$). For this chapter, we will assume that all propensities are positive and thus $\hat{\Delta}_{IPS}$ is unbiased.

Finally, different methods have been proposed to optimize f based on $\hat{\Delta}_{IPS}$. Joachims et al. [58] show that Rank-SVM [56] can be adapted to optimize IPS estimates for the average-relevant-position metric. Agarwal et al. [1] introduce a method that can optimize any differentiable model w.r.t. an IPS estimate of a metric based on a monotonically decreasing function. Lastly, Oosterhuis and de Rijke [96] show that the supervised LambdaLoss LTR framework [132] can easily be adapted to optimize IPS estimates as well.

3.3 Trust Bias

Besides position bias, other forms of bias are also known to affect user interactions with ranked lists. Joachims et al. [57] conclude that the trust users have in a ranking system affects their click behavior. Because users trust the results, they are more likely to perceive top-ranked items to be relevant, even when the displayed information about the item suggests otherwise. Similar to position bias, this causes items displayed at high ranks to have an unfair advantage, however, despite this similarity the effects of the two types of bias are not identical.

Recently, Agarwal et al. [2] have modeled trust bias by distinguishing between *perceived relevance* $\tilde{R} \in \{0, 1\}$ and *real relevance* R. Trust bias occurs because users are more likely to perceive items as relevant $\tilde{R} = 1$ if they are among the top ranked items in the list. In Agarwal et al.'s model, a click happens when a user examines and perceives an item to be relevant: $C = 1 \leftrightarrow E = 1 \wedge \tilde{R} = 1$. The model combines rank-based position bias (as described in Section 3.2.2) with trust bias, resulting in the following click probability:

$$P\big(C = 1 \mid q, d, k\big) = P\big(E = 1 \mid k\big) \cdot P\big(\tilde{R} = 1 \mid E = 1, R, k\big). \tag{3.9}$$

Furthermore, the probability of perceived relevance of an examined item is conditioned on the actual relevance and the rank k at which item d is displayed. For brevity, we use ϵ_k^+ and ϵ_k^- to denote these probabilities:

$$\begin{aligned} P\big(\tilde{R} = 1 \mid E = 1, R = 1, k\big) &= \epsilon_k^+, \\ P\big(\tilde{R} = 1 \mid E = 1, R = 0, k\big) &= \epsilon_k^-. \end{aligned} \tag{3.10}$$

Additionally, we write $\gamma_{q,d}$ for the probability of actual relevance: $\gamma_{q,d} = P(R = 1 \mid q, d)$. These conventions allow us to have the following succinct notation for the click probability:

$$P\big(C = 1 \mid q, d, k\big) = \theta_k \left(\epsilon_k^+ \gamma_{q,d} + \epsilon_k^- (1 - \gamma_{q,d})\right). \tag{3.11}$$

It is important to note that the combination of trust bias and position bias can be seen as an affine transformation between the relevance probabilities and click probabilities. If we choose $\alpha_k = \theta_k(\epsilon_k^+ - \epsilon_k^-)$ and $\beta_k = \theta_k \epsilon_k^-$, this affine transformation becomes apparent:

$$P\big(C = 1 \mid q, d, k\big) = \alpha_k P(R = 1 \mid q, d) + \beta_k. \tag{3.12}$$

We will use this property in Section 3.5 to introduce affine corrections for these biases.

An empirical analysis by Agarwal et al. [2] shows that their trust bias model better captures observed user behavior than the model that only considers position bias (Eq. (3.4)). Furthermore, Agarwal et al. propose an IPS estimator in order to correct for both trust bias and position bias. In the next section, we will first prove that this estimator cannot correct for these biases. Moreover, we subsequently prove that no IPS estimator is capable of doing so. Then, in Section 3.5 we introduce an estimator based on affine corrections, and prove that it is the first unbiased estimator that corrects for both position bias and trust bias.

3.4 Existing Methods and Trust Bias

In this section, we discuss Agarwal et al. [2]'s Bayes-IPS method designed specifically for trust bias. We prove that no IPS estimator is able to correct for trust bias, including Bayes-IPS.

3.4.1 Bayes-IPS

Agarwal et al. [2] have proposed the Bayes-IPS estimator to correct for trust bias and position bias. This estimator combines two corrections: (i) correcting for position bias by weighting inversely to θ; and (ii) correcting for trust bias by weighting each click to the probability of true relevance: $P(R = 1 \mid \tilde{R} = 1, E = 1, k)$. This results in the following estimator:

$$\hat{\Delta}_{Bayes\text{-}IPS}(f) = \frac{1}{N} \sum_{i=1}^{N} \sum_{(d,k) \in y_i} \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-} \frac{c_i(d)}{\theta_k} \cdot \lambda(d \mid q_i, f). \tag{3.13}$$

We note that $\hat{\Delta}_{Bayes\text{-}IPS}$ is still an IPS estimator; the difference with $\hat{\Delta}_{IPS}$ is that it uses the weights $\frac{1}{\theta_k} \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-}$ instead of $\frac{1}{\theta_k}$. In addition to θ_k, Bayes-IPS also needs to know the values of ϵ_k^+ and ϵ_k^-. Agarwal et al. use expectation maximization (EM) to estimate these values from click logs, and, using the estimated values, optimize a ranking model using $\hat{\Delta}_{Bayes\text{-}IPS}$. Their results show that optimizing with $\hat{\Delta}_{Bayes\text{-}IPS}$ is more effective than using $\hat{\Delta}_{IPS}$ and leads to significant improvements when ranking for search through emails or other personal documents [2].

While empirical results indicate that $\hat{\Delta}_{Bayes\text{-}IPS}$ is an improvement over $\hat{\Delta}_{IPS}$, neither estimator is unbiased w.r.t. trust bias. If trust bias is present, i.e., if $\exists k, k' \ (\epsilon_k^- \neq \epsilon_{k'}^-)$, then $\hat{\Delta}_{Bayes\text{-}IPS}$ is biased. We can show this by looking at the difference between $\hat{\Delta}_{Bayes\text{-}IPS}$ and Δ, which is not necessarily equal to zero. Let $\lambda_{q,d}$ be short for $\lambda(d \mid q, f)$, then:

$$\begin{aligned}
&\Delta(f) - \mathbb{E}_{q,y,c}\left[\hat{\Delta}_{Bayes\text{-}IPS}(f)\right] \\
&= \mathbb{E}_{q,y,c}\left[\sum_{(d,k)\in y_i}\left(\gamma_{q,d} - \frac{\epsilon_k^+}{\epsilon_k^+ + \epsilon_k^-}\frac{P(C=1 \mid q,d,k)}{\theta_k}\right)\cdot \lambda_{q,d}\right] \\
&= \mathbb{E}_{q,y,c}\left[\sum_{(d,k)\in y_i}\left(\left(1 - \frac{\epsilon_k^+\left(\epsilon_k^+ - \epsilon_k^-\right)}{\epsilon_k^+ + \epsilon_k^-}\right)\gamma_{q,d} - \left(\frac{\epsilon_k^+\epsilon_k^-}{\epsilon_k^+ + \epsilon_k^-}\right)\right)\cdot \lambda_{q,d}\right].
\end{aligned} \tag{3.14}$$

Clearly, it is non-trivial to derive under what conditions the difference between $\Delta(f)$ and $\mathbb{E}_{q,y,c}[\hat{\Delta}_{Bayes\text{-}IPS}(f)]$ is zero. Instead of further investigating Bayes-IPS, we will prove that no IPS estimator is unbiased w.r.t. trust bias under non-trivial circumstances, thereby also proving that no practical conditions exist where this difference is guaranteed to be zero.

3.4.2 IPS cannot Correct for Trust Bias

We proceed by considering whether any IPS estimator can be unbiased w.r.t. trust bias. Consider a generic IPS estimator $\hat{\Delta}_\rho$. We will derive the values the propensities ρ should have for unbiased CLTR:

$$\hat{\Delta}_\rho(f) = \frac{1}{N}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}\frac{c_i(d)}{\rho_{q_i,d,k}}\cdot\lambda(d \mid q_i, f). \tag{3.15}$$

Importantly, we have to limit the possible choices for ρ, because trivially unbiased estimators are theoretically possible [54]:

$$\forall d,k \ \rho_{q,d,k} = \frac{\frac{1}{N}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}\gamma_{d,k}\cdot\lambda(d \mid q_i, f)}{\frac{1}{N}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}c_i(d)\cdot\lambda(d \mid q_i, f)}. \tag{3.16}$$

To avoid such trivial situations, we use the following definition for circumstances where CLTR is not a trivial problem:

Definition 3.1. *We define* non-trivial circumstances *as situations where no information about the relevances γ is known. Furthermore, trust bias must be present, meaning users' trust must not be constant at all the ranks:*

$$\exists k, k' \ (\epsilon_k^- \neq \epsilon_{k'}^-). \tag{3.17}$$

Additionally, every displayed item should have a chance of being clicked and clicks at any rank k should be positively correlated with relevance:

$$\forall k \ (\theta_k(\epsilon_k^+ - \epsilon_k^-) > 0). \tag{3.18}$$

Lastly, the metric λ should not be indifferent to the ranking of f:

$$\exists q, d, f, f' \ (\lambda(d \mid q, f) \neq \lambda(d \mid q, f')). \tag{3.19}$$

With this definition we avoid the following scenarios: (i) ρ is chosen based on the known values of γ, in which case there is no need to estimate $\Delta(f)$ based on clicks; (ii) there is no trust bias, in which case every method is trivially unbiased w.r.t. trust bias; (iii) some items cannot receive clicks or clicks are not indicative of relevance, in these cases there is no signal to learn from; (iv) the metric is indifferent to the ranking function f, in which case there is nothing to evaluate since all ranking functions are equally good.

Naturally, an unbiased estimator should lead to the same optimal ranking as the full information case. For this, it is sufficient to have consistent pairwise rankings. To be clear about what we are going to prove about unbiasedness w.r.t. trust bias, we introduce the following formal definition:

Definition 3.2. *An IPS estimator $\hat{\Delta}_\rho$ is* unbiased w.r.t. trust bias*, if in all non-trivial circumstances ρ can be chosen so that it can correctly predict relative differences:*

$$\exists \rho, \forall f, f', \ \left(\Delta(f) > \Delta(f') \leftrightarrow \mathbb{E}_c\left[\hat{\Delta}_\rho(f)\right] > \mathbb{E}_c\left[\hat{\Delta}_\rho(f')\right]\right). \tag{3.20}$$

In other words, we define an estimator to be unbiased w.r.t. to trust bias, if it can unbiasedly predict the preference between any two rankers under any non-trivial circumstances. Again, it is important to avoid ρ being chosen based on knowledge of γ. If a $\hat{\Delta}_\rho$ meets our definition of unbiasedness it can safely be applied in any non-trivial circumstances; we argue that this covers all realistic CLTR situations.

Theorem 3.1. *No IPS estimator is unbiased w.r.t. trust bias.*

Proof. We will prove this by showing that there are non-trivial circumstances where no values of ρ exist for $\hat{\Delta}_\rho$ to correctly predict relative differences. We do so by starting from the most basic ranking example and deriving the values of ρ where $\hat{\Delta}_\rho$ is unbiased, we prove that no such values exist. In addition to this proof, we will show how our basic example can be extended to include rankings with more queries and items.

In our basic example, we consider a system that only receives a single query q_1: $P(q_1) = 1$ and that only has to rank two documents $D_{q_1} = \{d_1, d_2\}$. Therefore, two ranking functions can cover all possible rankings: f_1 that produces $[d_1, d_2]$ and f_2 that produces $[d_2, d_1]$. Lastly, the metric we consider is a top-1 metric, which means it is only affected by the top document of a ranking:

$$\begin{aligned} \lambda(d_1 \mid q_1, f_1) = \lambda(d_2 \mid q_1, f_2) > 0, \\ \lambda(d_2 \mid q_1, f_1) = \lambda(d_1 \mid q_1, f_2) = 0. \end{aligned} \tag{3.21}$$

Thus, in this basic example, we are trying to estimate whether d_1 should be ranked higher than d_2 or vice-versa.

The true difference in metric value between the rankers is:

$$\Delta(f_1) - \Delta(f_2) = (\gamma_{q_1,d_1} - \gamma_{q_1,d_2}) \cdot \lambda(d_1 \mid q_1, f_1). \tag{3.22}$$

Therefore, only the difference in item relevance matters for the relative difference:

$$\operatorname{sign}\big(\Delta(f_1) - \Delta(f_2)\big) = \operatorname{sign}(\gamma_{q_1,d_1} - \gamma_{q_1,d_2}). \tag{3.23}$$

The estimates of $\hat{\Delta}_\rho$ are based on N interactions with query q_1 where at each interaction d_1 and d_2 were displayed at rank 1 and 2, respectively. The difference in the expected estimates (cf. Eq. (3.11) and Eq. (3.15)) is therefore:

$$\begin{aligned}&\mathbb{E}_c\left[\hat{\Delta}_\rho(f_1)\right] - \mathbb{E}_c\left[\hat{\Delta}_\rho(f_2)\right] \\ &\quad= \frac{\theta_1\left(\left(\epsilon_1^+ - \epsilon_1^-\right)\gamma_{q_1,d_1} + \epsilon_1^-\right)}{\rho_{q_1,d_1,1}} - \frac{\theta_2\left(\left(\epsilon_2^+ - \epsilon_2^-\right)\gamma_{q_1,d_2} + \epsilon_2^-\right)}{\rho_{q_1,d_2,2}}.\end{aligned} \tag{3.24}$$

We note that this scenario falls under the definition of a non-trivial circumstance (Definition 3.1).

In order to be unbiased, the values of the propensities ρ must be chosen so that the requirement in Eq. (3.20) is met. Note that for two continuous functions to always have the same sign, they should agree on zero values. By combining Eq. (3.20) with Eq. (3.23) and Eq. (3.24), we can derive that ρ must meet the following requirement:

$$\gamma_{q_1,d_1} = \gamma_{q_1,d_2} \leftrightarrow \frac{\rho_{q_1,d_1,1}}{\rho_{q_1,d_2,2}} = \frac{\theta_1\left(\left(\epsilon_1^+ - \epsilon_1^-\right)\gamma_{q_1,d_1} + \epsilon_1^-\right)}{\theta_2\left(\left(\epsilon_2^+ - \epsilon_2^-\right)\gamma_{q_1,d_2} + \epsilon_2^-\right)}. \tag{3.25}$$

Under non-trivial circumstances, ρ has to be chosen without knowledge of γ, therefore we must find a single value for each of $\rho_{q_1,d_1,1}$ and $\rho_{q_1,d_2,2}$ that meets this requirement for all possible values of γ. Combining this fact with the fact that γ consists of probabilities, we can derive the following requirement from Eq. (3.25):

$$\forall x \in [0,1]\ \left(\frac{\rho_{q_1,d_1,1}}{\rho_{q_1,d_2,2}} = \frac{\theta_1\left(\left(\epsilon_1^+ - \epsilon_1^-\right)x + \epsilon_1^-\right)}{\theta_2\left(\left(\epsilon_2^+ - \epsilon_2^-\right)x + \epsilon_2^-\right)}\right). \tag{3.26}$$

From this we can directly derive the following requirement for the bias parameters ϵ:

$$\forall x \in [0,1]\ \left(\frac{\epsilon_1^+}{\epsilon_2^+} = \frac{\epsilon_1^-}{\epsilon_2^-} = \frac{\left(\epsilon_1^+ - \epsilon_1^-\right)x + \epsilon_1^-}{\left(\epsilon_2^+ - \epsilon_2^-\right)x + \epsilon_2^-}\right). \tag{3.27}$$

Thus, a solution for the propensities ρ in Eq. (3.26) only exists if the trust bias parameters ϵ meet the requirement in Eq. (3.27). Solving for ϵ shows that the latter requirement can be simplified to:

$$\frac{\epsilon_1^+}{\epsilon_2^+} = \frac{\epsilon_1^-}{\epsilon_2^-}. \tag{3.28}$$

Therefore, only in very specific cases where trust bias adheres to Eq. (3.28) do values of ρ exist that can meet Eq. (3.25). This proves the theorem since we have provided input cases where no IPS is unbiased. In fact, in non-trivial circumstances (Definition 3.1 and Eq. (3.17)), the probability of even being close to this regularity of Eq. (3.28) is so low that in practice we can safely say that it never happens. So, not only do non-trivial input

cases exist where no IPS can be unbiased, but almost all of the time we are dealing with such cases.

Therefore, we have proven that $\hat{\Delta}_\rho$ can never correctly predict the relative difference between f_1 and f_2 in this example under non-trivial circumstances. In conclusion, we have therefore proven that no IPS estimator is unbiased w.r.t. trust bias, since there are examples where under non-trivial circumstances, no propensities ρ can be chosen so that it unbiasedly infers the preference between two rankers. □

While the basic counterexample used in the proof of Theorem 3.1 is enough for proving that IPS estimators are biased w.r.t. trust bias, we note that it can easily be extended to cases with more queries and items. For any number of queries and items and any item pair d_3 and d_4, there always exist two rankers f_3 and f_4 that agree on all item placements expect that they swap the ranks of d_3 and d_4. Using a proof analogous to the above, one can prove that similar to Eq. (3.28) $(\epsilon^+_{k_3}/\epsilon^+_{k_4}) = (\epsilon^-_{k_3}/\epsilon^-_{k_4})$, where k_3 and k_4 are the display ranks of d_3 and d_4, respectively. This process can be repeated for other item pairs until the requirement $\forall k, k'\ ((\epsilon^+_k/\epsilon^+_{k'}) = (\epsilon^-_k/\epsilon^-_{k'}))$ is obtained. Thus, one can prove this very restrictive requirement to the trust bias, that applies regardless of the number of queries and documents. Only when the trust bias adheres to this requirement, is it possible that an IPS estimator may be able to correctly infer relative differences. This shows that IPS is not a practical solution to trust bias.

In summary, we have proven that no IPS estimator is unbiased w.r.t. trust bias without *a priori* knowledge of the relevance γ, and thus is not applicable in any practical circumstances. We have done so by taking a generic IPS estimator and deriving the possible values for the propensities ρ that would lead to unbiased results in the most basic ranking scenario. The proof of Theorem 3.1 shows that for most instances of trust bias such values do not exist. Thus, none of the existing IPS estimators can correct for trust bias or can be adapted to do so. For clarity, this includes: the original CLTR estimators [58, 130]; the dual learning algorithm by Ai et al. [4]; the IPS with corrections for item-selection bias by Ovaisi et al. [99]; the policy aware estimator [96]; and the Bayes-IPS estimator [2].

The problem with IPS appears to be that trust bias causes an affine transformation between relevance probabilities and click probabilities. For a single query item pair q, d displayed at rank k, ideally a propensity $\rho_{q,d,k}$ exists so that:

$$\gamma_{q,d} = \frac{\alpha_k \gamma_{q,d} + \beta_k}{\rho_{q,d,k}}. \tag{3.29}$$

Such a propensity does exist but it is dependent on γ:

$$\rho_{q,d,k} = \frac{\alpha_k \gamma_{q,d} + \beta_k}{\gamma_{q,d}}. \tag{3.30}$$

If $\beta_k = 0$ (i.e., $\epsilon^-_k = 0$), the transformation becomes linear and ρ becomes independent of γ: $\rho_{q,d,k} = \alpha_k$. Thus, the core issue is that IPS applies a linear transformation to observed clicks but a linear transformation cannot correct for the affine transformation caused by trust bias. As a solution to this problem, we will introduce a novel estimator that applies affine corrections to clicks.

3.5 Affine Corrections for Trust Bias

Next, we introduce our novel affine estimator: the first method that is proven to correct for trust bias. We also compare the affine estimator with the existing IPS estimator, and introduce an adaption of the EM algorithm for estimating trust bias.

3.5.1 A Novel Affine Estimator

In Section 3.3 we described how trust bias can be seen as an affine transformation from relevance probabilities to click probabilities (see Eq. (3.12)). Subsequently, in Section 3.4.2 we proved that IPS estimators cannot correct for trust bias because IPS can only apply linear transformations and no linear transformation can reverse the effect of an affine transformation (in non-trivial circumstances).

We now propose a novel estimator based on affine transformations to correct for both position bias and trust bias: the affine estimator. The estimator works for any situation where click probabilities are based on an affine transformation of relevance probabilities:

$$P\big(C = 1 \mid q, d, k\big) = \alpha_k P(R = 1 \mid q, d) + \beta_k. \tag{3.31}$$

This includes trust bias where $\alpha_k = \theta_k(\epsilon_k^+ - \epsilon_k^-)$ and $\beta_k = \theta_k \epsilon_k^-$. Spelled out in the notation of Eq. (3.31) and in the trust bias notation, the affine estimator is:

$$\begin{aligned}\hat{\Delta}_{\text{affine}}(f) &= \frac{1}{N}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}\frac{c_i(d) - \beta_k}{\alpha_k}\cdot\lambda(d \mid q_i, f) \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}\frac{c_i(d) - \theta_k\epsilon_k^-}{\theta_k(\epsilon_k^+ - \epsilon_k^-)}\cdot\lambda(d \mid q_i, f).\end{aligned} \tag{3.32}$$

We see that the affine estimator reweights clicks inversely to α_k, which is somewhat similar to IPS. However, the salient difference is that $\hat{\Delta}_{\text{affine}}$ also penalizes items by subtracting $\frac{\beta_k}{\alpha_k}$. This penalty compensates for *incorrect* clicks where the perceived relevance does not match the true relevance: $\tilde{R} = 1 \wedge R = 0$. Thus items displayed at ranks where more *incorrect* clicks take place receive more penalties, while simultaneously clicks are reweighted according to the position bias θ_k and to compensate for the penalties: $\epsilon_k^+ - \epsilon_k^-$. We note that unlike with the IPS estimator, an item that is displayed but not clicked can receive a negative weight. In expectation later clicks will compensate for this effect.

Theorem 3.2. *The affine estimator is unbiased w.r.t trust bias.*

Proof. First, we use the assumption that clicks are correlated with relevancy: $\forall k\ (\alpha_k \neq 0)$. Then we consider the expected value for a single click $c_i(d)$:

$$\mathbb{E}_c\left[\frac{c_i(d) - \beta_k}{\alpha_k}\right] = \frac{(\alpha_k \cdot \gamma_{q,d} + \beta_k) - \beta_k}{\alpha_k} = \gamma_{q,d}. \tag{3.33}$$

We can use this to derive the expected value of the affine estimator; it is equal to the true metric value:

$$\begin{aligned}
&\mathbb{E}_{q,y,c}\left[\hat{\Delta}_{\text{affine}}(f)\right] \\
&\quad = \mathbb{E}_{q,y}\left[\sum_{(d,k)\in y_i} \mathbb{E}_c\left[\frac{c_i(d)-\beta_k}{\alpha_k}\right]\cdot\lambda(d\mid q,f)\right] \\
&\quad = \mathbb{E}_{q,y}\left[\sum_{(d,k)\in y_i} \gamma_{q,d}\cdot\lambda(d\mid q,f)\right] = \Delta(f).
\end{aligned} \tag{3.34}$$

Therefore, the affine estimator is unbiased in expectation. □

The negative penalties (β_k/α_k) may be counter-intuitive. For a better understanding we consider a maximally non-relevant item $\gamma_{q,d} = 0$ that is displayed at rank k, M times. We expect to observe $M \cdot \beta_k$ clicks (all incorrect since $\gamma_{q,d} = 0$). The sum of the penalties for the item given by the affine estimator is $M \cdot (\beta_k/\alpha_k)$, while each click is weighted by $1/\alpha_k$. Thus, if we sum the weights for the clicks we expect $M \cdot (\beta_k/\alpha_k)$, therefore taking this sum minus the penalties correctly results in a zero weight for the item (in expectation). As with any estimator, for reliable estimates, M needs to be considerably large due to variance.

This concludes the introduction of our novel affine estimator. By performing affine transformations to clicks it is the first estimator that can correct for the effect of both position bias and trust bias.

3.5.2 Relation with IPS and other Properties

While the affine estimator is very distinct from IPS since it can perform corrections that IPS cannot, we consider the former to be an extension of the latter. In the most straightforward way any IPS-based estimator can be seen as a special case of the affine estimator where $\forall k\ (\beta_k = 0)$. More generally, we consider the situation without trust bias, i.e., where ϵ_k^+ and ϵ_k^- have the same value for every k: $\forall k, k'\ (\epsilon_k^- = \epsilon_{k'}^- = \epsilon^- \wedge \epsilon_k^+ = \epsilon_{k'}^+ = \epsilon^+)$ and where clicks are positively correlated with relevance: $\epsilon^+ > \epsilon^-$. Also, we will assume that summing λ over documents leads to a constant value:

$$\forall f, f', q\ \left(\sum_{d\in D_q}\lambda(d\mid q,f) = \sum_{d\in D_q}\lambda(d\mid q,f')\right). \tag{3.35}$$

This means that if all items are equally relevant the order of the items does not matter. We note that this holds for virtually all ranking metrics, e.g., DCG, precision, recall,

MAP, ARP, etc. Now, Eq. (3.32) can be rewritten as follows:

$$\begin{aligned}
&\hat{\Delta}_{\text{affine}}(f) \\
&= \frac{1}{N}\frac{1}{\epsilon^+ - \epsilon^-}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}\left(\frac{c_i(d)}{\theta_k} - \epsilon^-\right)\cdot\lambda(d \mid q_i, f) \\
&= \frac{1}{\epsilon^+ - \epsilon^-}\hat{\Delta}_{IPS}(f) - \frac{1}{N}\frac{\epsilon^-}{\epsilon^+ - \epsilon^-}\sum_{i=1}^{N}\sum_{(d,k)\in y_i}\lambda(d \mid q_i, f) \\
&= \frac{1}{\epsilon^+ - \epsilon^-}\hat{\Delta}_{IPS}(f) - \mathbb{C},
\end{aligned} \tag{3.36}$$

where $\mathbb{C}$ is a constant independent of f. Therefore, $\hat{\Delta}_{\text{affine}}$ unbiasedly predicts relative differences w.r.t. $\hat{\Delta}_{IPS}$:

$$\forall f, f',\ \hat{\Delta}_{\text{affine}}(f) > \hat{\Delta}_{\text{affine}}(f') \leftrightarrow \hat{\Delta}_{IPS}(f) > \hat{\Delta}_{IPS}(f'). \tag{3.37}$$

Consequently, we can conclude that optimizing f w.r.t. $\hat{\Delta}_{\text{affine}}(f)$ also optimizes w.r.t. $\hat{\Delta}_{IPS}$ when trust bias is not present. This further shows that the affine estimator should be viewed as a generalization of the existing IPS approach.

Furthermore, the notation of the affine estimator in Eq. (3.32) also reveals some other intuitive properties. We see that if for some k, $\alpha_k = 0$, then the estimator becomes undefined, thus if clicks are not correlated with relevance, the estimator cannot be applied. Interestingly, if we compare this with the trust bias model we see that there are only two cases when $\exists k\,(\alpha_k = 0)$ can occur: (i) when $\exists k\,(\theta_k = 0)$, i.e., at some rank k some items cannot be observed or clicked, hence nothing about the item at this rank can be learned; or (ii) when $\exists k\,(\epsilon_k^+ = \epsilon_k^-)$, i.e., at some rank k non-relevant and relevant items are equally likely to be clicked, thus there is nothing to learn from the click signal. Furthermore, something interesting happens if $\exists k\,(\epsilon_k^+ < \epsilon_k^-)$, i.e., if at some rank k *non-relevant* items are *more* likely to be clicked than *relevant* items. In this case, non-clicked items receive a positive penalty and clicks lead to negative scores, meaning the less clicked items are preferred since they are more likely to be relevant. All these cases are very intuitive and we consider it a great strength that they can be inferred from the affine estimator from just a brief analysis of its formulation.

3.5.3 Parameter Estimation

Agarwal et al. [2] describe how EM can be used to estimate the position bias and trust bias parameters. We also use the regression-based EM procedure for estimating the bias parameters. However, unlike Agarwal et al., who estimate three parameters per rank k, namely θ_k, ϵ_k^- and ϵ_k^+, we notice that only two have to be estimated:

$$\begin{aligned}
\zeta_k^+ &= P(C = 1 \mid R = 1, k) = \theta_k\epsilon_k^+ = \alpha_k + \beta_k \\
\zeta_k^- &= P(C = 1 \mid R = 0, k) = \theta_k\epsilon_k^- = \beta_k.
\end{aligned} \tag{3.38}$$

From these two parameters the value of α_k and β_k can be inferred directly ($\alpha_k = \zeta_k^+ - \zeta_k^-$), and these are the only parameters required for the trust bias click model (Eq. (3.12)) and the affine estimator (Eq. (3.32)).

To estimate these parameters we adapt the Expectation step, where the parameters are updated as follows:

$$\zeta_k^+ = \frac{\sum_{i=1}^{N} c_i(d) P(R=1|C=1,q_i,d,k)}{\sum_{i=1}^{N} c_i(d) P(R=1|C=1,...) + (1-c_i(d)) P(R=1|C=0,...)}, \tag{3.39}$$

and

$$\zeta_k^- = \frac{\sum_{i=1}^{N} c_i(d) P(R=0 \mid C=1,q_i,d,k)}{\sum_{i=1}^{N} c_i(d) P(R=0|C=1,...) + (1-c_i(d)) P(R=0|C=0,...)}, \tag{3.40}$$

where the conditional relevance probabilities $P(R \mid C,q,d,k)$ are computed using Bayes's law. This simplification allows us to estimate the parameters with less computational costs. And since fewer parameters are estimated, we expect EM to converge faster.

In the Maximization step, the γ values are estimated by a regression algorithm. We use ζ^- and ζ^+ obtained from the E-step to train the unbiased ranking function f based on $\hat{\Delta}_{\text{affine}}$. Previous work [2, 131] suggests to use a *sigmoid* as a final activation function to obtain valid probability values. However, we observed that the sigmoid function gives very similar relevance probabilities between most items. In contrast, the *softmax* function results in more varied values but it forces the probabilities to sum to one for each query. As a simple alternative we propose the *soft-min-max* function, which does not force probabilities to sum to one, but still results in varied values:

$$\text{soft-min-max}(x_i) = \frac{e^{x_i} - e^{\min(x_i)}}{e^{\max(x_i)} - e^{\min(x_i)}}. \tag{3.41}$$

Our experiments show that the choice of activation function leads to noticeable differences.

3.6 Experimental Setup

We follow the semi-synthetic setup that is prevalent in existing CLTR work [4, 55, 58, 96], where queries, documents, and relevance judgments are sampled from supervised LTR datasets, while clicks are simulated using probabilistic user models. First, we train a production ranker for each dataset; we randomly select 20 queries from each training set and use the supervised LTR LambdaMART method to optimize a ranking model. With these production rankers, we simulate a situation where a decent ranking system exists but still leaves plenty of room for improvement. On each dataset, we simulate user interactions by repeatedly: (i) uniform-random sampling a query from the training set, (ii) ranking the documents for that query with the production ranker, and (iii) simulating clicks on the resulting ranking using a probabilistic user model. This semi-synthetic setup allows us to vary the number of clicks available for learning, as well as the position bias and trust bias of the simulated user. Thus, we can analyze the effects these factors have on the affine estimator and other CLTR methods.

3.6.1 Datasets

We use two of the largest publicly available LTR datasets: Yahoo! Webscope [25] and MSLR-WEB30k [102]. Both were created by a commercial search engine, and each contains around 30 000 queries, each query has a set of preselected documents to be ranked. The datasets contain five level relevancy tags acquired through expert labeling for the preselected query-document pairs. Yahoo! has 24 documents per query on average and uses 700-feature vectors to represent query-documents; MSLR has 125 per query and uses 136 features. Each dataset is split into training, validation, and test sets; we only use the first fold of MSLR.

3.6.2 Click Simulation

Clicks are simulated on rankings produced by the production rankers by applying probabilistic click models.

Per the experimental setting, we simulate up to $8 \cdot 10^6$ clicks on the training set. The number of validation clicks is always 15% and 33% of the training clicks for Yahoo! and MSLR, respectively. These numbers were chosen to match the ratio between the number of training and validation queries in each dataset.

We apply Agarwal et al. [2]'s trust bias model with varying parameters (see Section 3.3). The relevances $\gamma_{q,d}$ are based on the relevance label recorded in the datasets; we follow Joachims et al. [58] and use binary relevance:

$$P(R = 1 \mid q, d) = \gamma_{q,d} = \begin{cases} 1 & \text{if relevance\_label}(q, d) > 2, \\ 0 & \text{otherwise.} \end{cases} \tag{3.42}$$

Similar to previous work [55, 58, 96], we set the position bias inversely proportional to the display rank:

$$P(E = 1 \mid k) = \theta_k = \left(\frac{1}{\min(k, 20)} \right)^{\eta}, \tag{3.43}$$

where we vary the η parameter: $\eta \in \{1, 2\}$.

To the best of our knowledge, this is the first CLTR that simulates trust bias, thus there is no precedent for the values of ϵ_k^+ and ϵ_k^-. In order to simulate trust bias as realistically as possible, we base our values on the empirical work of Agarwal et al. [2]. It appears that the bias Agarwal et al. inferred from actual user interactions can be approximated by the following formula:

$$\forall k \in \{1, 2, \ldots, 5\}, \quad \epsilon_k^+ \approx 1 - \frac{k+1}{100} \quad \wedge \quad \epsilon_k^- \approx \epsilon_1^- \frac{1}{k}. \tag{3.44}$$

Unfortunately, Agarwal et al. only observed interactions on top-5 rankings. To prevent ϵ_k^+ and ϵ_k^- from disappearing on ranks beyond $k = 5$, we apply the following

$$\epsilon_k^+ = 1 - \frac{\min(k, 20) + 1}{100}, \quad \epsilon_k^- = \epsilon_1^- \frac{1}{\min(k, 10)}. \tag{3.45}$$

We use the *incorrect-click* rate on the first rank: ϵ_1^-, as a hyper-parameter to vary the amount of trust bias. We found that our results are consistent across different

values for ϵ_1^-. To cover both cases with high and low trust bias, we report results with $\epsilon_1^- \in \{0.65, 0.35\}$.

3.6.3 LTR Algorithm

Similar to Ai et al. [4] and Agarwal et al. [1] we train neural networks for our ranking functions. Our preliminary results indicate that the configuration of the networks does not have to be fine-tuned. The reported results are produced using models with three hidden layers with sizes $[512, 256, 128]$ respectively. All layers use elu activations and 0.1 dropout was applied to the last two layers.

For the loss function we follow Oosterhuis and de Rijke [96] and use LambdaLoss to optimize DCG [132]. For updating the gradients, we use the AdaGrad optimizer [33] with a learning rate of 0.004 and 0.02 for Yahoo! and MSLR datasets respectively, for 32 epochs.

3.6.4 Experimental Runs

We evaluate the performance of our affine estimator, by comparing the nDCG@10 of the models it produces with those produced using other estimators. The following estimators are used as baselines:

1. **No Correction**: The naïve estimator where each click is treated as an unbiased relevance signal.
2. **IPS**: The original CLTR IPS estimator [58, 131] that only corrects for position bias (see Section 3.2.2).
3. **Bayes-IPS**: The only existing CLTR estimator [2] designed for addressing trust bias (see Section 3.4).

For a clearer analysis, we also report the performance of the following ranking models:

4. **Production**: The production ranker used in during the logging of simulated clicks.
5. **Full Info**: A model trained using supervised LTR on the true relevance probabilities, its performance illustrates the (theoretical) maximal performance possible on a dataset. We note that this is not a baseline as it does not learn from clicks but (unrealistically) from the true relevances.

All reported nDCG@10 results are an average of four independent runs. Our experiments cover both the situation where the bias (θ, ϵ^- and ϵ^+) is known, e.g., through previous experiments [3, 40, 131], and the situation where the bias has to be estimated still.

3.7 Results and Discussion

This section discusses our experimental results. We consider the ranking performance of the affine estimator compared to other estimators, in both the situation where the exact bias is known and where it has to be estimated.

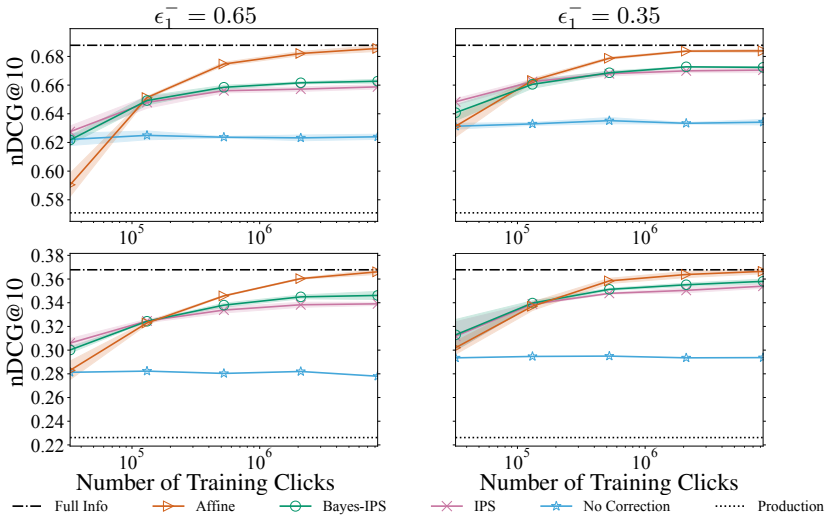


Figure 3.1: Comparison of different CLTR estimators in term of nDCG@10 on different numbers of clicks, under normal position bias ($\eta = 1$), and varying levels of trust bias. Estimators were given the true bias parameters. Results are averaged over four runs; shaded area indicates the standard deviation. Top row: Yahoo! Webscope dataset; bottom row: MSLR-WEB30k dataset.

3.7.1 Optimization with the Affine Estimator

First we consider whether *optimizing with the affine estimator leads to better performing ranking models than with existing estimators*.

Figures 3.1 and 3.2 show the performance (nDCG@10) reached by the different estimators under varying degrees of bias and different numbers of clicks available for training. We see that the naïve estimator has already converged after $3 \cdot 10^5$ clicks, since additional clicks do not increase its performance. In line with the empirical results of Agarwal et al. [2], we see that both IPS and Bayes-IPS improve over the naïve estimator, and that Bayes-IPS consistently outperforms IPS. However, when we compare with the Full Info ranker, we see that there is still a sizable gap between Full Info and Bayes-IPS in every tested setting on both datasets. In other words, neither IPS nor Bayes-IPS can approximate the optimal model under the tested degrees of trust bias. As predicted by the theory in Section 3.4, it thus appears that both these IPS estimators are biased w.r.t. trust bias.

In contrast, we see that the affine estimator does approximate the optimal model when position bias is mild ($\eta = 1$). However, under extreme position bias ($\eta = 2$) it has not reached convergence in any of our graphs. Based on the theory in Section 3.5.1, we expect convergence near the optimal model if it were given more training clicks. Furthermore, in all tested settings we observe the affine estimator to outperform the other estimators when more than 10^6 training clicks are available. Using the Student's t-test we found that all the improvements at $8 \cdot 10^6$ clicks are significant with $p \leq 0.001$,

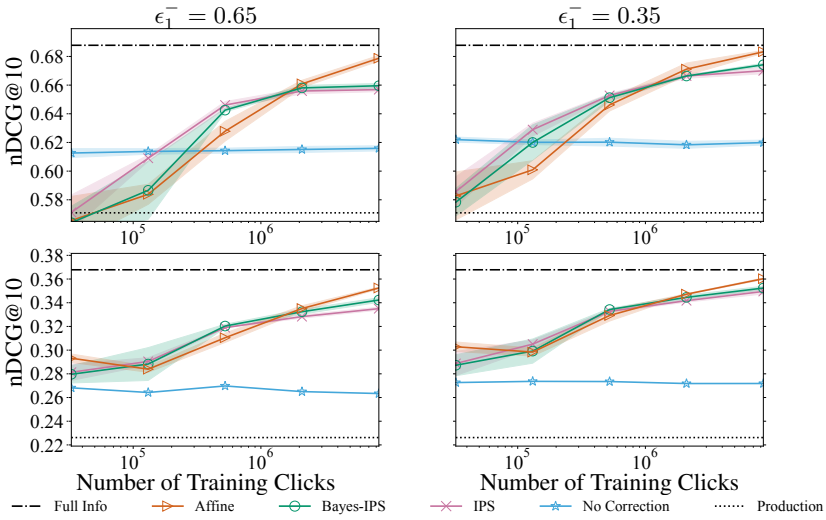


Figure 3.2: Comparison of different CLTR estimators in term of nDCG@10 on different numbers of clicks, under severe position bias ($\eta = 2$), and varying levels of trust bias. Estimators were given the true bias parameters. Results are averaged over four runs; shaded area indicates the standard deviation. Top row: Yahoo! Webscope dataset; bottom row: MSLR-WEB30k dataset.

except for the results on MSLR-WEB30k with $\eta = 1$ and $\epsilon_1^- = 0.35$ with a significance of $p \leq 0.002$. On small numbers of training clicks, the affine estimator has a similar or slightly lower performance than the other estimators. This could be explained by the bias-variance tradeoff: the Bayes-IPS and IPS estimators could have lower variance due to their bias, making them perform better on small amounts of data. Potentially, using propensity clipping on the affine estimator can increase its performance here [119].

In conclusion, our results strongly indicate that optimizing with the affine estimator results in better performing ranking models than with previously proposed estimators. In particular, on both datasets we see that, given enough click data, the affine estimator can be used to approximate the optimal ranking model, in settings with high or low degrees of trust bias or position bias.

3.7.2 Optimization with Estimated Biases

Next, we consider whether *optimization with the affine estimator is robust to estimated bias values.* This is important as in practice the values of bias parameters have to be estimated as well. While the theory proves that the affine estimator is unbiased when provided with the true bias values, we will now investigate whether it is still effective when they are estimated.

Figure 3.3 shows the performance (nDCG@10) reached by the affine estimator using bias parameters estimated from clicks (see Section 3.5.3), under varying degrees of position and trust bias. For clarity, both the ranking model optimization and the bias parameter estimation used the same clicks. Furthermore, the results in Figure 3.3 are

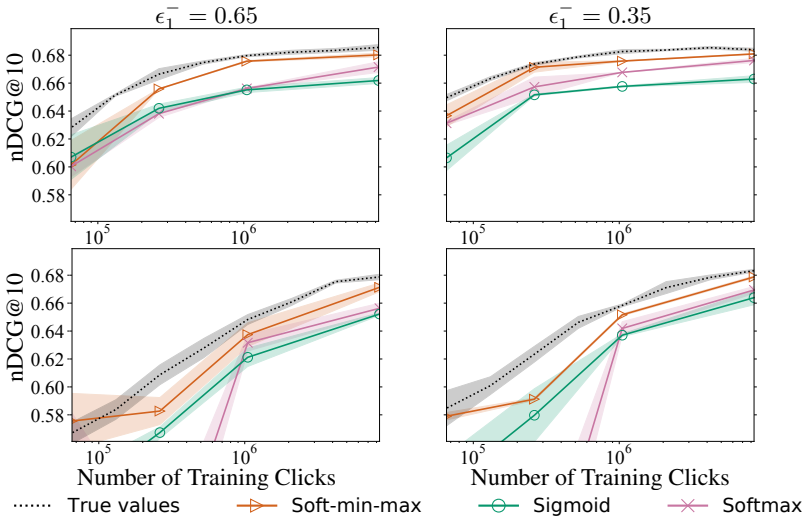


Figure 3.3: Comparison of different final activation functions to estimate the bias parameters, under varying levels of position and trust bias. Y-axis indicates the performance of ranking models optimized using the affine estimator. Results are averaged over four runs; shaded area indicates the standard deviation. All results are based on the Yahoo! Webscope dataset. Top row: normal position bias ($\eta = 1$); bottom row: severe position bias ($\eta = 2$).

separated for different final activation functions.

In Figure 3.3 we see that parameter estimation with the soft-min-max function leads to the best performance: soft-min-max outperforms the other functions in all settings, regardless of the number of training clicks. Though the difference between soft-min-max and optimization with the true bias values is noticeable, it appears to be a small difference, especially after 10^6 clicks. This suggests that the affine estimator with the soft-min-max function is robust to estimated bias values. Additionally, we see that the softmax function leads to decent performance when many clicks are available, but handles small numbers of clicks less well. Lastly, the sigmoid function results in the poorest performance.

Figures 3.4 and 3.5 show the estimated parameters after $8 \cdot 10^6$ clicks in the same settings. Interestingly, these figures show that none of the functions leads to extremely accurate bias estimation with EM. We see that except for the first position, Soft-min-max and sigmoid underestimate the values of α_k, while softmax overestimates it. While soft-min-max and softmax have accurate estimates of β_k, sigmoid appears to underestimate it. This further shows that the affine estimator is robust to estimated values, since soft-min-max leads to good performance while underestimating α_k. This seems to suggest that having an accurate estimate of β_k is more important than one for α_k. In theory, from Eq. (3.32) we see that, unlike β_k, α_k can be estimated within a constant factor of the true value without hurting the performance of $\hat{\Delta}_{\text{affine}}$. However, further analysis is required to fully understand what kind of inaccuracies still result in high

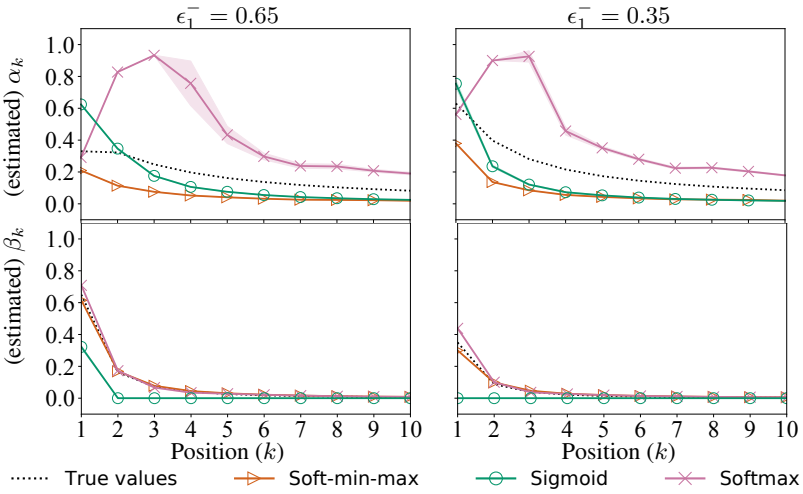


Figure 3.4: Bias parameters estimated on $8 \cdot 10^6$ clicks using different final activation functions, under normal position bias, and varying degrees of trust bias. Results are averaged over four runs; shaded area indicates the standard deviation. All results are based on the Yahoo! Webscope dataset. Top row: α_k; bottom row: β_k.

performance. These results also suggest that there are promising opportunities for novel ways to estimate trust bias from click data.

In conclusion, our results show that using the affine estimator still leads to good performance when it is based on estimated bias values. In particular, we have found that using the soft-min-max function leads to the best results, and that the affine estimator can still get near-optimal performance when bias values are not completely accurate. We conclude that the affine estimator is robust w.r.t. estimated bias values.

3.8 Conclusion

In this chapter, we have considered CLTR in situations with both position bias and trust bias and addressed RQ2 of the thesis. We have proven that no IPS estimator can correct for trust bias, including the Bayes-IPS estimator specifically designed for this bias [2]. The reason for this inability is that trust bias is an affine transformation between relevance probabilities and click probabilities, and IPS estimators can only correct for linear transformations.

As a solution, we have introduced the novel affine estimator, which applies affine transformations to clicks: it both reweights clicks and penalizes items for being displayed at ranks where the users' trust is high. We proved that the affine estimator is unbiased w.r.t. both position bias and trust bias, thus it is the first CLTR method that can deal with both of these biases simultaneously. Furthermore, the affine estimator can be considered an extension of the existing IPS approach: when no trust bias is present the affine estimator optimizes the same objective as the existing IPS estimator. Our

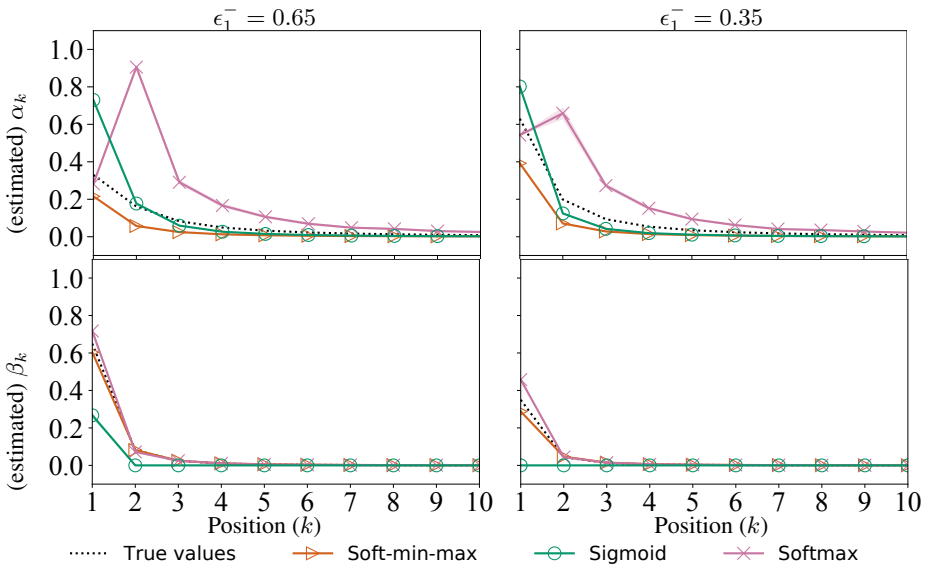


Figure 3.5: Bias parameters estimated on $8 \cdot 10^6$ clicks using different final activation functions, under severe position bias, and varying degrees of trust bias. Results are averaged over four runs; shaded area indicates the standard deviation. All results are based on the Yahoo! Webscope dataset. Top row: α_k; bottom row: β_k.

experimental results show that using the affine estimator CLTR can approximate the optimal model when both position bias and trust bias are present, while existing IPS-based estimators cannot. Furthermore, our results suggest that the estimator is robust to bias estimation, as performance is stable when the bias parameters are estimated from interactions.

With the introduction of our affine estimator, the CLTR framework has been expanded to correct for trust bias on top of position bias. Future work can continue this trend, for instance, by combining the policy-aware approach by Oosterhuis and de Rijke [96] with the affine estimator, perhaps an estimator that corrects for both item-selection bias and trust bias can be found. Furthermore, previous work has found position bias estimation using randomization to be very powerful [3, 131]. Thus, there seems to be potential for methods based on randomization for estimating trust bias, possibly another fruitful direction for future research.

Unbiasedness of affine correction, similar to IPS, depends on accurate estimations of bias parameters. In the next chapter, we provide an in-depth analysis of this dependency and propose a method whose unbiasedness does not rely on the accurate knowledge of bias parameters assumption.

4

Relevance and Propensity Have a Cyclic Dependency: Mixture-Based Correction as a Solution

In Chapter 3 we proposed affine correction (AC) as a generalization of inverse propensity scoring (IPS) that corrects for position bias and trust bias. IPS and AC provably remove bias, conditioned on an accurate estimation of the bias parameters. Estimating the bias parameters, in turn, requires an accurate estimation of the relevance probabilities. This cyclic dependency, which is addressed in RQ3, introduces practical limitations in terms of sensitivity, convergence, and efficiency. In this chapter, we propose a new correction method for position and trust bias in counterfactual learning to rank (CLTR) in which, unlike existing methods, the correction does not rely on relevance estimation. Our proposed method, mixture-based correction (MBC), is based on the assumption that the distribution of the click-through rates over the items being ranked is a mixture of two distributions: the distribution of click-through rates for relevant items and the distribution of click-through rates for non-relevant items. We prove that our method is unbiased. The validity of our proof is not conditioned on accurate bias parameter estimation. Our experiments show that MBC, when used in different bias settings and accompanied by different learning to rank algorithms, outperforms AC, the state-of-the-art method for correcting position and trust bias, in some settings, while performing on par in other settings. Furthermore, MBC is orders of magnitude more efficient than AC in terms of the training time.

4.1 Introduction

A number of techniques have been proposed to debias clicks and to estimate the probability of relevance based on the probability of clicks. A well-known method is *inverse propensity scoring* (IPS) [58, 130], which corrects for the position bias in clicks. IPS relies on the *examination hypothesis*, i.e., an item is clicked if it is examined

This chapter was published as: A. Vardasbi, M. de Rijke, and I. Markov. Mixture-based correction for position and trust bias in counterfactual learning to rank. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 1869–1878, 2021.

and perceived to be relevant by a user. As the name suggests, in IPS clicks are reweighed by the inverse of the examination probability, a.k.a. propensity. IPS is proved to be unbiased when the clicks suffer from position bias [58]. *Affine correction* (AC), introduced in Chapter 3, generalizes IPS to also correct for trust bias. AC has been proved to be unbiased when the clicks suffer from both position and trust bias. The proofs of the unbiasedness of IPS and AC depend on knowledge of the bias parameters. Accurately estimating the bias parameters, in turn, depends on obtaining accurate relevance estimations, which is as hard as the learning to rank (LTR) problem itself. In the literature, this cyclic dependency is solved by a regression-based EM (rbEM) algorithm that simultaneously learns the ranker as well as the bias parameters [2, 131], similar to what was used in Section 3.5.3. We argue that integration of a regression function into the standard expectation maximization (EM) leads to a number of practical limitations in terms of (i) sensitivity to the regression function, (ii) a lack of guarantees that EM converges to a zero gradient, and (iii) low efficiency of the algorithm. In this chapter, we address this problem by asking the following question:

RQ3 Is it possible to break the cyclic dependency between relevance and bias parameters when correcting for position and trust bias?

We answer this question by proposing a novel debiasing method, *mixture-based correction* (MBC). Inspired by the idea of score distributions [6], we assume that the probability of seeing a specific click-through rate (CTR) for an item at a position in a ranking is a mixture of CTR probabilities for relevant and non-relevant items appearing on that position. More specifically, we assume that an item is clicked if, and only if, one of the following two disjoint events occurs: (i) a user examines the item and the item is actually relevant (i.e., this is a click on a relevant item), or (ii) the user examines the item, the item is not relevant, but the user clicks on it anyway due a certain bias, e.g., trust in the search engine [2] or visual attractiveness of the item [27] (i.e., this is a click on a non-relevant item). For each position in a ranking, MBC estimates the distribution of CTRs for these two events and calculates the full distribution of CTRs on that position as their mixture. Then, the probability of relevance for a given item is calculated by Bayes' rule, as the posterior probability of relevance, given the observed CTR over that item. Finally, the estimated probabilities of relevance are used as labels in a standard LTR algorithm.

We prove that MBC gives an unbiased estimator of the probability of relevance, without any prior knowledge of the bias parameter values. This is a step forward, as IPS and AC *do* rely on prior knowledge of the bias parameter values to be unbiased. Below, we show theoretically how inaccurate bias parameters prevent AC from completely removing bias.

We confirm our theoretical advances with a set of semi-synthetic experiments. We show that the ranking performance of different LTR algorithms, trained on the relevance estimates of MBC, always converges to the ranking performance where the true relevance labels are available. We also compare MBC with the state-of-the-art correction method for position and trust bias, i.e., AC. We compare them by training LTR algorithms over MBC's and AC's respective corrected outputs. We show that in several cases MBC outperforms AC by filling the gap between AC's ranking performance and the true relevance case. Finally, since both MBC and AC depend to

the assumption of a click model to infer and correct for the bias, we conduct robustness experiments in terms of click model mismatch. Specifically, we show that when clicks adhere to the dependent click model (DCM) or the user browsing model (UBM), but a different click model, such as the position-based model (PBM), is assumed by the correction methods, MBC is more robust, i.e., its ranking performance is affected less compared to AC.

In summary, the contributions of this chapter are:

1. We propose a new debiasing method, mixture-based correction (MBC), for correcting position and trust bias, and prove its unbiasedness. Our proof is stronger than the unbiasedness proofs for existing methods such as AC from Chapter 3, as it does not rely on the assumption that the bias parameters are known.
2. We show experimentally that, when used with LTR methods, MBC outperforms AC, the state-of-the-art correction method for position and trust bias, in several settings, while having similar performance in other settings.
3. We show that MBC is orders of magnitude faster than AC, in terms of the training time.
4. We provide experimental evidence that MBC is more robust to click model mismatch compared to AC.

4.2 Background

The majority of prior work on unbiased LTR, tries to correct for the mismatch between the distribution of clicks and relevance probability due to bias. Bias in clicks means that not all relevant items have the same a priori chance of being clicked. E.g., position bias means that relevant items at the top of a result list usually absorb more clicks than lower ranked relevant items [57]; and trust bias means that users trust a search engine and click on higher ranked non-relevant items more than lower ranked items [2]. Usually, these types of bias are modeled with the help of click models [29].

Below, we review existing methods for correcting position and trust bias. After discussing AC as the state-of-the-art correction method for position and trust bias, we analyze its relevance estimation error and show how inaccurate bias parameters cause AC to remain biased. We also discuss other work related to this chapter.

4.2.1 A Review of AC

As we have seen in Chapter 3, in the presence of trust bias, the click probability should be written as follows (for brevity we drop the $(\cdot \mid q, d, k)$ conditions from all the probabilities, where q and d represent the query and item and k is the item position in the results list.):

$$\begin{aligned} P(C=1) = \; & P(E=1) \cdot P(R=1) \cdot P(C=1 \mid R=1, E=1) \\ & + P(E=1) \cdot P(R=0) \cdot P(C=1 \mid R=0, E=1) \\ = \; & \alpha P(R=1) + \beta. \end{aligned} \tag{4.1}$$

where C, E and R indicate click, examination and relevance binary indicators. We proved that the following correction gives an unbiased estimate of the relevance in this

situation:

$$\hat{r}_{q,d} = \frac{c_{q,d,k} - \beta_{q,d,k}}{\alpha_{q,d,k}}. \tag{4.2}$$

where $c_{q,d,k}$ is the click over document d of query q at position k; α and β are bias parameters (4.1); and $\hat{r}_{q,d}$ is the estimated relevance of d.

4.2.2 Regression-Based EM

IPS and AC are unbiased only if the value of the bias parameters such as α and β are known, or accurately estimated. Since the standard EM requires the availability of multiple sessions of the same query with different items ordering to work properly [26], Wang et al. [131] proposed to use *regression-based EM* (rbEM) to solve the sparsity problem. In the rbEM, the $P(R = 1 \mid q, d)$ values obtained in the M-step are first used to fit a regression function, and then, the output of this regression function is used in the next E-step. Though rbEM leads to good results in the CLTR framework with IPS and AC [2, 122, 131], in this chapter we argue that the integration of a regression function into the EM leads to multiple practical limitations, as we explain in Section 4.6.1.

4.2.3 Error Analysis of AC

Let us denote the true relevance of document d to query q by $r_{q,d}$, and the probability $P(r_{q,d} = 1)$ by $\gamma_{q,d}$. In the presence of trust bias, according to Eq. (4.1), we have:

$$\mathbb{E}_c\left[c_{q,d,k}\right] = \alpha_{q,d,k} \cdot \gamma_{q,d} + \beta_{q,d,k}. \tag{4.3}$$

In what follows we drop the subscripts for brevity.

Supposing α' and β' are estimates of α and β obtained from the rbEM algorithm, AC estimates r as follows:

$$\hat{r} = \frac{c - \beta'}{\alpha'}. \tag{4.4}$$

In order to have an unbiased estimator, we need to ensure that $\mathbb{E}_c\left[\hat{r}\right] = \mathbb{E}_r\left[r\right]$. Therefore, we are interested in $e = |\hat{r} - r|$. Using Eq. (4.1), we can calculate the expectation as follows:

$$\begin{aligned} \mathbb{E}_{c,r}\left[e\right] &= \gamma \left| 1 - \frac{\alpha + \beta - \beta'}{\alpha'} \right| + (1 - \gamma) \left| \frac{\beta - \beta'}{\alpha'} \right| \\ &= \gamma \frac{|\Delta\alpha + \Delta\beta| - |\Delta\beta|}{\alpha'} + \frac{|\Delta\beta|}{\alpha'}, \end{aligned} \tag{4.5}$$

where $\Delta\beta = \beta' - \beta$ and $\Delta\alpha = \alpha' - \alpha$ are estimation errors of the bias parameters. It is important to have the average error converge to zero as the number of sessions associated with the query grows. Eq. (4.5) shows that only when $\Delta\alpha \to 0$ and $\Delta\beta \to 0$, i.e., when the parameter estimations obtained from rbEM are accurate, this is the case. In summary, inaccurate bias parameters estimations cause the AC method to remain biased, even with infinitely many training sessions.

4.2.4 Mixture of Distributions

Our assumption of having a mixture of relevant and non-relevant distributions is related to score distribution models [6]. But MBC differs in two important ways. First, unlike scores, clicks are feedback and constitute a source of supervision. This makes CLTR used with MBC a supervised learning algorithm as opposed to the unsupervised algorithms based on score distribution models. Second, a score distribution model is built over the corresponding list of items for a single query, whereas our mixture model is built over the items with the same examination probability for different queries. In this sense, our model has a global view over all queries, while the score distribution models have a local view over one query.

4.2.5 Other Related Work

Instead of using rbEM, Ai et al. [4] propose to use the dual learning algorithm (DLA). DLA is shown to be effective in estimating the bias parameters and leading to an unbiased LTR. However, it only models the position bias and it is not clear how it can be extended to work with trust bias.

Qin et al. [103] use rbEM to estimate the attribute-based propensity, which considers different platforms and feedback sources in addition to the items positions. Dai et al. [31] define a utility based on the click probability and propose to optimize that utility directly, instead of optimizing retrieval metrics with the hope that they may indirectly improve the CTR. Their approach solves the problem of bias parameter estimation by directly learning a position-aware click model from user interactions.

What our proposed method, MBC, contributes on top of the related work discussed above is that it breaks the cyclic dependency between the bias parameters and relevance probability. This means that in MBC the bias parameters are inferred only using the user interactions, without any direct reliance on the relevance probabilities. In other words, in existing methods relevance estimation is unbiased if the bias parameters are accurately set, and the bias parameters can be set accurately if the relevance estimation is precise. In contrast, with MBC, the correction and bias parameter estimation are performed at the same time, without any reliance on relevance estimation. This enables us to avoid the use of regression functions inside the EM algorithm, which is shown in Section 4.6 to have practical limitations.

Finally, in [24, 121] it is argued that a mismatch between the actual model of clicks and the assumed click model for correction hurts ranking performance of the correction methods. We address this issue by showing that MBC is robust to click model mismatch, specifically, when actual clicks adhere to DCM or UBM while MBC assumes PBM for correction.

4.3 Mixture-based Correction

In this section, we explain our mixture-based correction (MBC) method and prove that it gives an unbiased estimate of relevance.

Algorithm 4.1: Mixture-based correction

Input: Click-through rates
Output: Estimates of relevance probability
1 Constitute sets of items with the same $P(E = 1)$;
2 **forall** *set of items with the same* $P(E = 1)$ **do**
3 Fit a 2-component mixture model to Eq. (4.6);
4 Output the relevance of items according to Eq. (4.7);
5 **end**

4.3.1 Method

Our idea is to infer the relevance of an item to a user's query based on the observed CTR for that query-item pair. Similarly to previous work on CLTR, we assume that user clicks on search results follow the examination hypothesis [2, 4, 58, 96, 122, 131], that is, a click on an item (or, consequently, the CTR for that item) is affected only by how likely the item is to be examined and perceived relevant by a user. So there are two components, namely, examination and relevance of an item, that contribute to a click on the item. Our goal is to estimate the relevance component.

To rule out the examination component, we consider items with the same examination probability $P(E = 1)$. In practice, $P(E = 1)$ is not known and one needs a click model to decide which items have the same $P(E = 1)$. We will address this issue later in this section. For now, assume that we know which items have the same $P(E = 1)$.

For a set of items with the same examination probability $P(E = 1)$ there is a certain distribution of CTRs, $P(CTR = x)$. Assuming binary relevance,[1] this distribution has two parts: one for relevant items and one for non-relevant items. So $P(CTR = x)$ can be seen as a mixture of two separate distributions: the distribution $P(CTR = x \mid R = 1)$ of CTRs of relevant items and the distribution $P(CTR = x \mid R = 0)$ of CTRs of non-relevant items. Formally:

$$P(CTR = x) = P(R = 1) \cdot P(CTR = x \mid R = 1) + P(R = 0) \cdot P(CTR = x \mid R = 0). \tag{4.6}$$

Now, we can reach our goal and calculate the probability of relevance based on the observed CTR using Bayes' rule:

$$P(R = 1 \mid CTR = x) = \frac{P(R = 1) \cdot P(CTR = x \mid R = 1)}{P(CTR = x)}. \tag{4.7}$$

These relevance probabilities can ultimately be used for CLTR. Algorithm 4.1 summarizes the above steps of MBC.

For our MBC method to work, it remains to discuss two things: (i) How to estimate the mixture in Eq. (4.6) (Line 3 in Alg. 4.1); and (ii) How to get items with the same examination probability (Line 1 in Alg. 4.1). This is what we turn to next.

[1] Graded relevance can be considered as the probability $P(R = 1)$. See Section 4.4 for further discussions.

Mixture estimation. To infer distributions $P(CTR = x \mid R = i)$ and priors $P(R = i)$ for $i \in \{0, 1\}$ in Eq. (4.6), parametric approaches can be used. For example, we can assume a Gaussian or a binomial mixture model. As is common with parametric mixture models, we use standard EM to learn the above distributions and priors [88].

The limitation of the parametric approach to estimating mixtures is that it depends on the choice of the underlying parametric distribution (e.g., Gaussian, binomial, etc). However, in Section 4.3.2, we show that as long as there are enough clicks, the choice of this distribution is not essential to our method. Also, in our experiments in Section 4.5.5, we compare Gaussian and binomial mixture distributions and show that, in practice, both converge to the same performance.

Items with the same examination. The MBC method works on sets of items with the same examination probability. Note that MBC does not require the exact values of examination probabilities, it only requires to know which items have the same examination. To detect the desired sets of items, we propose to use click models [29].

For instance, we can assume that clicks adhere to the PBM as is common in CLTR [2, 4, 58, 96, 122, 131]. In PBM, *all items at the same position in a results list* have the same examination probability. And that is all we need from the assumed click model. For cascade models, such as the DCM [50] and the UBM [34], *all items at the same position* and with *the same pattern of relevant items above them* have the same examination probability. In the case of DCM and UBM, the desired set of items can be detected recursively: to collect items with the same examination probability at position k, we first use MBC to detect all relevant items at positions above k, and then we group items with the same pattern of relevant items above k. Note that this recursive process is not cyclic: the grouping of clicks at rank k depends on the reliability of the model at rank $k - 1$. Again, no parameter estimation for the assumed click models is required.

Remark 4.1. Detecting items with the same examination may seem a bottleneck in real world scenarios. For MBC to work properly, the distributions of relevant and non-relevant CTRs should be separable. This means that the condition of *items with the same examination* can be loosened by *items with **almost** the same examination*. To elaborate, suppose a set of items are considered whose examination probabilities are either θ or θ'. Each examination probability leads to one distribution for relevant CTRs, and one for non-relevant CTRs: there are two relevant CTR distributions and two non-relevant CTR distributions in the set. The relevant (non-relevant) CTR distribution of all the items in the set is itself a mixture of two distributions corresponding to θ and θ'. As long as these two relevant/non-relevant distributions of all items are separable, MBC would be able to distinguish relevant items from non-relevants, and the LTR remains unbiased. This argument is easily extended to cases with more than two different examination probabilities. We will discuss this remark more in Section 4.5.3 and give a toy example for further clarifications.

In the remainder of this section, we prove that the MBC inferred relevance labels are unbiased estimations of the true relevance labels. Consequently, a LTR algorithm trained on these corrected values will be an unbiased LTR.

4.3.2 Unbiasedness of MBC

In this chapter, similar to most previous work on online LTR and CLTR, we assume that clicks on different sessions are independent [2, 4, 29, 55, 58, 95, 96, 121, 122, 131, 144]. As discussed in Section 4.3.1, we consider the set of items with almost the same examination probability and fit a mixture model for each such set. We assume that clicking on a relevant item is a random variable with mean μ_1 and variance σ_1^2. Similarly, clicking on a non-relevant item has mean μ_0 and variance σ_0^2. For a unique query, repeated in n sessions, assume the clicks over the item x are given by $\{c_1^{(x)}, c_2^{(x)}, \dots, c_n^{(x)}\}$. We prove that the CTR defined as

$$v_x = \frac{c_1^{(x)} + c_2^{(x)} + \cdots + c_n^{(x)}}{n}, \tag{4.8}$$

can be used to estimate the relevance of x, given that n is large enough.

Theorem 4.1. *Assuming independent clicks in different sessions, the clustering of the CTR signal into relevant and non-relevant items converges in probability to the true relevance of the items.*

Proof. Based on the assumptions, the $c_i^{(x)}$'s constitute a sequence of independent and identically distributed (i.i.d.) random variables. According to the central limit theorem (CLT), as n grows, v_x will converge in probability to $\mathbb{E}_c\left[c_i^{(x)}\right]$, which is either μ_0 or μ_1. We are interested in the case where a full recovery of mixture models is possible, i.e., one can fully separate the distribution of relevant CTRs from the distribution of non-relevant CTRs. The variance of v_x diminishes linearly by n. Consequently, given any threshold value for the variance for which a full recovery is possible, there is a sufficiently large n that leads to the given threshold value. This means that there exists an n such that a full recovery of the mixture components is possible. □

We used the central limit theorem for the proof of Theorem 4.1, which does not rely on any specific distribution. In order to get a better understanding of what constitutes a sufficiently large n for full recovery, we discuss the special case of a Gaussian mixture. There is a rich literature on the analysis of the recoverability of Gaussian mixture models [see, e.g., 28]. In [81] a simple condition is given for *almost full recovery* of Gaussian mixtures, which translates to the setting of this chapter as follows:

$$n = \Omega\left(\left(\frac{\max(\sigma_0, \sigma_1)}{\mu_1 - \mu_0}\right)^2\right). \tag{4.9}$$

This means that for any μ_i and σ_i values, if we increase the number of sessions n so that Eq. (4.9) holds, the CTR of relevant and non-relevant items can be almost fully separated. In our experiments, where the parameters are set based on previous real-world studies such as [2], Eq. (4.9) becomes $n = \Omega(1)$ and we find $n \geq 15$ to be a suitable value based on the convergence of ranking performance.

4.4 Experimental setup

In this section, we discuss the experimental setup used to demonstrate the effectiveness of our proposed MBC method. Following previous CLTR studies [2, 4, 55, 58, 96, 122], we measure the effectiveness of click debiasing by the ranking performance of LTR when used on top of debiasing methods. Our experiments are performed on two publicly available LTR datasets with query-document features and relevance labels, while clicks are simulated.

4.4.1 Datasets

As a regular choice in LTR research [4, 55, 58, 96, 122], we use two popular LTR datasets: Yahoo! Webscope [25] and MSLR-WEB30k [102]. For each query, these datasets contain a list of documents with human-generated 5-level relevance labels. Yahoo! has 29 670 queries, 23.84 documents per query on average, and uses 700-feature vectors to represent query-documents; MSLR has 31 339 queries, 120.19 documents per query, and uses 136 features. We use the default provided training, validation and test splits for each dataset. We only use the first fold of MSLR.

4.4.2 Click Simulation

In our experiments, we measure the performance of LTR trained on user clicks. The Yahoo! and MSLR datasets, however, do not contain click information. Following a long line of previous studies [4, 53, 55, 58, 96, 122], we simulate clicks as follows.

Production ranker. First, we simulate a production ranker that is used to create a ranked list of items for each query. Following [58], we train a LTR method on a very small number of randomly selected queries. The intuition is to provide a production ranker that is better than a random ranker, but still has room for improvement. We select 20 random queries from each dataset and use LambdaMART [21] to train a production ranker. To train LambdaMART, we use the LightGBM package (version 3.2.1.99) [62] with the following parameters: 300 trees, 31 leaves and a learning rate of 0.05. Training in this way with 20 queries gives us a decent ranker that performs considerably better than a random ranker, while still having room for improvement. Hence, we can exploit user interactions to improve this production ranker.

The production ranker is then used to rank items for each query. We cut the list of items at top-m, as do real-world search engines. We report results for $m = 20$. We also performed experiments with the top-50, but since the results showed no significant difference compared to the top-20, we do not report them here.

User clicks. To simulate clicks, we follow previous studies on trust bias [2, 122]. Given a list of items returned by the production ranker for a query q, we first compute the click probability for each item x and position k using Eq. (4.1) and the PBM assumption. (For experiments in Section 4.5.3 we replace PBM with DCM and UBM.) Then, for each item x and position k we simulate a click by sampling from this Bernoulli distribution. Unless stated otherwise, we use $8M$ clicks (with uniformly repeated queries) to train the correction methods. In our experiments, both MBC and AC begin to converge to their final performance after $2M$ clicks. We choose $8M$ to be on the safe side. Eq. (4.1)

depends on two quantities: (i) relevance probabilities; and (ii) bias parameters. We will discuss both below.

Relevance probabilities. Both datasets used in our experiments provide graded relevance labels. For simulating the clicks in a graded relevance setting, a transformation function is required to change the integer grades into valid relevance probabilities. We employ the following two strategies, assuming $y \in \{0, \dots, y_{\max}\}$ is the relevance grade:

1. **Binarized**: Following [58, 122] relevance probability can itself be binary: $P(R = 1 \mid y) = 1 \quad \text{iff} \quad y > \frac{y_{\max}}{2}$.
2. **Graded**: The grades can also be turned into probabilities using a linear transformation [98]: $P(R = 1 \mid y) = \frac{y}{y_{\max}}$.

Bias parameters. Similarly to [4, 55, 58, 96, 122], the position bias for a position k is computed as $P(E = 1 \mid k) = k^{-\eta}$, where the parameter η controls the severity of the position bias. Usually, $\eta = 1$ is considered in the CLTR literature [4, 58, 122]. In our experiments we consider $\eta \in \{1, 2\}$.

For the trust bias, we follow [122] and for each position k compute the parameters as follows:[2]

$$P(C = 1 \mid R = 1, E = 1, k) = 1 - \frac{\min(k, 20) + 1}{100} \tag{4.10}$$

$$P(C = 1 \mid R = 0, E = 1, k) = \frac{0.65}{\min(k, 10)}. \tag{4.11}$$

4.4.3 LTR

We train a LTR method over the corrected output of different click debiasing methods. In order to show the consistency of our results over different LTR methods, we use two LTR approaches: (i) **LambdaMART** [21]; and (ii) **DNN**, a neural network similar to that of [122]. The LambdaMART implementation of LightGBM only works with integer labels, but the input of LTR in our experiments is the relevance probability, which is non-integer. To solve this problem, we made some minor modifications to the source code of LightGBM in order to make it work with non-integer inputs as well.

4.4.4 Baselines

We compare the results of MBC to those of AC [122], the state-of-the-art click debiasing method for position and trust bias. We do not include IPS in our comparison, since (i) it cannot correct for trust bias, and (ii) AC is a generalization of IPS that leads to the same performance when trust bias is absent [122]. In summary, we compare the ranking performance of LTR trained on debiased clicks of our MBC method, with the following settings:

1. **Ideal-AC**: AC with true bias parameters. This gives the highest potential of AC and is not realistic, since it uses the true values for bias parameters;
2. **AC**: LTR trained on debiased clicks of AC, using rbEM for propensity estimation (See 4.6.2 for the choice of regression function);

[2] Similar values were reported in the real-world experiments in [2].

Table 4.1: NDCG@10 comparison of MBC and AC on Yahoo! Webscope and MSLR-WEB30k datasets under normal position bias ($\eta = 1$). Superscripts * and † indicate significance compared to the other correction method with $p < 0.01$ and $p < 0.1$, respectively.

		Yahoo! Webscope		MSLR-WEB30k	
		Binarized	**Graded**	**Binarized**	**Graded**
LambdaMART	*Trained using clicks*				
	No Correction	0.692	0.691	0.400	0.402
	AC	0.758	0.763	0.426	**0.477**†
	MBC	**0.760***	**0.767**†	**0.428**	0.474
	Trained using oracle knowledge				
	Ideal-AC	0.758	0.771	0.423	0.486
	Rel. Probs	0.759	0.774	0.429	0.491
DNN	*Trained using clicks*				
	No Correction	0.713	0.716	0.401	0.415
	AC	0.736	**0.746***	0.404	**0.448***
	MBC	**0.744***	0.744	**0.406**	0.440
	Trained using oracle knowledge				
	Ideal-AC	0.742	0.749	0.409	0.451
	Rel. Probs	0.743	0.750	0.416	0.453

3. **No correction**: LTR trained on clicks without any debiasing;
4. **Relevance probabilities**: LTR trained on the true relevance probabilities, obtained from the true relevance labels. This depends on the strategy that is used to transform the integer relevance grades into probabilities (see paragraph *Relevance probabilities* in Section 4.4.2).

In our experiments, we use the same LTR algorithm for all the methods listed above, so that any differences in ranking performance are solely due to the correction method, and no other factors such as LTR performance.

4.4.5 Metrics

We measure the ranking performance of different methods by normalized discounted cumulative gain (NDCG). All reported nDCG@10 results are an average of eight independent runs. We use the Student's t-test to determine significant differences.

4.5 Results

Our experimental results are centered around three benefits of MBC compared to AC. The first – and most important – benefit is the ranking performance (Section 4.5.2) and

Table 4.2: NDCG@10 comparison of MBC and AC on Yahoo! Webscope and MSLR-WEB30k datasets under severe position bias ($\eta = 2$). Superscripts * and † indicate significance compared to the other correction method with $p < 0.01$ and $p < 0.1$, respectively.

		Yahoo! Webscope		MSLR-WEB30k	
		Binarized	**Graded**	**Binarized**	**Graded**
LambdaMART	*Trained using clicks*				
	No Correction	0.689	0.689	0.396	0.397
	AC	0.725	0.740	0.337	0.431
	MBC	**0.753***	**0.748***	**0.420***	**0.454***
	Trained using oracle knowledge				
	Ideal-AC	0.725	0.740	0.337	0.432
	Rel. Probs	0.759	0.774	0.429	0.491
DNN	*Trained using clicks*				
	No Correction	0.710	0.707	0.402	0.400
	AC	0.736	**0.746***	0.405	**0.448***
	MBC	**0.741***	0.740	**0.419***	0.439
	Trained using oracle knowledge				
	Ideal-AC	0.737	0.746	0.405	0.446
	Rel. Probs	0.743	0.750	0.416	0.453

efficiency (Section 4.5.4) improvement. The second is its robustness to click model mismatch (Section 4.5.3). The third benefit is that it requires almost no hyper-parameter tuning as opposed to rbEM. Since MBC is based on a standard mixture model, it is free of the so-called hyper-parameters prevalent in deep learning models. On the other hand, rbEM uses a regression function that has a significant impact on the effectiveness and efficiency of the unbiased LTR algorithm (as we show in Section 4.6.2).

4.5.1 An Insider's Look into Corrected Clicks

Before proceeding with the main experimental analysis, we compare the shape of the corrected clicks obtained from MBC and AC. This qualitative analysis is insightful in understanding the intrinsic difference between the two methods.

Figure 4.1 looks into the corrected CTR of MBC and AC. In this figure, the CTR in two different positions of a ranked list are observed. We used the binarized relevance probability (Section 4.4) in this figure for simpler illustration purposes. We see that with enough sessions per query, the relevant and non-relevant items become completely disjoint. Consequently, MBC is able to correctly distinguish between the relevant and non-relevant items and infer their relevance label. The complex transformation of CTRs into relevance labels in MBC allows for accurate estimations of relevance. On the other

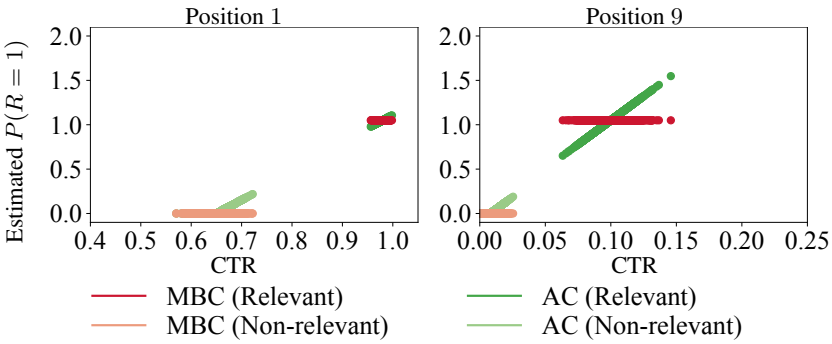


Figure 4.1: Correction comparison of our MBC method with AC in inferring the relevance label from the observed CTR.

hand, the linearity of AC leads to wider range of relevance labels, scattered around the true values of zero and one.

4.5.2 Ranking Performance of MBC and AC

In this section, we try to determine the effectiveness of MBC method in debiasing clicks, and compare it to AC as the state-of-the-art method for correcting position and trust bias. For that, we measure the ranking performance of a selected LTR algorithm trained over the corrected output of MBC and AC.

Tables 4.1 and 4.2 shows the comparison of MBC and AC in terms of NDCG@10, under normal and severe position bias, respectively. In these two tables, we compare the performance for different LTR algorithms: LambdaMART and DNN, and different strategies for transforming relevance grades to probabilities: binarized and graded.

The results show that in most cases MBC performs significantly better than AC. When LambdaMART is used as the LTR algorithm, MBC significantly outperforms AC on both datasets, both binarized and graded settings and both bias severity cases ($\eta = 1$ and $\eta = 2$), the only exception being the graded MSLR with normal position bias. Things are different for the DNN case. AC significantly outperforms MBC in the graded setting in both datasets. We also see that, with a single correction method, in some settings LambdaMART performs better than DNN, while in others DNN is the winner. Considering the LTR algorithm as a hyperparameter, i.e. for each correction method in each setting, selecting the LTR with the higher ranking performance, we can claim that MBC corrects more effectively than AC, as the best performing MBC leads to better results than the best performing AC. For instance, in graded Yahoo! with severe bias, the best performing AC is obtained from DNN: 0.746 vs 0.740, while the best performing MBC is due to LambdaMART: 0.748 vs 0.740. We see that LambdaMART MBC outperforms DNN AC in this case.

As can be seen in Tables 4.1 and 4.2, there is a gap between AC (where the bias parameters are estimated using rbEM) and Ideal-AC (where the bias parameters are assumed to be known by an oracle). This shows the dependency of AC on the accuracy

Table 4.3: NDCG@10 comparison of MBC and AC with PBM assumption on Yahoo! Webscope and MSLR-WEB30k datasets, when DCM cascade model is used for simulating the clicks. Superscripts * and † indicate significance compared to the other correction method with $p < 0.01$ and $p < 0.1$ respectively.

		Yahoo! Webscope		MSLR-WEB30k	
		Binarized	**Graded**	**Binarized**	**Graded**
LambdaMART	No Correction	0.694	0.691	0.405	0.402
	AC	0.739	0.753	**0.411**†	0.469
	MBC	**0.752***	**0.760***	0.407	**0.477***
	Rel. Probs	0.756	0.770	0.409	0.485
DNN	No Correction	0.709	0.712	0.407	0.407
	AC	0.728	0.733	0.364	0.433
	MBC	**0.741***	**0.743***	**0.408***	**0.446***
	Rel. Probs	0.743	0.749	0.412	0.453

of the bias parameters. As a reminder, this dependency is one of our motivations to propose the novel MBC method.

With severe position bias ($\eta = 2$), as a result of very low examination probabilities, we observe a noticeable drop in the performance of AC and Ideal-AC in some cases. Specifically, for the binarized MSLR with LambdaMART, we observe that AC performs even worse than the no correction case. This observation underlines the need for variance reduction techniques for the AC method.

Remark 4.2. We see that in some of the binarized cases, MBC performs slightly better than the Rel. Probs case, which may seem counterintuitive. The reason is that the evaluation is performed on the graded test set for comparability considerations. As a result, the relevance grades of $\{3, 4\}$ as well as $\{0, 1, 2\}$ are treated the same in the training, but distinguished in evaluation. This is further observable in comparing the Rel. Probs in the binarized and graded settings: the binarized Rel. Probs is not the theoretical upper bound for the ranking performance, the graded Rel. Probs is.

4.5.3 Click Model Mismatch

Next, we investigate the effect of a mismatch between a click model that generates clicks and a click model that is used for debiasing:

> *How do MBC and AC methods perform when PBM is assumed for debiasing, whereas user clicks adhere to a different click model?*

This is an important question as in reality none of the existing click models can fit user clicks perfectly and there is always a mismatch between a click model that user clicks adhere to and the one that is assumed for debiasing. For further discussion please refer to Chapter 2.

Table 4.4: NDCG@10 comparison of MBC and AC with PBM assumption on Yahoo! Webscope and MSLR-WEB30k datasets, when UBM cascade model is used for simulating the clicks. Superscripts * and † indicate significance compared to the other correction method with $p < 0.01$ and $p < 0.1$ respectively.

		Yahoo! Webscope		MSLR-WEB30k	
		Binarized	**Graded**	**Binarized**	**Graded**
LambdaMART	No Correction	0.691	0.690	0.397	0.400
	AC	0.753	0.759	0.387	**0.474***
	MBC	**0.756***	**0.763**†	**0.410***	0.469
	Rel. Probs	0.756	0.770	0.409	0.485
DNN	No Correction	0.709	0.716	0.391	0.406
	AC	0.736	**0.743**	0.402	**0.445***
	MBC	**0.741***	0.742	**0.410**†	0.441
	Rel. Probs	0.743	0.749	0.412	0.453

In this section, we want to examine the robustness of different correction methods in terms of their assumed click model. Specifically, we simulate clicks based on two well-known models: DCM [50], that is a cascade-based click model, and UBM [34], that has features of both position-based and cascade-based models. In order to have realistic experiments, we learn the parameters of these click models using the Yandex dataset,[3] which contains a large amount of clicks from a production search engine, and the PyClick library.[4] Since the Yandex dataset has the top-10 results, the parameters are obtained for the top-10 and our experiments in this section are reported for the top-10 setting.[5] Then, we use the learned parameters to simulate clicks similarly to Section 4.4.2, this time using DCM and UBM instead of PBM.

For each case, we debias clicks using MBC and AC with the PBM click model. Tables 4.3 and 4.4 show the ranking performance of these correction methods.

We observe that MBC always improves over the no correction case, while AC fails to do so in some cases: Binarized UBM (Table 4.4) with LambdaMART and Binarized DCM (Table 4.3) with DNN. Furthermore, in most cases, there is a gap between the corrections and the Rel. Probs performance. This gap is due to the mismatch between the actual and assumed click models.

Comparing MBC and AC, we see that MBC outperforms AC in most cases, suggesting that our MBC method is more robust to the click model mismatch. This observation coincides with our expectation, since MBC does not rely on the parameters of click models: as long as the click probabilities over relevant and non-relevant items are separable for each position, MBC with PBM works fine. In other words, unlike AC where the examination probability of each position is estimated by a single value,

[3] `https://www.kaggle.com/c/yandex-personalized-web-search-challenge`

[4] `https://www.github.com/markovi/PyClick`

[5] Hence, the Rel. Probs results in this section may be different from those in Table 4.1, where the top-20 is used instead.

in MBC different items at the same position are allowed to have different examination probabilities (see Remark 4.1).

To further illustrate the difference, we give a toy example. Suppose there are two sets of sessions S_1 and S_2 such that the true (hidden) examination probability of an item at position 4 is 0.8 for sessions in S_1 and 0.4 for sessions in S_2 respectively. Using Eq. 4.10, the probabilities of click at position 4 on a relevant item are 0.768 and 0.384 and on a non-relevant item are 0.13 and 0.065 for sessions in S_1 and S_2 respectively. The best AC can do is to estimate the examination probability by the average of $\frac{0.8+0.4}{2} = 0.6$, leading to inaccurate relevant probabilities for items in both sets of sessions. MBC, instead, separates the CTRs obtained from sessions. Given enough sessions from each set, a clustering method similar to what we use in MBC, can distinguish between relevant CTRs with mean $\in \{0.768, 0.384\}$ and non-relevant CTRs with mean $\in \{0.13, 0.065\}$. In other words, as long as the minimum relevant click probability is greater than the maximum non-relevant click probability, MBC manages to separate the two distributions.

4.5.4 Efficiency of MBC

We measured the running time of MBC and AC on multiple cores of Intel(R) Xeon(R) Gold 5118 CPU @2.30GHz. Here, we only report the time required for *correcting* clicks, since the LTR part is the same in both MBC and AC. MBC, requires around 110 seconds to estimate the mixture distributions and correct clicks. Each iteration of the rbEM parameter estimation for AC requires 370 seconds for fitting the regression function and around 50 additional seconds to update the bias parameters and target relevance probabilities. The fact that at least 30 iterations are required to get a decent performance of AC (see Section 4.6.2) means that AC requires a minimum of $(370+50)\cdot 30 = 12600$ seconds to correct clicks. This means that MBC runs approximately 114 times faster than AC.

Of course, the choice of the regression function plays an important role in the above computations. However, it is worth noting that even with a magical zero-time regression function, AC would still require around $50 \cdot 30 = 1500$ seconds for updating the bias parameters and target relevance probabilities. This hypothetical setting gives a lower bound of around 13 times for the efficiency superiority of our MBC method over AC.

4.5.5 MBC with Different Mixture Distributions

MBC relies on mixture models for correcting the clicks. In this section, we will address this question:

> *How do different assumptions for the distribution model of mixture components affect the correction quality?*

Particularly, to compare different distribution shapes, we execute two variations of MBC:

1. **MBC (Gaussian):** A Gaussian (normal) distribution is usually the default choice for modeling real world data, and due to the central limit theorem it is usually a safe choice.

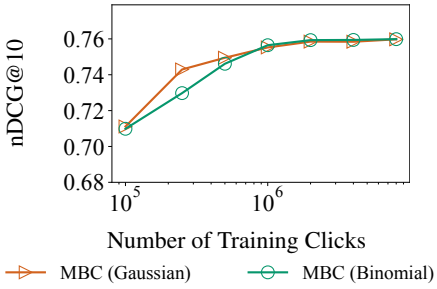


Figure 4.2: Ranking comparison of different mixture distributions for MBC in term of nDCG@10 with respect to different numbers of clicks on Yahoo! Webscope dataset.

2. **MBC (Binomial):** We include a Binomial distribution test, since the clicks are usually considered to have a Bernoulli distribution which makes CTR, follow a Binomial distribution.

Figure 4.2 shows the effect of the assumed distribution shape on the performance of MBC. These experiments coincide with the theory provided in Section 4.3. However, we observe that the Gaussian model converges faster than Binomial model. We hypothesize that, since the Binomial model is less generalizable than the Gaussian model, it cannot recover the signal in the presence of high levels of noise in the low data regime.

4.6 Analysis of Regression-based EM

In this section, we list and discuss about three specific practical limitations of rbEM with IPS and AC. As stated earlier, in AC and IPS there is an inherent cyclic dependency between relevance and bias parameters which leads to the use of iterative algorithms like EM. All of the limitations we list here come from the fact that standard EM cannot be used to infer the bias parameters and a regression function has to be used in the middle. Needless to say, these practical limitations do not apply to MBC.

4.6.1 Practical Limitations of rbEM for AC

Sensitivity to the choice of regression function. In rbEM with AC, the regression function is responsible for providing the relevance probabilities in the M-step. This means that the EM is no longer using the values obtained through the likelihood maximization, but an estimate of them obtained from the regression function. Consequently, the performance of EM greatly depends on the performance of the underlying regression function. We show empirically that different choices for the regression function lead to different performances in the ranking of the final unbiased LTR algorithm.

Not necessarily converging to zero gradient. Unlike standard EM, rbEM is not guaranteed to move in the direction of zero gradient. The reason is simple: in the M-step of rbEM, the relevance probabilities that maximize the likelihood function are replaced with the outputs of the regression function, in favor of addressing the otherwise

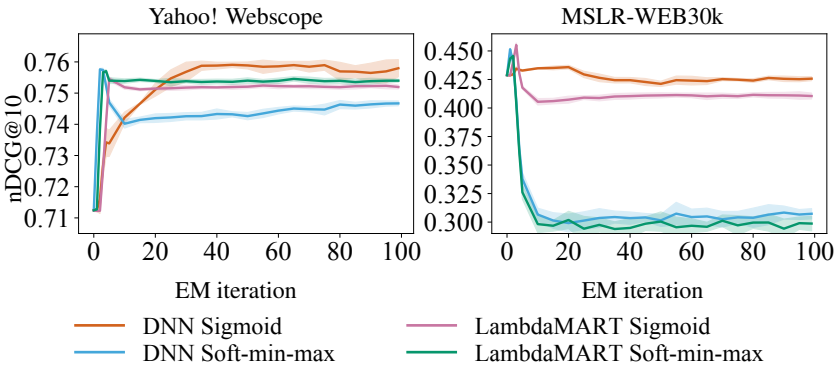


Figure 4.3: Regression-based EM ranking performance with different regression functions with respect to EM iterations. Left: Yahoo! Webscope dataset; right: MSLR-WEB30k dataset.

unavoidable sparsity issue. Therefore, the convergence proof of the standard EM, no longer holds for rbEM. Our observations suggest that when using the rbEM, more iterations do not necessarily lead to better performance, as opposed to the standard EM.

Low efficiency. The rbEM, by design, requires a regression function to be fitted to the relevance probabilities at each maximization step of the EM algorithm. We have discussed this issue in Section 4.5.4.

4.6.2 Instability of rbEM for AC

In practice, rbEM is used to estimate the bias parameters for AC. In this set of experiments, we address the following two questions about the performance of rbEM for AC:

1. *What is the impact of the choice of regression function for rbEM on the ranking performance of AC?*
2. *How does the ranking performance of rbEM AC vary as a function of the number of iterations?*

Neither of the above questions concern the standard EM, as discussed before. Introducing a regression function in an EM algorithm is a powerful idea to solve the issues of standard EM in CLTR, but it also brings its concerns as well.

We try different regression functions with different loss functions and report the ranking performance on different iterations of rbEM. We use the following regression functions: (i) LambdaMART; and (ii) a neural network similar to that of [122]. Since the regression function is used for fitting to relevance *probabilities*, we use the cross entropy loss. Following the literature, we test two cross entropy variations: (i) Sigmoid cross entropy, similar to [2, 131]; (ii) Soft-min-max cross entropy, similar to [122].

Figure 4.3 summarizes the ranking performance of AC with rbEM, with different regression functions, as a function of EM iterations. Based on the observations in this figure, the answers to the questions of this section are: *big* and *a lot*. Concerning the first question, we see that different regression functions lead to large differences in

ranking performance. More interestingly, the ordering of the regression functions is not preserved in different datasets.

The second question is about the change of the performance as the number of EM iterations increases. Figure 4.3 shows that the ranking performance does not always improve with more EM iterations. On the Yahoo! Webscope dataset, the DNN with sigmoid loss has a slight performance drop at iteration 80. On the MSLR-WEB30k dataset, the performance of DNN with sigmoid loss is decreasing between iterations 20 and 50. Another observation relating to the EM iterations are the sudden performance drops at some iterations: DNN with Soft-min-max loss in both datasets. These observations indicate that, unlike the standard EM, the rbEM cannot necessarily be trusted with regard to iterations: The performance is not always increasing with the number of iterations.

Based on the above discussions and according to Figure 4.3, our choice for the rbEM baseline for comparison with other methods is as follows: We chose the DNN with sigmoid cross entropy loss function as the regression function. When there are no anomalies, the DNN Sigmoid performs well up until iteration 100. In the cases where there is an anomaly, we use the results of the last iteration before the anomaly.

4.7 Conclusion

We have proposed a new correction method for position and trust bias, to be used in CLTR, and we have proven its unbiasedness. Our method, mixture-based correction (MBC), assumes that the distribution of CTRs of different items is a mixture of two distributions: relevant and non-relevant. Once this mixture is estimated, the relevant items are easily identified and can be used to train a LTR algorithm. Consequently, correction and learning to rank are two separate phases in our MBC method. This breaks the cyclic dependency between bias parameter estimation and relevance inference in existing correction methods, answering RQ3 positively. Unlike those methods, in which the unbiasedness relies on accurate bias parameters estimation, the unbiasedness proof of MBC does not rely on knowledge of relevance. This solves some of the practical limitations of existing methods for correcting position and trust bias which depend on the bias parameter estimation and usually use the rbEM for that.

Particularly, we have found that the cyclic dependency in the existing methods, leads to at least three practical limitations: (i) Severe sensitivity to the choice of the regression function; (ii) EM not necessarily converging towards the zero gradient; and, (iii) Low efficiency due to repeated use of the regression function. MBC is a new approach that solves all of these limitations as a side benefit.

We have performed extensive semi-synthetic experiments to analyze the strength of MBC at correcting click bias. Our experiments show that MBC outperforms AC, the state-of-the-art correction method for position and trust bias, in most of the settings. Furthermore, we have provided evidence that MBC is more robust to the click model mismatch than AC.

The main limitation of MBC is that it requires repeating sessions of each query with the same order of items, given by parameter n in Eq. (4.9). In contrast, unbiasedness of AC does not rely on repetition of sessions of each query.

This chapter concludes the first part of this thesis, concerning *bias* in counterfactual learning to rank (CLTR). Next, we will focus on *fairness* of ranking in Part II. In particular, Chapter 5 is about a new type of bias that directly affects fairness of ranking and requires attention on top of fairness of exposure. Finally, in Chapter 6 we will present a novel distribution representation that can be used for optimizing any fairness metric, both in deterministic and stochastic rankings.

Part II

Fairness

5

Fairness of Exposure Is Not Enough: Group Membership Bias

When learning to rank from user interactions, search and recommendation systems must address biases in user behavior to provide a high-quality ranking. In the previous three chapters, we re-examined three assumptions in this regard. In this chapter, we are going to address another type of bias in user interactions that has a direct impact on the fairness of exposure in addition to the ranking quality. The bias that we draw attention to here, is when sensitive attributes, such as gender, have an impact on a user's judgment about an item's utility. For example, in a search for an expertise area, some users may be biased towards clicking on male candidates over female candidates. We call this type of bias *group membership bias* or group bias for short.

Increasingly, we seek rankings that not only have high utility but are also fair to individuals and sensitive groups. Merit-based fairness measures rely on the estimated merit or utility of the items. With group bias, the utility of the sensitive groups is under-estimated, hence, without correcting for this bias, a supposedly fair ranking is not truly fair. Consequently, in this chapter we address RQ4, by first analyzing the impact of group bias on ranking quality as well as two well-known merit-based fairness metrics and show that group bias can hurt both ranking and fairness. Then, we provide a correction method for group bias that is based on the assumption that the utility score of items in different groups comes from the same distribution. This assumption has two potential issues of sparsity and equality-instead-of-equity; for both we use an amortized approach as a solution. We show that our correction method can consistently compensate for the negative impact of group bias on ranking quality and fairness metrics.

5.1 Introduction

Modern online search and recommender systems leverage user interaction data to enhance their ranking quality. When using human interactions, however, we need to account for human bias and the possibility of learning unfair ranking policies. In the

This chapter is currently under review as: A. Vardasbi, M. de Rijke, F. Diaz, and M. Dehghani. Group Membership Bias. arXiv preprint arXiv:2308.02887, 2023.

context of learning to rank (LTR), the term *bias* usually refers to unequal treatment of items with equal utility by users [57]. For example, position bias occurs when items at the top of a ranked list receive more clicks than those relevant lower down: higher items in a list absorb more *exposure*. Studies show that such a bias, if left uncorrected, degrades the ranking quality of a system trained on the user interactions [4, 58, 131]. This was the focus of Chapters 2, 3, and 4, as well. Consequently, a system should return rankings that strive to a certain extent for fairness of exposure. There are different definitions for fairness of exposure in ranking, leading to different metrics [32, 105, 114, 134], however, the core idea is the same: items with similar levels of utility should receive similar exposures by the system. Studies show that without meeting fairness of exposure, bias towards the privileged groups or individuals is reinforced, both in what the system learns from the ongoing interactions [39, 51], and in users' judgments about the utility of items [61, 118].

Group bias. The system can only ensure that items with similar utility receive comparable exposure to users by arranging them accordingly. However, this alone is insufficient. Studies show that users' judgments about the utility of items are affected by their perception of the item's group membership [61, 67, 118]. This means that, even when the exposure of two high-utility items from two different groups is the same, the users may judge them differently and one group may receive more clicks than the other. We refer to this behavior as *group membership bias* or *group bias* for short. Our study focuses on the scenario of two groups, where the term "affected" refers to the group whose items are prone to underestimation and receive fewer clicks than they ideally should. In this chapter, we answer the following question:

RQ4 What is the impact of group bias on the quality and fairness metrics in a ranking and how to correct for this bias, without substituting equality for equity?

Theoretical and experimental analysis. We analyze the impact of group bias on ranking quality and merit-based fairness of exposure metrics. We consider clicks as the primary measure of user interactions, which are used in two ways: (i) For head queries, clicks are memorized and the ranking is performed via tabular search. In this case, the training labels are directly used to generate the ranking presented to the user, and user preferences can often be obtained with high accuracy [97]. (ii) For tail queries the clicks are used as supervision signals to train an LTR model. In this case, the outputs of the LTR model are used to generate the ranking that is shown to the user. With that in mind, we provide a theoretical analysis of the impact of group bias on various metrics over the training labels. This gives us an understanding of the impact of group bias on the head queries directly, and on the tail queries indirectly. Theoretically analyzing the output of a LTR model involves considering the architecture and loss function of the LTR model, which is beyond the scope of this chapter. For the experimental part, however, we analyze the impact of group bias both on the training labels as well as the outputs of an LTR model.

Impact on ranking. Previous work on implicit bias [23, 64] has shown that similar to other types of bias, implicit bias can degrade the ranking quality of systems. In this work, we add to their theoretical results by quantifying this degradation with an approximation formula for the normalized discounted cumulative gain (NDCG) metric. Furthermore, we provide an important part that is missing in previous work, i.e., an

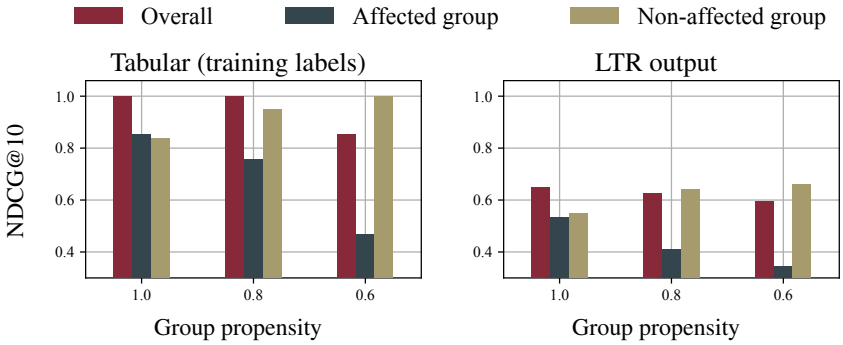


Figure 5.1: The impact of group bias on ranking performance for the Yahoo! dataset.

experimental analysis of the impact of group bias on the outputs of an LTR model. Figure 5.1 shows an example of the impact of group bias on the ranking performance, measured by NDCG. In these plots, the bars associated with the "(non-) affected group" label show the NDCG@10 when *only* the relevant items from the (non-) affected group are considered as relevant. Furthermore, lower group propensities mean more severe group bias. On the left, we observe the impact on the training labels, i.e., tabular search, while the right plot shows the impact on the outputs of a general LTR model. In both plots, we observe that the affected group is hurt by the group bias, while the other group has gained. Importantly, in both regimes, the overall ranking quality is degraded by increasing the group bias.

Impact on fairness. Unlike other types of bias that may affect fairness indirectly, group bias has a direct impact on fairness: Clicks suffering from group bias can lead the system to undervalue the utility scores of a particular group (see, e.g., Figure 5.1). Consequently, when the expected exposure is assigned to groups based on these biased estimates of the utility, the ranking may not be *truly* fair. For our analyses, we consider two widely used metrics for the fairness of exposure, namely disparate treatment ratio (DTR) [114] and expected exposure loss (EEL) [15, 32]. Each metric has a definition for the *ideal expected exposure* in terms of the utility, that leads to the fairest ranking. Distinguishing between the true (unbiased) utility and observed (biased) utility, we provide formulas for the change in the true fairness metrics, when the target expected exposure is obtained from the biased utility. Figure 5.2 shows an example of the impact of group bias on the DTR and EEL fairness metrics. DTR (left plot) is a multiplicative metric, so we measure the ratio between the target exposure of the affected group, to the target exposure of the non-affected group.[1] We then normalize this ratio with the ratio obtained from the full information case, i.e., using true utilities without group bias. A DTR score of 1 means the fairest exposure. EEL (right plot) is an additive metric, so we measure the ℓ_2 loss between the target exposure vector in the biased case and the full information case. For EEL, a loss of 0 means the fairest exposure. In both metrics, we observe that group bias leads to noticeable deviations from the full information case.

Correction. To correct for group bias, it should first be modeled. We follow previous

[1]We follow [114] and always keep the ratio below unity.

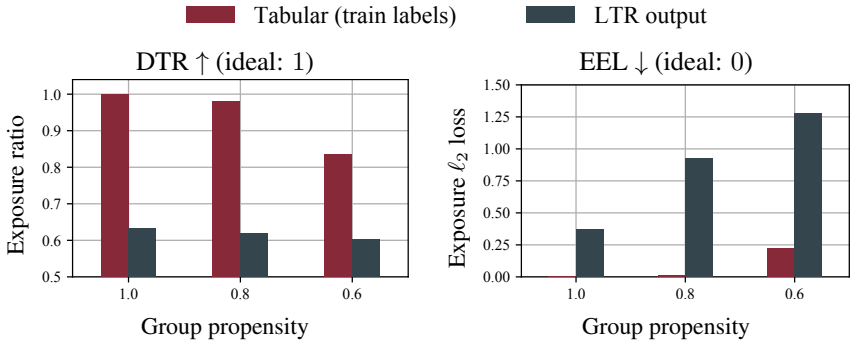


Figure 5.2: The impact of group bias on fairness metrics for the Yahoo! dataset.

work on implicit bias [64] and model group bias with a multiplicative factor. This allows us to use the inverse propensity scoring (IPS) method to correct for the bias [58, 131]. Measuring group bias, however, is not as simple as measuring position or trust bias. We argue that group bias measurement requires some assumptions on the distribution of the true utility scores of different groups. Following [23, 38, 64], one can assume that the true utility scores of both groups come from the same distribution. Naively assuming that the utility scores for each query should come from the same distribution, leads to a trivial solution with equality instead of equity. In other words, *equity* is based on the premise that exposure should be distributed based on utility, i.e., merit-based fairness. Assuming that the utility of different groups is equal for each query, means that different groups should receive equal exposure all the time, which means equality. To counter this, we propose to aggregate the utility scores of items across all the queries with similar expected group propensity and measure the group bias parameter over this aggregated set of scores. We show that our correction method based on the above amortized measurement of the bias parameter is effective for restoring both the ranking quality and fairness metrics.

5.2 Related Work

There is an increasing number of studies indicating the existence of group bias. Implicit bias, a special case of group bias, in which the preference of one group over the other is unintentional, has been widely studied in human behavior studies [e.g., 20, 41, 59]. More recently, implicit bias has been formalized in the set selection problem [64] and extended to the ranking scenario [23, 38].

Here, we list a small number of example studies indicating that group membership affects users' judgment. Studies by Lyness and Heilman [83] on performance evaluations and promotions of managers indicate that standards for women's promotion were stricter, e.g., women had to show roughly twice as much evidence of competence as men to be seen as equally competent. In [61], it is observed in a user study that in a career search, results that are consistent with stereotypes for a career are rated higher. Sühr et al. [118] pose the important question of whether "fair ranking improve[s] minority

outcomes?" and arrive at the result that persistent gender preferences of employers can limit the effectiveness of fair ranking algorithms, and that fair ranking is more effective when the features of an underrepresented candidate are similar to the average overrepresented group features. Recently, Krieg et al. [67] in their user study on gender sensitive queries from [66] show that perceived gender bias affects judgment. Vlasceanu and Amodio [129] conduct two user studies and observe that societal and algorithmic gender bias affect each other: the algorithmic outputs of search engines track pre-existing societal-level gender biases; and, at the same time, exposure of users to these biased results shape users' cognitive concepts and decisions.

All of the above examples confirm that group bias exists in user interactions. In this chapter, we study its impact on ranking and fairness measures and propose a method to correct for it. Closest to this chapter is the work by Celis et al. [23] who show that implicit bias degrades ranking quality and that by ensuring equality of exposure, the ranking quality can be improved. What we add on top of this work is to provide a formalization of the change of ranking and merit-based fairness metrics as a result of group bias. We also provide extensive experimental analyses of the impact of group bias on the output of an LTR model.

The idea of our amortized correction to counter sparsity and equality-instead-of-equity has similarities to the notion of amortized fairness of exposure [13], where the exposures and utilities of individuals (or groups) are aggregated across multiple queries and the fairness metric is calculated according to the aggregated exposure and utility. This corresponds to fairness evaluation. In contrast, we aggregate the items associated with multiple queries to find the group bias parameter that minimizes the distance between the utility distribution of the affected and non-affected groups. This corresponds to group bias correction.

Remark 5.1. Our terminology of *group membership bias* should not be confused with the *in-group bias*, where a user favors members from their own group over out of the group members [91, 139]. In this study, we follow a body of previous work on implicit and explicit bias and only focus on the group membership of the items and do not consider the group of the users. Consequently, the issues related to in-group bias, such as loyalty versus neutrality, are out of the scope of this chapter.

5.3 Group Membership Bias

5.3.1 Definitions

As discussed in Section 5.2, prior work shows that the judgment of a user about the relevance of an item may be affected by the item's group. Either unconsciously (as in implicit bias [38, 64]) or due to stereotypical bias [61, 67], users tend to rate one group higher than the other. In this chapter, we do not aim to deal with the source of this biased behavior and only focus on its impact on algorithms and metrics. We call this behavior the *group membership bias*. Following the well-known examination hypothesis [29] that says: An item is clicked by a user if it is (i) examined; and (ii) found attractive by that user, one can attribute group bias to the attractiveness part:

$$P(A \mid q, d, g) = f\big(P(R \mid q, d), g\big), \tag{5.1}$$

where A, R, q and d stand for attractiveness, relevance, query and document, respectively, and g is the group of which d is a member. Eq. (5.1) states that the attraction of an item to the user not only depends on the item's true relevance to the query, but is also a function of the item's group. Following the literature on implicit bias [64], we assume this dependency to have a multiplicative form as follows:

$$P(A \mid q, d, g) = \beta_g \cdot P(R \mid q, d). \tag{5.2}$$

We call β_g the *group propensity*. Notice that β_g is not necessarily fixed across all queries.

Remark 5.2. Here we consider one sensitive attribute for simplicity of notation. Extending our discussions to more attributes with intersectional groups is possible using the formulation in [23, 89].

5.3.2 Ranking Regimes

Here, we distinguish between two LTR regimes: (i) tabular search for head queries; and (ii) general LTR model for tail queries. Note that the majority of previous studies focused only on the general LTR regime [e.g., 92, 115], or the tabular regime [e.g., 13, 114]. In contrast, in this chapter and Chapter 6, we consider both LTR regimes.

Tabular search for head queries. In tabular search, users' historical interactions with the head queries are directly used to estimate items' utilities [60, 69–71, 74, 104, 143]. In this regime, we assume that $P(A)$ can accurately be inferred from clicks: other types of bias such as position and trust bias are corrected for and only group bias remains in the signals. Our theoretical results on the impact of group bias on different metrics, lie in this regime.

General LTR model for tail queries. For new queries and ones that a tabular model is not confident about, an LTR model is used. We assume that this LTR model is trained over the accurate estimations of $P(A)$ from the head queries. Writing $r_{q,d}$ for the relevance of item d to query q, ranking metrics per query can usually be expressed in the following form:

$$\Delta_q = \sum_d \lambda_{q,d} \cdot r_{q,d}, \tag{5.3}$$

where λ is a metric-specific coefficient, e.g., for discounted cumulative gain (DCG) we have $\lambda^{DCG}_{q,d} = -\log\big(1 + \text{rank}(d)\big)^{-1}$. Using attractiveness instead of the true relevance to train an LTR model, means that instead of Δ_q, the following metric is being optimized:

$$\hat{\Delta}_q = \sum_d \lambda_{q,d} \cdot P(A \mid q, d) \overset{(5.2)}{=} \sum_g \beta_g \sum_{d \in g} \lambda_{q,d} \cdot P(R \mid q, d). \tag{5.4}$$

Comparing equations (5.3) and (5.4), it is easy to see that $\hat{\Delta}_q$ is biased:

$$\mathbb{E}_R\left[\Delta_q\right] = \sum_d \lambda_{q,d} \cdot P(R \mid q, d) \neq \hat{\Delta}_q, \tag{5.5}$$

unless β_g is equal across all groups, i.e., there is no group bias. Similar to studies on position and trust bias [2, 58, 121–123], in our experiments, we analyze the effect of

group bias on the ranking quality of the LTR model (RQ1). Due to the relationship between group bias and fairness concerns, we go one step further and assess how leaving the group bias uncorrected affects the optimization of fairness metrics.

5.4 Theoretical Results

We assume that there are two groups $G_{\mathcal{A}}$ (affected) and $G_{\mathcal{N}}$ (non-affected), with group propensities $\beta_{\mathcal{A}} < 1$ and $\beta_{\mathcal{N}} = 1$. We further assume binary relevance, i.e., $r \in \{0, 1\}$, and assume that within each group, attractiveness is monotone with respect to relevance:

$$\forall d, d' \in G_i, \text{ if } r_d = 1 \;\&\; r_{d'} = 0, \text{ then } P(A \mid q, d) > P(A \mid q, d'). \tag{5.6}$$

For brevity, we write a_d for the attractiveness probability of item d, assuming that there is no confusion about the query. Let the number of candidate items for a query be n, out of which $n_{\mathcal{A}}$ and $n_{\mathcal{N}}$ items belong to groups $G_{\mathcal{A}}$ and $G_{\mathcal{N}}$, respectively. We indicate the number of relevant items with $n^+_{\mathcal{A}}$ and $n^+_{\mathcal{N}}$. To assess the impact of group bias on different metrics, we measure the change in the target metric when the observable attractiveness probabilities are considered as a proxy for the true relevance scores.

5.4.1 Ranking Quality

For ranking quality, we calculate the NDCG of the list obtained from sorting items based on their attractiveness probability and measure its deviance from the ideal NDCG, i.e., 1. By definition, group bias affects the attractiveness probabilities for one group (here $G_{\mathcal{A}}$). Let a^* be the minimum attractiveness value for the relevant items of group $G_{\mathcal{N}}$:

$$a^* = \min_{d \in G_{\mathcal{N}}} \{a_d \mid r_d = 1\}. \tag{5.7}$$

Items of $G_{\mathcal{A}}$ with higher attractiveness values than a^*, are ranked correctly with probability 1: Group bias has dampened their attractiveness probabilities, but still none of the non-relevant items is ranked higher than them. We define an auxiliary random variable ν to be the fraction of relevant items from the affected group $G_{\mathcal{A}}$ that are ranked correctly with probability 1:

$$\nu = \frac{|\{d \mid d \in G_{\mathcal{A}} \;\&\; r_d = 1 \;\&\; a_d > a^*\}|}{|\{d \mid d \in G_{\mathcal{A}} \;\&\; r_d = 1\}|}. \tag{5.8}$$

For uniformly distributed scores in the interval of $[0, 1]$, the mean value of ν has a closed form as follows:

$$\mathbb{E}[\nu] = \begin{cases} 2 - \frac{1}{\beta_{\mathcal{A}}}, & \text{if } \beta_{\mathcal{A}} > 0.5 \\ 0, & \text{otherwise.} \end{cases} \tag{5.9}$$

Theorem 5.1. *In the presence of group bias, for uniformly distributed attractiveness scores, the change in the NDCG of the list, sorted based on the items attractiveness, can be approximated by a linear function of $\mathbb{E}[\nu]$, i.e., the fraction of affected relevant items that are still as attractive as the non-affected relevant items.*

Proof. Our monotonicity assumption of the within-group attractiveness, Eq. (5.6), ensures that no relevant item is ranked lower than non-relevant items of $G_{\mathcal{A}}$. This means that the $1-\nu$ fraction of the affected relevant items lies somewhere between $n^+_{\mathcal{N}}+\nu n^+_{\mathcal{A}}$ and $n_{\mathcal{N}}+n^+_{\mathcal{A}}$ positions. The expected DCG of the list would be as follows:

$$\mathbb{E}\left[DCG\right] = \sum_{i=1}^{n^+_{\mathcal{N}}+\nu n^+_{\mathcal{A}}} \frac{1}{\log(1+i)} + \sum_{i=n^+_{\mathcal{N}}+\nu n^+_{\mathcal{A}}+1}^{n_{\mathcal{N}}+n^+_{\mathcal{A}}} \frac{\xi_i}{\log(1+i)}, \tag{5.10}$$

where ξ_i depends on the distribution of the attractiveness scores. For a uniform distribution, we have:

$$\xi_i = \frac{(1-\nu)n^+_{\mathcal{A}}}{n_{\mathcal{N}} - n^+_{\mathcal{N}} + (1-\nu)n^+_{\mathcal{A}}}. \tag{5.11}$$

Finally, using numerical analysis to approximate the average DCG in Eq. (5.10) by a linear function of ν, leads to a small approximation error, e.g., a relative error of at most 5% in a top-20 setup. □

5.4.2 Merit-Based Fairness Metrics

Next, to see the impact of group bias on fairness metrics, we analyze two well-known merit-based fairness of exposure metrics, namely EEL and DTR. For both metrics, we calculate the target exposure in two cases: the full information case where the true relevance scores are used to compute the target exposure; and, the group biased case where the attractiveness probabilities are used as proxies for relevance to compute the target exposure. By *change in true target exposure* we mean the difference between the above two cases.

EEL

In the next theorem, we calculate the change in the target exposure of $G_{\mathcal{N}}$ as a result of group bias.

Theorem 5.2. *In the presence of group bias, assuming the position-based model (PBM) user browsing model with logarithmic decay of exposure as in DCG,*[2] *the change in the target exposure of EEL can be approximated as follows:*

$$\Delta(\textit{EEL}) = c \cdot \log\left(\frac{\textit{\# True relevant items}}{\textit{\# Perceived relevant items}}\right), \tag{5.12}$$

where c is a constant depending on $n^+_{\mathcal{N}}$, $n_{\mathcal{N}}$, and $n^+_{\mathcal{N}}$.

Proof. As we are working with two groups, and the sum of the group exposures is fixed, to measure the change in the target exposure vector, it is sufficient to measure the change in the target exposure of one group and multiply it by 2.

[2]Here we follow [37, 125] for this assumption. Similar analysis can be performed for other exposure models.

To compute the expected exposure for EEL, the utility values should be discrete. With a slight abuse of notation, we assume that a^* (instead of Eq. (5.7)) is the threshold used for discretization of the attractiveness probabilities,[3] and we use $\bar{a}_d$ for the discretized value of a_d. Since $\beta_{\mathcal{N}} = 1$, we assume that $\bar{a}_d = r_d$ for $d \in G_{\mathcal{N}}$. However, for the affected items, because of $\beta_{\mathcal{A}} < 1$, not all the scores are necessarily correct. We re-use ν from Eq. (5.8) to show the fraction of affected relevant items that are still recognized as relevant.

For the average exposure of the relevant and non-relevant items we use the following two approximations:

$$\frac{1}{m}\sum_{i=1}^{m}\frac{1}{\log(1+i)} \approx \alpha\log(m) + c \tag{5.13}$$

$$\frac{1}{n-m}\sum_{i=m}^{n}\frac{1}{\log(1+i)} \approx \alpha'\log(m) + c', \tag{5.14}$$

where α and α' are constants, obtained by numerical analysis. For example, for $n = 20$, $\alpha = -0.146$ and $\alpha' = -0.022$ lead to relative approximation errors of at most 5%. In the full information case, there are a total of $n^+_{\mathcal{N}}+n^+_{\mathcal{A}}$ relevant items, i.e., $m = n^+_{\mathcal{N}}+n^+_{\mathcal{A}}$ in Eq. (5.13) and (5.14). But with group bias, only $m = n^+_{\mathcal{N}} + \nu n^+_{\mathcal{A}}$ of the items are recognized as relevant. Consequently, the change in the target exposure as a result of group bias can be approximated as follows:

$$\Delta(\text{EEL}) = 2\big(\alpha n^+_{\mathcal{N}} + \alpha'(n_{\mathcal{N}} - n^+_{\mathcal{N}})\big)\log\left(\frac{n^+_{\mathcal{N}} + n^+_{\mathcal{A}}}{n^+_{\mathcal{N}} + \nu n^+_{\mathcal{A}}}\right). \qquad \square$$

DTR

DTR is a multiplicative metric. To have a meaningful measure for the change in DTR in the presence of group bias, one has to compute the ratio of the target expected exposure in the full information (E) and group-biased ($\tilde{E}$) settings.

Theorem 5.3. *In the presence of group bias, the change in the target exposure of DTR, equals the fraction of affected relevant items that are still as attractive as the non-affected relevant items.*

Proof. Using the same notation as in the previous sections, and noting that because of the binary relevance assumption, the utility of each group is equal to the number of its relevant items, this ratio is computed as follows:

$$\rho(\text{DTR}) = \frac{\tilde{E}_{\mathcal{A}}}{\tilde{E}_{\mathcal{N}}}\cdot\frac{E_{\mathcal{N}}}{E_{\mathcal{A}}} = \frac{\nu n^+_{\mathcal{N}} + \nu n^+_{\mathcal{A}}}{n^+_{\mathcal{N}} + n^+_{\mathcal{A}}} = \nu. \qquad \square$$

5.5 Group Bias Correction

Our multiplicative formulation of group bias in Eq. (5.2) allows us to use IPS to correct for group bias, once we know the value of the propensity β. The unbiasedness proof of

[3] Usually $a^* = 0.5$ is the least controversial threshold.

IPS for this case is exactly the same as that of position bias in [58, 131]. However, similar to position bias, the unbiasedness proof depends entirely on the accurate estimation of the bias parameter [123].

Unlike position bias, group bias cannot be measured by intervening in the ranked list of items. The reason is that the bias attribute in position bias can be changed without modifying the content of the items: Each item can be shown in different positions, hence, detaching propensity from relevance. In contrast, for group bias, the bias attribute, i.e., group membership, is a characteristic of the item that *cannot* be changed. As such, users' interactions with items cannot be measured for different values of the bias attribute.

Instead, to measure group bias, previous work on implicit bias (with the same problem formulation as Eq. (5.2)), assumes that the utility scores of different groups come from the same distribution [23, 38, 64]. Here, we use the same assumption, but extend it to an amortized criterion.

5.5.1 Measurement

Let $\mathbf{A}_{\mathcal{A}}$ and $\mathbf{A}_{\mathcal{N}}$ be the set of (observed) attractiveness scores, and $\mathbf{R}_{\mathcal{A}}$ and $\mathbf{R}_{\mathcal{N}}$ be the set of (latent) relevance scores of $G_{\mathcal{A}}$ and $G_{\mathcal{N}}$, respectively. Let $\Delta_{\mathbb{D}}$ be a non-parametric test for the equality of one-dimensional probability distributions such as the Kolmogorov-Smirnov (KS) [85] test. The assumption that the utility scores of the two groups come from the same distribution means that:

$$\lim_{|\mathbf{R}_{\mathcal{A}}|,|\mathbf{R}_{\mathcal{N}}|\to\infty} \Delta_{\mathbb{D}}\left(\mathbf{R}_{\mathcal{A}}, \mathbf{R}_{\mathcal{N}}\right) = 0. \tag{5.15}$$

Assuming Eq. (5.2) to be the relation between $\mathbf{A}_{\mathcal{A}}$ and $\mathbf{R}_{\mathcal{A}}$, the best estimation of $\beta_{\mathcal{A}}$ is given by the following optimization problem:

$$\hat{\beta}_{\mathcal{A}} = \operatorname*{argmin}_{\beta_{\mathcal{A}}} \Delta_{\mathbb{D}}\left(\frac{\mathbf{A}_{\mathcal{A}}}{\beta_{\mathcal{A}}}, \mathbf{A}_{\mathcal{N}}\right), \tag{5.16}$$

where $\mathbf{A}_{\mathcal{A}}/\beta_{\mathcal{A}}$ is the set obtained by dividing all the scores in $\mathbf{A}_{\mathcal{A}}$ by $\beta_{\mathcal{A}}$. In our experiments, we choose the KS test for $\Delta_{\mathbb{D}}$ and use grid search to solve the one-dimensional optimization of Eq. (5.16).

It only remains to define how $\mathbf{A}_{\mathcal{A}}$ and $\mathbf{A}_{\mathcal{N}}$ sets should be constructed. Naively constructing these sets per query has two issues: (i) *Sparsity*: Usually, we do not have a large number of items with non-zero exposure, associated with one query in real-world search engines. On the other hand, statistical tests measuring the distance between probability distributions work best with large numbers of data points. This means that considering the items of one query in Eq. (5.16) may not lead to reliable solutions due to high variance. (ii) *Equality-instead-of-equity*: Assuming the same distribution for the utility of different groups can make the notion of equity meaningless, as the implicit assumption in merit-based fairness metrics is that different groups may not necessarily have the same utility. Correcting the utility estimations in such a way that the utility scores of different groups are forced to have close distributions makes the target exposure of different groups almost equal.

Remark 5.3. Our assumption that the utility scores come from the same distribution comes from the principle of maximum entropy: unless there are explicit and justified

reasons indicating that different groups have different utility score distributions, it is only reasonable to assume the same distribution. Prior work on implicit bias [23, 38, 64] is based on this same assumption.

5.5.2 Amortized Correction

Instead of using Eq. (5.16) per-query, in the amortized correction method, we first aggregate the items across queries with (almost) the same group propensity. The sets $\mathbf{A}_{\mathcal{A}}$ and $\mathbf{A}_{\mathcal{N}}$ contain the attractiveness scores of these aggregated items. This aggregation addresses both issues mentioned in Section 5.5.1: (i) With multiple queries, the size of the sets $\mathbf{A}_{\mathcal{A}}$ and $\mathbf{A}_{\mathcal{N}}$ grows, reducing the variance. (ii) Amortized equality does not force per-query equality. As an example, assume there are two queries with the same group propensity 0.8, each with two items from different groups. The relevance probabilities of the items from the affected group and non-affected group are 0.5 and 1 for the first query, and 1 and 0.5 for the second, respectively. In the per-query approach, two different propensities are inferred for the two queries to make the corrected utilities of the two groups equal in each query, leading to equality of exposure. However, in the amortized approach, we have $\mathbf{A}_{\mathcal{A}} = \{0.4, 0.8\}$ and $\mathbf{A}_{\mathcal{N}} = \{1, 0.5\}$. Solving Eq. (5.16) gives us a single value for β. In this case, groups will have different utility scores after correction, and equity of exposure has not been replaced by equality.

The amortized correction, however, introduces a new challenge: How to detect queries with almost the same group propensity, before measuring their group propensity? One way to break this cyclic dependency is by using extra knowledge. Notice that in order to detect queries with almost the same group propensity, it is only required to have a clustering of queries. Previous work shows that such a clustering exists for a number of group attributes such as gender [66]. In this chapter, we assume that there exists a given clustering of queries into clusters with almost the same group propensity. After confirming the effectiveness of our amortized correction method, we further show in our experiments that even loosely clustering the queries when an accurate and more specific clustering is not available, improves the ranking quality and fairness metrics over the naive case of not correcting for group bias.

5.5.3 Upshot

We relied on prior studies for the existence of group bias in user interactions and provided theoretical results about its impact on the ranking and merit-based fairness metrics. Then, we proposed an amortized correction method for group bias. Next, we test our theoretical findings and arguments experimentally.

5.6 Experimental Results

We investigate the following questions regarding group bias in our experiments: (i) Is the impact of group bias on degrading the ranking quality and fairness metrics consistent for different sensitive attributes and in different datasets? (ii) Can our correction method effectively correct for group bias? (iii) How does the amortized approach compare to

the per-query approach for correction? (iv) How robust is our correction method to the accuracy of clustering the queries based on their group propensity?

5.6.1 Setup

Dataset. We use four datasets with provided sensitive attributes and two with synthesized sensitive attributes. (i) **IIT-JEE**: The dataset comprises the scores of candidates who took the Indian Institutes of Technology Joint Entrance Exam (IIT-JEE) in 2009. This information was made public in June 2009, following a Right to Information request [68]. It contains the scores of about $385k$ students, the student's gender ($98k$ women and $287k$ men), their birth category (see [10]), and zip code. This dataset was used in prior work on implicit bias [e.g., 23]. We normalize the scores to the $[0, 1]$ interval using min-max normalization. Furthermore, we simulate queries by grouping the students based on their birth category and zip code. This gives $48.6k$ queries, among which we only keep the ones with both genders and at least one normalized score above 0.5 and one below 0.5. The filtering gives us $2.9k$ queries with a total of $205k$ scores. (ii–iii) **TREC 2019 and 2020**: The academic search dataset provided by the TREC Fair Ranking track 2019 and 2020 [14]. These datasets come with 632 and 200 train queries, respectively, with an average of 6.7 and 23.5 documents per query. Following [111, 125], we divide the items (i.e., papers) into two groups based on their authors' h-index. (iv) **MovieLens** $1M$: The classic movie recommendation dataset comprising $1M$ movie ratings that were provided by $6k$ users for $3.9k$ different movies. We scraped IMDB to obtain the country of origin and box office cumulative worldwide gross values for each item (i.e. movie). For the sensitive attributes, we consider two groupings as follows. In MovieLens$_{[Co.]}$, we divide the movies based on their first listed country of origin into United States (US) and non-US groups with $2.7k$ and $1.2k$ movies and $807k$ and $193k$ ratings, respectively. In MovieLens$_{[BO]}$, we divide the movies based on their box office with a threshold of $100M\$$ into high and low-grossing groups with 388 and $3.5k$ movies and $324k$ and $676k$ ratings, respectively.

LTR dataset. To analyze the impact of group bias in the general LTR regime, following prior work on unbiased LTR research [55, 58, 122, 123], we use two well-known LTR datasets: Yahoo! Webscope [25] and MSLR-WEB30k [102]. The Yahoo! and MSLR datasets are represented by query-document feature vectors of lengths 501 and 131, respectively, and both have graded relevance labels from 0 to 4. For our experiments on head queries and a tabular regime, we use the training set of the Yahoo! and MSLR datasets, with 20k queries and 473k documents and 19k queries and 2.2M documents, respectively. The Yahoo! dataset contains 6.7k test queries and 163k test documents; MSLR contains 6k and 749k queries and documents in its test set. Test queries are considered tail queries in our experiments and we analyze the impact of group bias as well as the effectiveness of our correction method on the LTR outputs over these queries.

Sensitive attribute for LTR datasets. We extend prior work [32, 125, 135] and utilize a data-driven approach for selecting features as sensitive attributes and dividing items into two groups based on some threshold on that feature. Our criterion for selecting a feature as a sensitive attribute is as follows: For each feature we divide the items in two groups based on a threshold equal to the mean minus one standard deviation of that

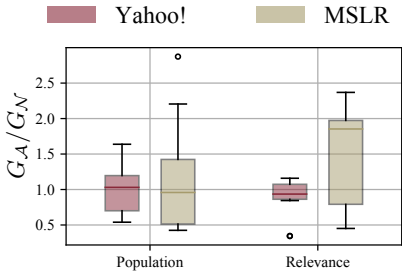


Figure 5.3: Ratio of affected to non-affected group members in terms of population and average utility score (relevance) for different sensitive attributes in Yahoo! and MSLR datasets.

feature. If more than 95% of queries have at least one item from both groups, we select the feature as a candidate for sensitive attribute. Based on this criterion, we have selected features $[5, 88, 100, 141, 155, 264, 393, 426]$ and $[11, 14, 15, 126, 127, 130, 131, 132]$ from the Yahoo! and MSLR datasets.[4] Figure 5.3 gives an overview of the ratio of affected to non-affected group members in terms of population and average utility score. In what follows we use, e.g., Yahoo!$_{[426]}$ for the Yahoo! dataset with feature number 426 as the sensitive feature. For each feature, we assume two groupings based on two thresholds: (i) mean value; and (ii) mean minus one standard deviation. This gives us a total of 32 different setups.

Bias simulation. To simulate group bias, we use Eq. (5.2), but to make the simulation more realistic, we add a normal noise to the $\beta_{\mathcal{A}}$ value for each query. We experiment with two propensities $\beta_{\mathcal{A}} \in \{0.6, 0.8\}$ and use $\sigma_{\beta} = 0.1$ for the standard deviation of the normal noise. In Section 5.6.4, to add to the uncertainty of the setup, we also experiment with higher noise variances of $\sigma_{\beta} \in \{0.2, 0.3\}$. For our correction method, we found that adding a amount of small noise to the scores for breaking the ties, without swapping the order of the grades, helps to have a smoother curve for β in Eq. (5.16).

LTR model. For the general LTR model (for tail queries) we use a neural network with attention and LambdaRank Loss as in [101].

5.6.2 Impact of Group Bias

First, we show that group bias, on both tabular and LTR regimes, consistently has a negative impact on the ranking quality and fairness metrics. To do so, we run experiments on two datasets, namely Yahoo! and MSLR, each with 8 different features as the sensitive attribute, and two different thresholds for separating the groups (see the "Sensitive attribute" paragraph on Section 5.6.1). This gives us a total of 32 different setups. For each setup, we simulate the attractiveness probabilities with $\beta_{\mathcal{A}} \in \{0.8, 0.6\}$[5] and compare NDCG@10, DTR and EEL metrics against the full

[4] Feature numbers start from 1.

[5] We report results for these two values as *mild* and *severe* cases of group bias. Our experiments with other values led to monotone results.

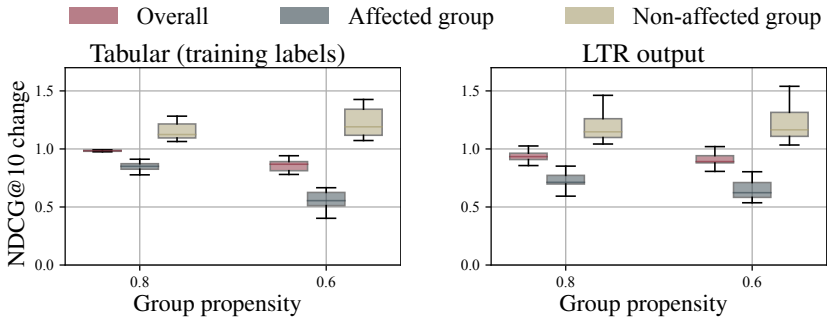


Figure 5.4: The impact of group bias on ranking quality for the Yahoo! and MSLR datasets with different sensitive attributes.

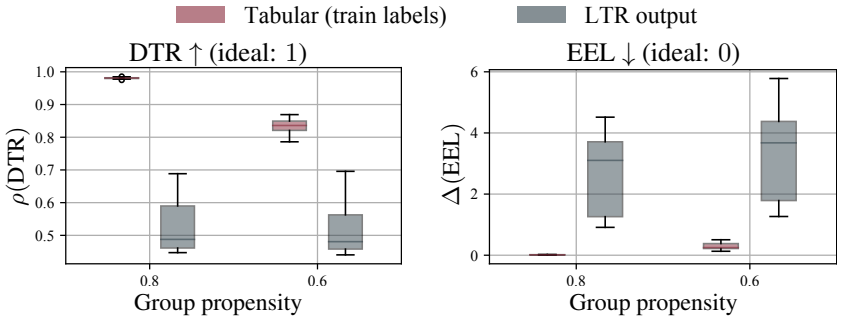


Figure 5.5: The impact of group bias on fairness metrics for the Yahoo! and MSLR datasets with different sensitive attributes.

information case.

Figure 5.4 shows a summary of the impact of group bias on the ranking performance of both tabular and LTR regimes. These results show that the observation of Figure 5.1 on one specific sensitive attribute is consistent across different datasets and other sensitive attributes: Group bias degrades the ranking quality of the affected group in the tabular regime and this damage is also reflected in the LTR output, trained over the biased training labels. As a result of pushing down the relevant members of the affected group, the non-affected group gains ranking quality, i.e., the NDCG of the non-affected group with group bias is higher than the full information case. However, the overall ranking is worsened with group bias. Comparing the tabular (left) and LTR output (right) plots in Figure 5.4, we observe that increasing the severity of group bias from 0.8 to 0.6, affects the tabular regime more. This may be because unlike on the tabular regime, the impact of group bias on the LTR outputs is indirect.

Figure 5.5 shows a summary of the change in fairness metrics of both tabular and LTR regimes as a result of group bias. For example, a value of $\rho(\text{DTR}) = 0.5$ in the left plot means that on average, the target (i.e., ideal) exposure computed by the biased attractiveness scores differs from the true target exposure computed in the full

Table 5.1: The impact of our amortized group bias correction on ranking and fairness metrics in the tabular regime, under mild group bias ($\beta = 0.8$). $\hat{\beta}_{\mathcal{A}}$ shows the estimated value of the bias parameter as in Eq. (5.16). For each metric, the columns "*B*" and "C" show the "*Biased*" and "Corrected" performances, respectively. Superscripts * indicate a significant improvement over the biased case with $p < 0.001$.

	$\hat{\beta}_{\mathcal{A}}$	NDCG@10 ↑		ρ(DTR) ↑		Δ(EEL) ↓	
		B	C	*B*	C	*B*	C
Yahoo!$_{[426]}$	0.825	*0.987*	0.996*	*0.820*	0.955*	*0.447*	0.120*
MSLR$_{[127]}$	0.843	*0.975*	0.991*	*0.813*	0.948*	*1.687*	0.308*
IIT-JEE	0.727	*0.989*	0.991	*0.799*	0.906*	*0.504*	0.341*
MovieLens$_{[Co.]}$	0.822	*1.000*	1.000	*0.800*	0.962*	*1.101*	0.513*
MovieLens$_{[BO]}$	0.781	*1.000*	1.000	*0.799*	0.974*	*2.330*	0.895*
TREC 2019	0.838	*0.997*	1.000	*0.888*	0.954*	*0.041*	0.020*
TREC 2020	0.821	*0.995*	0.999	*0.815*	0.954*	*0.356*	0.114*

information case by a factor of 0.5. Similarly, a value of $\Delta(\text{EEL}) = 3$ in the right plot means that on average, the target exposure of the biased case has an ℓ_2 distance of 3 to the full information target exposure. These results show that the observation of Figure 5.2 on one specific sensitive attribute is consistent across different datasets and other sensitive attributes: Group bias changes the target exposure in the tabular regime and this change is reflected in the LTR output, trained over the biased training labels. Consequently, when the system distributes the exposure according to the target exposure to make a ranking fair, if the scores are suffering from group bias, the result is not truly fair.

5.6.3 Amortized Correction

So far, our theoretical results in Section 5.4 and empirical results in Section 5.6.2 confirm the negative impact of group bias on ranking and fairness metrics. To correct for this bias, we have proposed an amortized correction method (Section 5.5). In the next set of experiments, we show the effectiveness of our proposed correction method in compensating for the negative effect of group bias.

Tables 5.1 and 5.2 compare the ranking quality, in terms of NDCG@10, as well as two merit-based fairness metrics, DTR and EEL, between the biased and corrected cases in the tabular regime. In all datasets and both bias parameter values, our correction method improves the ranking quality and fairness metrics over the biased case. With some exceptions for the ranking quality with mild group bias ($\beta = 0.8$), the improvements are significant. For each bias parameter value, we have also included the estimated value obtained from Eq. (5.16). We observe that our estimated bias values, i.e. $\hat{\beta}_{\mathcal{A}}$, are close to their corresponding true (but unknown during the correction) values β.

We further analyze the effectiveness of our correction method in the general LTR regime. Tables 5.3 and 5.4 contain the comparison of ranking quality and fairness metrics between the biased and corrected cases in our tested LTR datasets, i.e., Yahoo!

Table 5.2: The impact of our amortized group bias correction on ranking and fairness metrics in the tabular regime, under severe group bias ($\beta = 0.6$). $\hat{\beta}_{\mathcal{A}}$ shows the estimated value of the bias parameter as in Eq. (5.16). For each metric, the columns "*B*" and "C" show the "*Biased*" and "Corrected" performances, respectively. Superscripts * indicate a significant improvement over the biased case with $p < 0.001$.

	$\hat{\beta}_{\mathcal{A}}$	NDCG@10 ↑		ρ(DTR) ↑		Δ(EEL) ↓	
		B	C	*B*	C	*B*	C
Yahoo!$_{[426]}$	0.626	*0.885*	0.988*	*0.641*	0.941*	*1.809*	0.172*
MSLR$_{[127]}$	0.648	*0.780*	0.966*	*0.627*	0.926*	*7.146*	0.471*
IIT-JEE	0.547	*0.970*	0.985*	*0.600*	0.901*	*1.401*	0.410*
MovieLens$_{[Co.]}$	0.612	*0.998*	1.000*	*0.602*	0.958*	*7.113*	1.908*
MovieLens$_{[BO]}$	0.579	*0.994*	1.000*	*0.600*	0.961*	*24.800*	2.831*
TREC 2019	0.634	*0.986*	0.999*	*0.771*	0.937*	*0.129*	0.028*
TREC 2020	0.614	*0.945*	0.995*	*0.627*	0.941*	*1.152*	0.137*

Table 5.3: The impact of our amortized group bias correction on ranking and fairness metrics in general LTR regime, under mild group bias ($\beta = 0.8$). For each metric, the columns "*B*", "C", and "F" show the "*Biased*", "Corrected", and "Full info." performances, respectively. Superscripts * indicate a significant improvement over the biased case with $p < 0.001$.

	NDCG@10 ↑			ρ(DTR) ↑			Δ(EEL) ↓		
	B	C	F	*B*	C	F	*B*	C	F
Yahoo!$_{[426]}$	*0.61*	0.64*	0.65	*0.32*	0.43*	0.48	*1.97*	0.74*	0.67
MSLR$_{[127]}$	*0.28*	0.31*	0.32	*0.67*	0.68	0.68	*4.38*	2.02*	1.78

and MSLR. We also report the full information case in the table. Similar to the tabular regime, here we also observe performance improvements as a result of our correction method, compared to the biased case. Except for the DTR metric in MSLR, all the improvements are significant with $p < 0.001$. Compared with the full information case, we observe that in the Yahoo! dataset, our correction method leads to full recovery of NDCG@10, while in the MSLR dataset, there remains a slight gap toward the full information quality. One reason for this difference could be the distribution of relevant items in the affected and non-affected groups: In Yahoo!$_{[426]}$ the ratio between the mean relevance of items in $G_{\mathcal{A}}$ to $G_{\mathcal{N}}$ is 1.05, whereas the same quantity in MSLR$_{[127]}$ is 2.21. Therefore, the assumption of similar utility score distributions for both groups is closer to reality in Yahoo!$_{[426]}$ than in MSLR$_{[127]}$. Similarly to NDCG, we observe that DTR and EEL are almost fully recovered from group bias in the Yahoo! dataset, whereas in the MSLR dataset, there remains a larger gap toward the full information case after correction.

Finally, Figure 5.6 shows a summary of the ranking quality of biased (left) and corrected (right) utility scores in the tabular regime on all 32 setups of the LTR datasets

Table 5.4: The impact of our amortized group bias correction on ranking and fairness metrics in general LTR regime, under severe group bias ($\beta = 0.6$). For each metric, the columns "*B*", "C", and "F" show the "*Biased*", "Corrected", and "Full info." performances, respectively. Superscripts * indicate a significant improvement over the biased case with $p < 0.001$.

	NDCG@10 ↑			ρ(DTR) ↑			Δ(EEL) ↓		
	B	C	F	*B*	C	F	*B*	C	F
Yahoo!$_{[426]}$	*0.58*	0.64*	0.65	*0.31*	0.46*	0.48	*2.71*	0.73*	0.67
MSLR$_{[127]}$	*0.26*	0.31*	0.32	*0.67*	0.68	0.68	*5.80*	2.15*	1.78

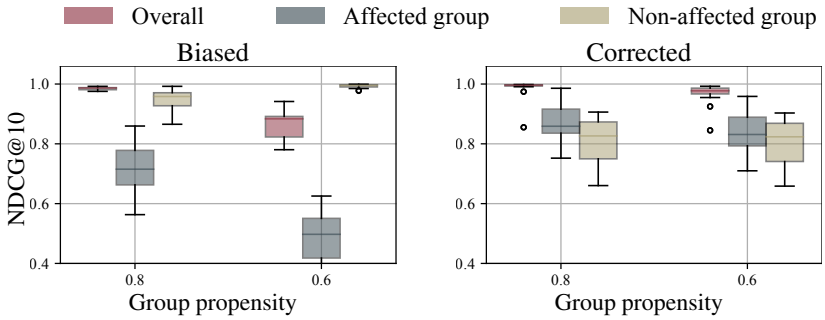


Figure 5.6: The ranking performance of biased (left) and corrected (right) scores for the Yahoo! and MSLR datasets with different sensitive attributes.

mentioned in Section 5.6.2. In all but two cases, we observe that our correction method effectively improves the ranking quality over the biased case and achieves NDCG@10 close to 1. The two outlier cases correspond to Yahoo!$_{[5]}$,[6] where the ratio between the average utility of the affected group and the non-affected group is as low as 0.3. This is the same outlier as in Figure 5.3. As our correction method is based on the same distribution assumption, this severe violation leads to inferred propensities that are noticeably lower than the actual propensity (i.e., 0.49 and 0.36 instead of 0.8 and 0.6). It is worth mentioning that in a slightly less severe violation of the same distribution assumption, i.e., MSLR$_{[130]}$ with a utility ratio of 0.45, our correction method is able to improve the ranking quality over the biased case. One interesting future direction would be to find out if this phenomenon, i.e., having the true average utility of the underrepresented group considerably lower than the other group, happens in real-world settings and how to correct for the bias in such cases.

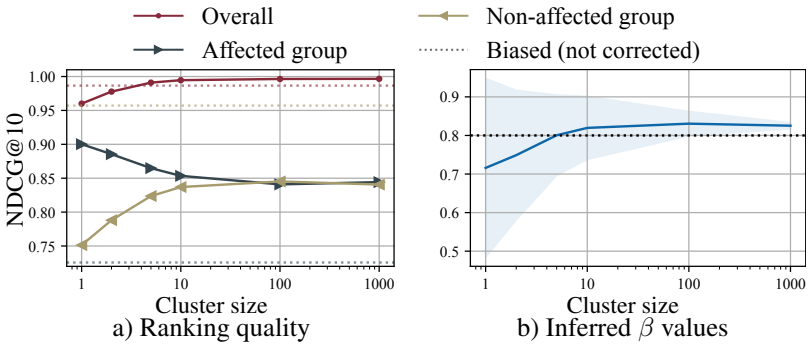


Figure 5.7: The impact of cluster size of group propensity on the amortized correction for group bias ($\beta_{\mathcal{A}} = 0.8$) on ranking performance for the Yahoo![426] dataset. Ranking quality of corrected scores (a), and accuracy of the inferred group propensity (b).

5.6.4 Ablation Study

Impact of Cluster Size on Correction

In Section 5.5.2 we argued against measuring the group propensity for each query. Here, we analyze the impact of cluster size on the correction method. We start from the extreme case of one query per cluster and increase the cluster size until the ranking quality converges. Figure 5.7 shows the ranking quality of the corrected scores as a function of the cluster size. The overall ranking quality (red line) improves as the cluster size grows and it converges to its final value at around a cluster size of 10. For the severe group bias ($\beta_{\mathcal{A}} = 0.6$), which we omit due to space restrictions, the same pattern is observed, but with a convergence point of 100. In both cases, using a cluster size below the convergence point leads to corrected rankings that are even worse than the biased ranking. Comparing the ranking quality of the affected group (black line) with the non-affected group (golden line), we observe that smaller clusters result in over-compensation of group bias. The reason is revealed in Figure 5.7(b): for smaller cluster sizes, the inferred propensity is under-estimated, leading to larger corrected scores for the affected group members. Consequently, the scores of the affected group are boosted more than they really should. One other interesting observation in Figure 5.7(b) is the high variance of the inferred propensity for small clusters (issue (i) in Section 5.5.2).

Impact of Clustering Accuracy

Finally, we address the challenge of inaccurate clustering of queries based on their group propensity that we raised at the end of Section 5.5.2. The main goal of the following sets of experiments is to show that our correction method, even when accurate clustering of queries is not available, is still effective in improving the ranking quality over the biased case. To confirm this, we add to the uncertainty of our simulation setup in two different ways: (i) *Higher variance*: We increase the variance of the group propensity when simulating the attractiveness probabilities. We consider $\sigma_\beta \in \{0.2, 0.3\}$. With the

[6]For each feature, two different thresholds for separating the groups are used.

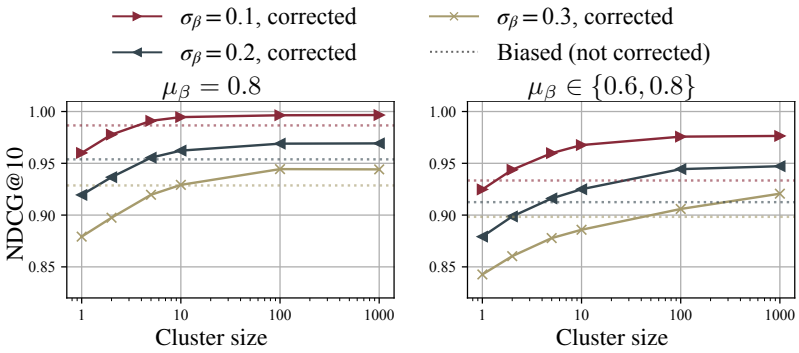


Figure 5.8: Effectiveness of our amortized correction method when accurate clustering based on group propensity is not available. Yahoo![426] dataset.

increased variance, the group propensity of queries can go far from the mean value, and, as we correct the queries with a single inferred value for the propensity, the probability of a mismatch between the actual propensity and the propensity used for correction increases. (ii) *Two modes*: Instead of using a unimodal normal distribution to simulate group propensity, we use a mixture model with two modes $\{0.6, 0.8\}$. This means that for half of the queries, the group propensity follows a normal distribution with a mean of 0.6 while for the other half, the normal distribution has a mean of 0.8 and during inference, we are not given the information about which query belongs to which mode.

Figure 5.8 shows the ranking quality of the corrected scores with respect to different cluster sizes in the increased uncertainty setups described above. In both plots, we observe that increasing the variance of the simulated group propensity both increases the negative impact of group bias on ranking (dotted lines) and makes it harder to correct for the bias (solid lines). The important result of these experiments, however, is that even though the uncertainty about group propensity is high, our amortized correction method almost always improves the ranking quality over the biased case. Note that in all setups, per-query correction as well as clusters with a small size lead to worse ranking qualities than the biased scores. Interestingly, when there are two modes of group propensity (right plot), our correction method, oblivious to the mode membership and assuming a fixed propensity, is able to correct the scores and achieve a ranking performance higher than the biased case.

5.7 Conclusion and Future Work

In this chapter, we have addressed group membership bias, which is based on the observation that a user's perception of an item's group membership may affect their judgment about the utility of an item. To address RQ4, we have provided extensive theoretical and empirical analyses of the impact of group bias on the ranking quality and two fairness of exposure metrics, DTR and EEL. By utilizing an auxiliary variable ν as the fraction of affected relevant items that are still as attractive as the non-affected relevant items, we have shown that, in the presence of group bias, NDCG and DTR

change linearly with ν, while the change in EEL has a more complex form in terms of ν.

Correcting for group bias, which is a type of *content-based* bias, is not as easy as *context-based* types of bias such as position and trust bias. To measure group bias, assumptions based on fairness constraints should be made about the utility distribution of different groups. However, such assumptions can potentially make the equity-based notion of fairness meaningless. Amortized correction for group bias is our solution to this issue, as global equality does not contradict local equity. We have experimentally confirmed that our correction method, when its assumptions are met, is able to fully recover the scores suffering from group bias, in the sense that the ranking and fairness metrics after correction achieve the values of the full information case.

Our amortized correction, however, raises a challenge of its own, as it is not easy to cluster queries with almost the same group propensity without knowledge of group propensity. Our experiments have shown that even when an accurate clustering of queries is not available, loosely clustering the queries for the amortized correction still leads to better rankings than the biased scores. More interestingly, per-query correction as well as clusters of small size lead to worse ranking qualities than the biased case.

There are several future directions to this study. Here, we analyzed a multiplicative model of group bias, and our theoretical and empirical results are based on this formulation. One way to extend our results is to consider more complex models for group bias. Another possible future direction is to propose measurement and correction methods that perform better with increased uncertainty of group propensity. Moreover, as group bias is based on users' perception of group membership, it can change over time. Analyzing group bias in a dynamic setting is therefore another interesting future direction. This study deals with the impact of group bias on fair exposure and, hence, we only consider so-called treatment-based fairness metrics. In contrast, some studies focus on impact-based fairness [108], where the objective is to make sure that items receive a fair amount of impact, e.g., clicks. While our work suggests a way to correct for group bias in historical clicks in order to make the exposure in future rankings fair, a next direction would be to account for group bias when optimizing for impact-based fairness.

In the next chapter, we re-examine the assumption of the availability of large number of repetitive sessions for fairness optimization in stochastic rankings and propose a novel representation for permutation distribution that can be used to optimize fairness metrics both in deterministic and stochastic ranking systems.

6

Optimizing for Fairness Metrics in Ranking: Probabilistic Permutation Graph Search

So far, we have studied several assumptions in the unbiased and fair counterfactual learning to rank (CLTR) literature. In this chapter, we conclude the thesis by re-examining the assumption of repetitive sessions for expected fairness and providing a general framework for optimizing fairness metrics that is effective in both high-repetition and low-repetition session settings. To elaborate, there are several measures for fairness in ranking, based on different underlying assumptions and perspectives. Plackett-Luce (PL) optimization with the REINFORCE algorithm can be used for optimizing black-box objective functions over permutations. In particular, it can be used for optimizing fairness measures. However, though effective for queries with a moderate number of repeating sessions, PL optimization has room for improvement for queries with a small number of repeating sessions.

In this chapter, to address RQ5, we present a novel way of representing permutation distributions, based on the notion of permutation graphs. Similar to PL, our distribution representation, called probabilistic permutation graph (PPG), can be used for black-box optimization of fairness. Different from PL, where pointwise logits are used as the distribution parameters, in PPG pairwise inversion probabilities together with a reference permutation construct the distribution. As such, the reference permutation can be set to the best sampled permutation regarding the objective function, making PPG suitable for both deterministic and stochastic rankings. Our experiments show that PPG, while comparable to PL for larger session repetitions (i.e., stochastic ranking), improves over PL for optimizing fairness metrics for queries with one session (i.e., deterministic ranking). Additionally, when accurate utility estimations are available, e.g., in tabular models, the performance of PPG in fairness optimization is significantly boosted compared to lower quality utility estimations from a learning to rank model, leading to a large performance gap with PL. Finally, the pairwise probabilities make it possible to impose pairwise constraints such as "item d_1 should always be ranked higher than item

This chapter was published as: A. Vardasbi, F. Sarvi, and M. de Rijke. Probabilistic permutation graph search: Black-box optimization for fairness in ranking. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 715–725, 2022.

d_2." Such constraints can be used to simultaneously optimize the fairness metric and control another objective such as ranking performance.

6.1 Introduction

Several fairness measures are being considered in the search and recommendation literature. Different measures have been proposed based on different definitions of fairness, aimed at different environments. For instance, Singh and Joachims [114] consider the disparate treatment ratio (DTR), which ensures equity of exposure: each group should get exposed proportional to their utility. DTR for group fairness can be used for deterministic rankers, which produce one ranking per query, as well as stochastic rankers, with different randomly sampled rankings per query. It also makes no assumptions about the utility values of the ranked items. In contrast, the expected exposure loss (EEL) is based on the premise that groups (or individuals) with the same relevance level should have equal expected exposures [32]. By definition, EEL should be used for stochastic rankers. It is also assumed in EEL that the relevance levels have discrete values.

Optimizing fairness measures. The goal of this chapter is not to list and compare different fairness definitions; much has already been written about this [106, 128]. Neither do we want to unify different fairness measures, since they deal with different aspects of fairness, and more importantly, it has been shown that there is an inherent trade-off between some fairness conditions [65], which makes it impossible to have a unified fairness measure. What we aim to accomplish is to provide a general framework that can be used for *optimizing any fairness measure*.

We focus on post-processing methods for fairness and assume that the utility value of the items is given, either from external sources such as unbiased clicks or an estimate computed by a learning to rank (LTR) model. In order to remain general, the framework should work with black-box access to the function that evaluates ranking fairness. Optimization of fairness in ranking is a special case of permutation optimization, and reinforcement learning (RL) is usually the default paradigm for combinatorial optimization: A model-free policy-based RL can be used for permutation optimization [11]. Using the well-known REINFORCE algorithm [133], and sampling from a PL distribution, it is possible to optimize any fairness objective function on permutations [115].

A PL distribution with the REINFORCE algorithm works very well for optimizing stochastic ranking evaluation metrics such as EEL. However, there are two directions in which PL-based optimization has room for improvement. First, when the number of repeating sessions for a query is very small, the high variance of PL [43] leads to sub-optimal results. Second, when an accurate estimate of the utilities is available, the solutions obtained from PL are only slightly improved over the case of noisy utility values. The first case has practical importance, as it is desirable to have a good fairness solution as soon as possible, before there are a large number of repeating sessions, i.e., a method that provides fair solutions both for deterministic and stochastic rankings. The second case has both theoretical and practical importance. From a theoretical point of view, since fairness measures usually depend on utility values,[1] noise in utility

[1] Both EEL and DTR metrics do.

$[d_1, d_2, d_3, d_4] \rightarrow [d_2, d_3, d_1, d_4]$ $[d_3, d_2, d_4, d_1] \rightarrow [d_1, d_2, d_3, d_4]$

Figure 6.1: Examples of permutation graphs.

estimates propagates into the fairness optimization, whereas with accurate utility values, the performance of the fairness optimizer is isolated and not affected by noise from utility values. This gives a more accurate comparison between different optimization algorithms. From a practical point of view, given enough historical sessions, unbiased and accurate estimates of utility can be obtained from clicks [58, 74, 123, 143]. For instance, extremely high-performance rankings can be achieved from tabular models in practice [116]. We expect to observe a boost in fairness optimization when noisy estimates are replaced with unbiased low-noise estimates of utility.

Black-box permutation optimization. To bridge the performance gap left by PL in the above two cases, we seek to answer the following question in this chapter:

RQ5 Is it possible to have a single general fairness optimization method that performs well for both stochastic and deterministic rankings?

We propose a novel permutation distribution for black-box permutation optimization with the REINFORCE algorithm. Our permutation distribution is based on the notion of permutation graphs [35]. A permutation graph is a graph whose nodes are the items in a ranked list, and whose edges represent inversions in a permutation over the ranked list. We refer to the initial permutation of the ranked list as the *reference permutation*. Two examples of permutation graphs are given in Figure 6.1. In the left graph, over the reference permutation of $[d_1, d_2, d_3, d_4]$, d_1 is moved from the first position to the third, causing two inversions (d_1, d_2) and (d_1, d_3). In the right graph, over the reference permutation of $[d_3, d_2, d_4, d_1]$, d_1 is brought forward, and d_3 is swapped with d_2, causing a total of four inversions, as can be seen in the graph.

Building on permutation graphs, we define *probabilistic permutation graph* (PPG) as a permutation distribution from which the REINFORCE algorithm can sample. Roughly speaking, PPG is a weighted complete graph, whose edge weights are the probabilities of inverting the two endpoints in a permutation. Unlike PL, in which pointwise logits are used as distribution parameters, PPG is constructed from a *reference permutation* together with pairwise inversion probabilities. The reference permutation in PPG helps in deterministic scenarios, as it can be set to the best permutation sampled during the REINFORCE algorithm. This is not possible for a PL distribution, as the deterministic permutation of PL comes from sorting the items based on their logits and such a permutation is not necessarily the best sampled permutation during training.

In addition to PPG's gain over PL due to its reference permutation in deterministic as well as accurate relevance estimation scenarios, the pairwise inversion probabilities give PPG another useful property that PL lacks. In PPG, pairwise constraints can be added during optimization, without any additional computational overhead: it is enough

to set some edge weights to non-trainable parameters. For example, the constraint of "item d_1 should always be ranked higher than item d_2" can be added by setting to zero the weight of the (d_1, d_2) edge. Business related, time-aware, or context-aware pairwise constraints can be thought of, all of which are easily handled by PPG (see Section 6.3.4).

Our fairness optimization method is a post-processing method that acts on lists of items, ranked based on utility. Compared to in-processing methods, such as [115], a post-processing method allows for richer dynamics, in the sense that having the fairness optimizer separate from the ranker, makes any change on the fairness side independent of the ranking side. For example, with a post-processing method, if the sensitive groups of attention change over time, either by adding new sensitive features to the existing ones or replacing them, there is no need to re-train the ranker with a new set of group labels. Our black-box fairness optimization method even adds to this flexibility, as the objective fairness measure itself is allowed to change over time or in different contexts. Adding to this flexibility, while in-processing fairness optimization methods are an option in feature-based ranking models, they simply cannot be used in tabular models where the best rankings are memorized.

Our contributions. Our contributions in this chapter are listed below:

1. We define probabilistic permutation graph (PPG), a novel permutation distribution.
2. We present PPG search as a general framework for optimizing fairness in ranking. Our method is general in the sense that it works with black-box access to the objective function. As such, PPG search can be used in contexts beyond fairness and diversity, to find a permutation that optimizes a general objective function.
3. We experimentally show the effectiveness of PPG search on two popular objective functions, EEL and DTR, and various public datasets by comparing its performance with state-of-the-art methods for optimizing fairness.
4. We experimentally verify the effect of pairwise constraints on controlling the ranking performance while optimizing for fairness.

6.2 Background and Related Work

Fairness. Widely used ranking algorithms at the core of many online systems such as search engines and recommender systems raise fairness considerations, since these algorithms can directly control the exposure each item receives and potentially have societal impacts [8].

Similar to other areas of machine learning, various approaches have been proposed to evaluate fairness in ranking. Zehlike et al. [141] distinguish two lines of work based on how fairness is measured for a ranking policy: probability-based approaches determine the probability that a given ranking is generated by a fair policy [7, 22, 23, 44, 117, 137, 140], while exposure-based methods aim to ensure that the expected exposure is fairly distributed among items (individual fairness), or item groups (group fairness) [13, 32, 52, 90, 92, 110, 111, 114, 115, 135].

From the methods belonging to the second category, many have followed the concept of demographic parity, which enforces a proportional allocation of exposure between groups [22, 44, 137, 140]. This approach to fairness does not consider merit and only

takes into account the group size. On the other hand, disparate treatment is a merit-based approach to fair ranking proposed in [114] which makes exposure allocation dependent on the merit of each group. They developed a framework which allows for other forms of fairness definitions that can be formulated as linear constraints.

Methods introduced in these publications are dependent on their notion of fairness. However, fairness goals can be application specific. In this work, we propose a black-box optimization method for fairness in ranking that is able to optimize a ranking policy w.r.t. any arbitrary fairness definition.

Gradient estimators for permutations. The group of all permutations of size n is called the *symmetric group* and is denoted by S_n. Permutation optimization can be stated as follows:

$$\min f(b), \quad b \in S_n,$$

where $f : S_n \to \mathbb{R}$ can be any general function on permutations. For this combinatorial optimization problem, the gradient solution is not as clear as continuous differentiable optimization. Instead, a policy-based RL approach can be used to optimize the expectation of $f(b)$ over a differentiable probability distribution $P(b \mid \theta)$, represented by a vector of continuous parameters θ [12]:

$$F(\theta) = \mathbb{E}_{P(b|\theta)} \left[f(b) \right]. \tag{6.1}$$

$F(\theta)$ is optimized when $P(b \mid \theta)$ is totally concentrated on b^*, the optimum solution of $f(b)$. For Eq. (6.1), the REINFORCE estimator [133] can be used:

$$\nabla_\theta F(\theta) = \mathbb{E}_{P(b|\theta)} \left[f(b) \cdot \nabla_\theta \log P\left(b \mid \theta\right) \right]. \tag{6.2}$$

Finally, since S_n is exponentially large, the expectation in Eq. (6.2) can be estimated through Monte-Carlo (MC) sampling:

$$\nabla_\theta F(\theta) \approx \frac{1}{\lambda} \sum_{i=1}^{\lambda} f(b_i) \cdot \nabla_\theta \log P\left(b_i \mid \theta\right), \tag{6.3}$$

where $b_1, \ldots, b_\lambda \in S_n$ are samples drawn from $P(b \mid \theta)$.

It remains to discuss the probability distribution on S_n. For a thorough exploration of probability distributions for permutations we refer the reader to [42]. The PL model [82, 100] is by far the most widely used permutation distribution in the REINFORCE algorithm, both in general permutation optimization [43] and the fairness literature [93, 115]. PL is represented by a parameter vector $\theta \in \mathbb{R}^n$, and the probability of a permutation $b = [b_1, \ldots, b_n] \in S_n$ under PL is calculated as:

$$P(b \mid \theta) = \prod_{i=1}^{n-1} \frac{\theta_{b_i}}{\sum_{j=i}^{n} \theta_{b_j}}. \tag{6.4}$$

Sampling from PL and estimating the log derivative of the probability $P(b \mid \theta)$ is usually done using the Gumbel Softmax trick [48, 84].

In this work, we use the REINFORCE algorithm but with our novel PPG distribution instead of PL.

Tabular search. In practice, not all queries are served with a feature-based LTR model. Tabular models usually achieve optimal performance for head and torso queries for which enough user interactions are available [70]. In particular, underperforming torso queries are given to the tabular models for a better *memorized* ranking [45, 86, 116, 120, 143]. The online LTR literature is filled with methods for tabular model optimization [60, 70, 74]. As tabular models are not limited by the capacity of the LTR model, they usually converge to extremely high ranking performance [97, 143].

This work relates to tabular search by exploiting its results as a use case. We show in our experiments that, given accurate estimates of utility measures of the items, e.g., their relevance labels, our proposed method for fairness optimization performs significantly better than state-of-the-art fairness optimizers. Tabular models are practical and important evidence to show that the availability of accurate relevance estimates is not just hypothetical.

6.3 Probabilistic Permutation Graph Search

Given a list of items D, together with a black-box objective function $f : S_n \to \mathbb{R}$ that can be queried for every permutation of D, our goal is to find a distribution over permutations that optimizes f. We assume that the utility of items, e.g., their relevance level, is given, or an estimate of the utility is obtainable using a LTR model. Therefore, our task is to post-process a ranked list and output a permutation, or a sequence of permutations, that optimize the given objective function. To do so, we use the REINFORCE algorithm to search the permutation space and update the PPG distribution parameters. Below, we first formally define PPG and provide a log derivative formula for the PPG distribution. After that, we propose a method for efficiently sampling from the PPG distribution. Finally, we discuss pairwise constraints, including practical examples.

6.3.1 PPG Distributions

We define PPG to be a probabilistic graph from which permutation graphs are sampled.

Definition 6.1 (Probabilistic permutation graph (PPG))**.** *Given a list of items D and a reference permutation π_0 on D, a PPG corresponding to π_0 is a weighted complete graph $G = (D, E, w)$, whose edges are weighted by a probability value obtained from $w : E \to [0, 1]$. For each edge $e \in E$, its weight $w(e)$ indicates the Bernoulli sampling probability that e is included in a permutation graph over π_0 sampled from G.*

In what follows, we write e_{ij} for the edge (d_i, d_j) and $w_{e_{ij}}$ for its weight (or w_{ij} when no confusion is possible). According to the above definition, PPG represents a permutation distribution. To sample a permutation from a given PPG, a Bernoulli sampling process is run on the edges of G with their corresponding weights. Suppose that, after edge sampling, $E_\pi \subset E$ is the set of edges that are selected (i.e., positively sampled) by the sampler and $E \backslash E_\pi$ is the set of remaining, left out edges (i.e., negatively

sampled). The probability of this outcome is calculated as:

$$P(E_\pi \mid w) = \prod_{e \in E_\pi} w_e \cdot \prod_{e \in E \setminus E_\pi} \big(1 - w_e\big). \tag{6.5}$$

Now, we notice that the set of all permutation graphs over a list of items D is only a small subset of all possible graphs. This means that E_π may not correspond to the edges of a valid permutation graph. If that happens, i.e., if the graph constructed from the E_π edge set is not a permutation graph, we drop E_π and repeat the sampling process until we find a valid permutation graph. Checking if a given graph is a permutation graph or not is possible in linear time [87]. We write $\mathcal{P}$ for the space of all permutation graphs. Then, the probability that a randomly sampled graph from G is a permutation graph can be computed in theory by:

$$\rho = \sum_{E_\pi \in \mathcal{P}} P(E_\pi \mid w). \tag{6.6}$$

Note that in practice, computing ρ requires an exponential $n!$ number of computations and, hence, is not feasible. We will come back to this point in Section 6.4 below. Using ρ, the permutation probability mass can be written as:

$$P(E_\pi \mid w, E_\pi \in \mathcal{P}) = \frac{1}{\rho} \prod_{e \in E_\pi} w_e \cdot \prod_{e \in E \setminus E_\pi} \big(1 - w_e\big). \tag{6.7}$$

It is not hard to show that repeatedly sampling graphs until we sample a valid permutation graph will lead to the conditional probability distribution of Eq. (6.7). We interchangeably use $E_\pi \in \mathcal{P}$ for both the positively sampled edges of the permutation graph, and the permutation corresponding to the graph.

To work with the REINFORCE algorithm (Section 6.2), we need to compute the log derivative of the probability distribution of PPG. In what follows we assume that $E_\pi \in \mathcal{P}$ and drop the conditions from our notation for brevity. We start by using Eq. (6.7) and taking the log derivative as follows:

$$\begin{aligned} \frac{\partial \log P(E_\pi)}{\partial w_e} &= \frac{\partial}{\partial w_e} \log \left(\prod_{e \in E_\pi} w_e \cdot \prod_{e \in E \setminus E_\pi} \big(1 - w_e\big) \right) \\ &\quad - \frac{\partial}{\partial w_e} \log \rho \\ &= \frac{\mathbb{I}\big(e \in E_\pi\big) - w_e}{w_e \cdot \big(1 - w_e\big)} - \frac{1}{\rho} \frac{\partial \rho}{\partial w_e}. \end{aligned} \tag{6.8}$$

where $\mathbb{I}(\cdot)$ is the indicator function. In Section 6.4 we show the second term can be ignored to obtain the following approximation:

$$\frac{\partial \log P(E_\pi)}{\partial w_e} \approx \frac{\mathbb{I}\big(e \in E_\pi\big) - w_e}{w_e \cdot \big(1 - w_e\big)}. \tag{6.9}$$

So far, we have defined the PPG and provided the log derivative of its probability distribution. However, repeatedly dropping the sampled graphs until a valid permutation graph is found, is not efficient. Next, we present an efficient method for sampling permutations from the PPG distribution.

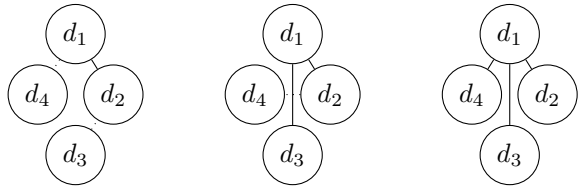


Figure 6.2: Different cases in Example 6.1. Dotted lines show the impossible-to-be-selected edges due to the position of d_1 in the bottom sub-list.

6.3.2 Efficient Sampling of Permutation Graphs

In order to avoid redundantly sampling non-permutation graphs from a PPG, we propose a divide and conquer approach for efficiently constructing a valid permutation graph. The idea is to divide the list of items into two sub-lists, obtain a permutation on each of the sub-lists by recursively sampling from the given PPG, and merge the two sampled sub-lists into a single permuted list. Below, we discuss our *merge sampling* method in more detail, but first, let us give an example to illustrate the general idea.

Example 6.1. Suppose a reference permutation $[d_1, d_2, d_3, d_4]$ and a PPG are given. In order to sample a permutation, we proceed as follows:

(I) ***Divide.*** We divide the list into two sub-lists $[d_1, d_2]$ and $[d_3, d_4]$.

(II) ***Sample sub-lists.*** Each sub-list is sampled independently according to PPG. Here, each pair of $[d_1, d_2]$ and $[d_3, d_4]$ are swapped with a probability equal to their weight, i.e., w_{12} and w_{34}. Suppose only the first pair is swapped $[d_1, d_2] \rightarrow [d_2, d_1]$.

(III) ***Merge.*** Finally, we merge $[d_2, d_1]$ and $[d_3, d_4]$. In doing so, we keep the order of items in each sub-list unchanged, i.e., no new edges are added between the items of the same sub-list. We start from the last item of the first sub-list, d_1. d_1 can go to three possible positions as depicted in Figure 6.2: (Case 1) Before d_3, so e_{13} is negatively sampled. In this case, since the order of the first sub-list should remain unchanged, d_2 has only one possible position. (Case 2) Between d_3 and d_4, so only e_{13} is positively sampled. In this case, d_2 has two possible positions: before and after d_3. (Case 3) After d_4, so both e_{13} and e_{14} are positively sampled and d_2 has three possible positions. These three possibilities are tested sequentially. First, we sample e_{13}. If it is not selected, it means d_1 should remain before d_3 and we are done (Case 1). Otherwise, d_1 should go after d_3. Then, we sample e_{14} and stop if it is not selected (Case 2). Finally, if e_{14} is also selected, we have case 3. After d_1 has been merged, we repeat the above process for merging d_2.

(IV) ***Probability correction.*** If e_{13} is negatively sampled, neither e_{14} or e_{23} can be sampled (Figure 6.2, Case 1). Therefore, when sampling e_{13}, we have to account

for the impossible outcome of not selecting e_{13} but selecting at least one of e_{14} or e_{23}. The probability of this event is calculated as:

$$q_{13} = (1 - w_{13})(w_{14} + w_{23} - w_{14}w_{23}).$$

The sum of possible outcomes would then be equal to $1 - q_{13}$ and the sampling probability of e_{13} should be corrected by $w_{13}/\big(1 - q_{13}\big)$. Similar arguments can be given for Case 2, where e_{13} is positively sampled. Similarly, not selecting e_{14} but selecting e_{24} is impossible. So, sampling of e_{14} should be done by a probability equal to $w_{14}/\big(1 - q_{14}\big)$, where $q_{14} = (1 - w_{14})w_{24}$. □

In the above example, we saw the simplest case of merging two sub-lists of size two. More generally, assume we want to merge two permuted sub-lists $[d_1, \dots, d_T]$ and $[d_{T+1}, \dots, d_n]$. Here we re-indexed the items to simplify the notation. We start by merging the last item from the top sub-list d_T into the bottom sub-list. Assume d_{t+1} from the top sub-list is merged just after d_b in the bottom sub-list. Now, for merging d_t, we know that it should be placed before d_{t+1}, meaning that at most it can be inverted with d_b, but not any item after that in the bottom sub-list. Starting from d_{T+1} on the bottom sub-list, we sample the inversions using the corrected probabilities and stop as soon as one inversion was sampled negatively. Assume the inversions of d_t with $d_{T+1}, \dots, d_{i-1}$ were positively sampled and we want to sample the inversion with d_i. There are two independent impossible outcomes: e_{ti} is sampled negatively, but (A) at least one of the e_{tj} for $i < j \leq b$ is sampled positively; or (B) at least one of the $e_{t'i}$ for $t' < t$ is sampled positively. The probability of these impossibilities can easily be calculated. We write q_{ti} for the probability of an impossible outcome due to negatively sampling e_{ti}:

$$q_{ti} = \big(1 - w_{ti}\big)\big(P(A) + P(B) - P(A)P(B)\big). \tag{6.10}$$

As the sum of all possible events is $1 - q_{ti}$, the inversion of e_{ti} should be sampled with the corrected probability of $w_{ti}/\big(1 - q_{ti}\big)$.

The recursive algorithm of sampling a permutation graph is shown in Algorithm 6.1. The main part of the algorithm is the **Merge** function in line 14. The pseudo-code for the merge function is given in Algorithm 6.2. The worst-case complexity of calculating q_{ti} by Eq. (6.10) is $\mathcal{O}(n)$ which makes the worst-case complexity of **Merge** (Algorithm 6.2) equal to $\mathcal{O}(n^3)$. In practice, we update the reference permutation each time a better permutation with respect to the objective function was sampled during training. As the training goes on, the reference permutation gets closer to the optimum permutation and the model gets more confident in it. This means that the inversion probabilities are constantly decreasing. In the extreme case when the probability of changing the reference permutation becomes very small, the average complexity of **Merge** becomes linear and the total complexity of **Sample** (Algorithm 6.1) becomes equal to $\mathcal{O}(n \log n)$.

Experiments show that our efficient sampling method does not provide a uniform sampling: in the trade-off between accurately sampling according to a given PPG distribution and having an efficient sampler, our method is inclined to the latter. Our experimental results in Section 6.6.1 verify that this sampling method is effective in fairness optimization. Further analyzing the accuracy-efficiency trade-off of sampling PPG distributions and proposing sampling methods with different degrees of accuracy and efficiency is an interesting direction that we leave for future work.

Algorithm 6.1: Sample(D, G)**.** Sampling Permutation Graphs

Input: $D = [d_1, d_2, \ldots, d_n]$, $G = (D, E, w)$
Output: D' (a permutation on D, sampled from G)

if $n == 2$ **then**
  Bernoulli sample with probability w_{12}
  if *Positive* **then**
    Return $[d_2, d_1]$
  else
    Return $[d_1, d_2]$
  end
end
Split D in half to D_t and D_b
Set G_t to the upper left square of G corresponding to D_t
Set G_b to the lower right square of G corresponding to D_b
$D'_t =$ **Sample**(D_t, G_t)
$D'_b =$ **Sample**(D_b, G_b)
Return **Merge**(D'_t, D'_b, G)

6.3.3 Learning PPG Weights

We use the REINFORCE algorithm to train the weights of our PPG distribution as described in Section 6.2. For MC sampling from the permutation distribution as required in Eq. (6.3), we use the **Sample** algorithm as described in Algorithm 6.1. The only difference with the standard REINFORCE is that we keep track of the minimum value for the objective function and the corresponding permutation. After each permutation has been sampled and the objective function has been calculated, we update the reference permutation with the best sampled permutation.

Algorithm 6.3 contains the pseudo-code for learning the PPG weights. In line 8, P is the permutation matrix of E_{π_i}.

6.3.4 Pairwise Constraints

A good property of PPG search is that pairwise constraints on the permutations can be handled without extra computational complexity. Pairwise constraints in the form of forbidden inversions, can be handled by setting to zero the appropriate set of edges in the PPG. Here we give some practical examples of this type.

Intra-group fixed-ranking. An intra-group constraint ensures that the ranking of a group of items remains unchanged among themselves. Consider, for example, $D = [d_1, d_2, d_3, d_4]$ with the group $g = \{d_2, d_3, d_4\}$. An intra-group fixed-ranking constraint on g means that the ranking of the items within g should remain the same as the reference permutation. The permutation $[d_2, d_1, d_3, d_4]$ meets this constraint, but in $[d_1, d_3, d_2, d_4]$ or $[d_4, d_1, d_3, d_2]$ the ranking of items of g is changed and the constraint is violated. Multiple groups can be defined over the item list, and the groups need not be disjoint. To handle intra-group constraints, in the initial PPG, we simply set to zero the

Algorithm 6.2: Merge(D_t, D_b, G)**.** Merging Two Sampled Permutations

Input: $D_t = [d_1, \ldots, d_T]$, $D_b = [d_{T+1}, \ldots, d_n]$, $G = (D_t \cup D_b, E, w)$
Output: D' (a merged permutation on $D_t \cup D_b$, sampled from G)

Initialize the last merged index $b_{last} = n + 1$
for t *from* T *down to* 1 **do**
  Insert d_t on top of D_b
  for i *from* $T + 1$ *to* b_{last} **do**
    Calculate q_{ti} using Eq. (6.10)
    Bernoulli sample with probability $w_{ti}/\big(1 - q_{ti}\big)$
    if *Negative* **then**
      $b_{last} = i$
      Break
    else
      Invert: $[\ldots, d_t, d_i, \ldots] \rightarrow [\ldots, \underline{d_i, d_t}, \ldots]$
    end
  end
end
Return D_b

weights of all the edges between items within the same group. This ensures that none of these edges will be sampled during the learning process. As a result, no inversion is performed over the items within the same group.

A use case for this constraint is in group fairness, where we post-process the output of an LTR algorithm to make the ranking fair with respect to some grouping. Each inversion in the output of the LTR algorithm means degradation in our estimated best ranking. Therefore, the inversions are performed only to improve fairness. But inverting two items from the same fairness group does not change the fairness metric. Such inversions that degrade the ranking and do not improve fairness can be avoided by setting an intra-group constraint on the permutations.

Inter-group fixed-ranking. Inter-group constraints can be used to ensure that one group is ranked higher than the other group. Consider, for example, $D = [d_1, d_2, d_3, d_4, d_5]$ with the groups $g_1 = \{d_1, d_2\}$ and $g_2 = \{d_4, d_5\}$. The permutation $[d_2, d_3, d_1, d_4, d_5]$ meets the inter-group constraint, as all the items of g_1 are ranked higher than all the items of g_2, the same as the reference permutation. But $[d_1, d_4, d_3, d_2, d_5]$ violates the constraint because $d_4 \in g_2$ is ranked higher than $d_2 \in g_1$. To handle this constraint, in the initial PPG, we set to zero the weights of all edges whose endpoints are in different groups. Consequently, there would be no inversion between two items from different groups.

A use case for this constraint is in amortized fairness, where fairness is measured over multiple sessions (with the same or different queries). Since the objective function is calculated over multiple lists, in PPG search we can concatenate the lists and search for a solution over the concatenated list. In this case, items from different sessions should not be inverted: we are only allowed to change the permutation within each

Algorithm 6.3: Learning PPG weights

Input: π_0, $G = (D, E, w)$, f, η
Output: Updated π_0 and G

while *Not converged* **do**
  for $i = 1, \ldots, \lambda$ **do**
    Use Algorithm 6.1 to sample a permutation E_{π_i}.
    Use Eq. (6.9) to calculate the log derivative of PPG.
    if $f(E_{\pi_i}) < f(\pi_0)$ **then**
      $\pi_0 = E_{\pi_i}$
      Update G accordingly:
        $G = P^T \cdot G \cdot P$
    end
  end
  Use Eq. (6.3) to estimate gradients.
  Update the weights by gradient descent, using learning rate η.
end
Return π_0, G

session. Using an inter-group fixed-ranking constraint, with each session considered as a group, is a simple solution for such a use case.

Time-aware ranking. In some scenarios, such as news search or job search, items are associated with time and it is sometimes important not to rank very old items before recent items. A time-aware ranking constraint is a special case of inter-group fixed-ranking, where groups are defined based on time.

Context-aware ranking. Another special case of an inter-group fixed-ranking constraint is context-aware ranking. Assume, for example, the case of sponsored links. We want to post-process the ranking to make it fair, but in addition to that, we want to have the relevant sponsored links to appear at the top of the list. We can define groups based on the sponsored status of items and set an inter-group constraint on the PPG graph.

We use the first two examples, namely intra-group and inter-group fixed-ranking, in our experiments. With the intra-group constraint, we control the ranking performance of permutations while minimizing the fairness, and with the inter-group constraint, we train over multiple sessions of one query with a single PPG model.

6.4 Log Derivative of Probability Distribution

Here we discuss the approximate log derivative of the probability distribution, given in Eq. (6.9). Rewriting Eq. (6.8), we have:

$$\frac{\partial \log P(E_\pi)}{\partial w_e} = \underbrace{\frac{\mathbb{I}\big(e \in E_\pi\big) - w_e}{w_e \cdot \big(1 - w_e\big)}}_{\alpha_w} - \underbrace{\frac{1}{\rho}\frac{\partial \rho}{\partial w_e}}_{\beta_{\rho,w}}.$$

The first term, α_w, only depends on the weight w_e and sampled graph E_π. However, the second term, $\beta_{\rho,w}$, depends on ρ which cannot be calculated feasibly because of the exponential size of the permutation space. Consequently, we explain why it is save to consider $\alpha_w - \beta_{\rho,w} \approx \alpha_w$ in gradient descent. First note that, fixing all the weights other than w_e, ρ is a linear function of w_e:

$$\rho = w_e \cdot A + \big(1 - w_e\big) \cdot B,$$

where A is the probability sum of all the permutation graphs containing e and B is the probability sum of all the permutation graphs not containing e. Therefore, we have

$$\beta_{\rho,w} = \frac{A - B}{w_e \cdot \big(A - B\big) + B}.$$

We know from graph theory that [35]:

$$E \in \mathcal{P} \Rightarrow \bar{E} \in \mathcal{P}, \tag{6.11}$$

where $\bar{E}$ is the complement graph of E. For training the PPG weights, we initialize all the weights to 0.5 and slightly change them using the gradients. Simple counting shows that, when all the edges have a 0.5 probability of sampling, we have:

$$A = B = \frac{n!}{2^{\frac{n(n-1)}{2}}}, \tag{6.12}$$

which means that for the initial weights $\beta_{\rho,w} = 0$. We further show that, even when $\beta_{\rho,w} \neq 0$, the gradient and α_w *always* have the same sign. This is a critical condition in gradient descent with small learning rate, as the weights are guaranteed to update in the correct *direction*. To show α_w and $\alpha_w - \beta_{\rho,w}$ always have the same sign, we notice that when α_w and $\beta_{\rho,w}$ have different signs, the proposition is true. It remains to prove the proposition for the case where α_w and $\beta_{\rho,w}$ have the same signs:

1. Case 1: $\alpha_w = \frac{1}{w_e} > 0$ and $\beta_{\rho,w} > 0$

$$w_e(A - B) + B > A - B \xRightarrow{(A-B>0)} \beta_{\rho,w} < \frac{1}{w_e}. \tag{6.13}$$

2. Case 2: $\alpha_w = \frac{-1}{1-w_e} < 0$ and $\beta_{\rho,w} < 0$

$$\begin{aligned} &A \cdot w_e + B(1 - w_e) > -(A - B)(1 - w_e) > 0 \\ &\xRightarrow{(A-B<0)} -\beta_{\rho,w} < \frac{1}{1 - w_e}. \end{aligned} \tag{6.14}$$

To sum up, we have shown that in the working range of weights, i.e. ≈ 0.5, we have $\beta_{\rho,w} \approx 0$. And more importantly, in general, α_w and $\alpha_w - \beta_{\rho,w}$ always have the same sign, ensuring that the gradient descent updates the weights in the correct direction. Our experiments show a very negligible sensitivity of PPG search to the learning rate, which translates to negligible sensitivity to the gradient size.

6.5 Experimental Setup

In order to show the effectiveness of PPG search, we perform experiments on real-world public data and compare the performance of different fairness methods. Below, we detail the datasets, experimental setup, and the baselines that we compare to.

6.5.1 Data

MSLR. This is a regular choice in counterfactual LTR research [58, 122, 123], as well as fairness studies [32, 93, 135]. We use Fold 1 of MSLR-WEB30k [102] with 5-level relevance labels. On average, MSLR has 120.19 documents per query. We follow [135] and divide the items into two groups based on their QualityScore2 (feature id 133) with a threshold of 10. This gives a 3:2 ratio between the groups' population. We choose a subset of the MSLR test set for post-processing based on the following criteria. We filter the queries for which there is not at least one fully relevant item, i.e., level 4. We also filter the queries that only contain relevant items from one group, as the DTR metric cannot be used for such queries. For the remaining queries, we subsample queries with long item lists to have a maximum of 20 items, based on their LTR score. The intuition is that usually, a top-k cut of the items are shown to users in online search engines. This is in line with previous fairness and online LTR studies [e.g. 122, 123, 135].

TREC. Our second dataset is the academic search dataset provided by the TREC Fair Ranking track[2] 2019 and 2020 [14]. TREC 2019 and 2020 editions of the dataset come with 632 and 200 train and 635 and 200 test queries, respectively, with an average of 6.7 and 23.5 documents per test query. This dataset has been previously used in fair ranking research [52, 63, 111]. Following [111], we divide the items (i.e., papers) into two groups based on their authors' h-index.

6.5.2 Setup

LTR model. We use a neural network with two hidden layers of width 256, ReLU activations, dropout of 0.1, and a learning rate of 0.01. As it is important to have a calibrated LTR for fairness, as noted in [32], we use a pointwise loss function in the form of mean-squared error (MSE). We notice that the dynamic range of our LTR model is limited: around 95% of the items are concentrated in an interval of length 0.05 in all of the datasets. As both DTR and EEL metrics rely on relevance estimates, to calibrate the scores onto a realistic interval, we min-max normalize the scores per query onto $[0, 5)$ and then discretize them to integer grades from 0 to 4, similar to the relevance grades in popular LTR datasets [102].

Metrics. We evaluate models for fair ranking in terms of user utility and item fairness. For utility, we use normalized discounted cumulative gain (NDCG) and report NDCG@10. For fairness we evaluate models using two metrics: DTR [114] and EEL [32].

Given two item groups, G_1 and G_2, DTR, measures how unequal exposure ($\exp$) is allocated to the two groups based on their merit, which, in our case, translates to the

[2] `https://fair-trec.github.io/`

average utility (u) of each group:

$$DTR(G_1, G_2 \mid P) = \frac{\exp(G_1 \mid P)/u(G_1 \mid q)}{\exp(G_2 \mid P)/u(G_2 \mid q)} \tag{6.15}$$

where P is the ranking policy. The optimal DTR value is 1: each group is exposed proportional to their utility. Here, as we are minimizing the fairness metric, we always keep the DTR > 1. This means that G_1 and G_2 may change for different queries.

EEL is defined to be the Euclidean distance between the expected exposure each group receives and their target exposure in an ideal scenario where items with the same utility have the same probability of being ranked higher than each other, while all the items with higher utility should always be placed above lower utility items [32]. The optimal value of EEL is 0.

Expectation over sessions. The fairness metrics DTR and EEL are generally calculated as an expectation over multiple sessions of a query. For PPG search, when taking the expectation over N sessions, we concatenate the list of items for each query to itself, N times, and set the inter-group fixed-ranking constraint (Section 6.3.4) to prevent items from different sessions from being mixed up.

6.5.3 Baselines

PL optimization is the state-of-the-art method for black-box optimization of fairness metrics [93, 115], as well as general permutation functions [43]. As PPG is a substitute for PL, the most important baseline in our experiments is PL optimization.[3] Specific to the EEL metric, an in-processing method is proposed in the original EEL paper [32], which cannot be compared to our post-processing method. They also use result randomizations based on PL and show it achieves good fairness-ranking trade-offs. Therefore, to the best of our knowledge, the PL optimization of EEL is the sole state-of-the-art algorithm available in the literature.

For DTR optimization we compare PPG to the state-of-the-art convex optimization method introduced by Singh and Joachims [114], denoted by FOE. This method is a post-processing approach to fair ranking that finds a utility-maximizing marginal probability matrix P that avoids disparate treatment by solving a linear optimiziation problem. To compute the stochastic ranking policy, it uses the Birkhoff-von Neumann algorithm [16] to decompose P into permutation matrices that correspond to rankings. Following [111] we use two variants of FOE based on hard vs. soft doubly stochastic matrix constraints, and call them FOE^H and FOE^S, respectively.[4] Similar to PPG, this model generates a stochastic ranking policy that directly optimizes for DTR, making it a suitable baseline to compare with.

In both the DTR and EEL case, we also include the ranking and fairness performance of the LTR model without any post-processing, as well as the performance of randomizing the items, i.e., picking permutations uniformly at random from S_n, denoted by RAND. Given enough sessions per query, randomized rankings usually have outstanding fairness but relatively bad ranking performance.

[3] We use the public implementation of [43] with some minor modifications.

[4] We used the implementation from `https://github.com/MilkaLichtblau/BA_Laura`.

6.6 Results

We run experiments to address the following research questions:

RQ5.1 Compared to existing methods for ranking fairness optimization, what is the effectiveness of PPG search in finding the fairest ranking for different fairness measures?

RQ5.2 How does PPG search perform compared to PL for queries with a small number of repeating sessions?

RQ5.3 Can the possibility of pairwise constraints in PPG search be used to control other measures, e.g., ranking performance, while optimizing fairness?

RQ5.4 How does an accurate estimate of utility, e.g., the relevance labels, affect different fairness optimization methods?

6.6.1 Fairness Optimization Performance

We first address (RQ5.1) and (RQ5.2) concerning the performance comparison of PPG search to existing fairness optimization methods. Figures 6.3 and 6.4 show performance comparison of our PPG search and PL for optimizing DTR and EEL, respectively, on three datasets TREC 2019, TREC 2020, and MSLR, for different numbers of sessions per query.[5] For both fairness measures, lower means fairer, i.e., better. We omit FOE^H in this figure for better visualization, as it is surpassed by FOE^S in all three tested datasets.

EEL. Regarding (RQ5.1), we observe in Figure 6.4 that for EEL, our PPG method performs better than PL on all three datasets, both being far from the fairness obtained from randomized items. The PL method, after enough sessions, converges to the fairness performance of the LTR output. In terms of ranking performance when optimizing for EEL, PPG is slightly worse than PL on the MSLR and TREC 2019 datasets.

DTR. When optimizing for DTR, PPG and PL both hurt the initial fairness of the LTR output. Here, FOE^S, designed specifically for DTR optimization, has the best performance on all three datasets. On the MSLR and TREC 2020 datasets, PPG and PL perform closely to the randomized items. Comparing PPG and PL to each other, PPG is slightly fairer on the TREC 2019 and MSLR datasets. In the case of DTR optimization for the MSLR dataset, it is interesting to note that randomization slightly hurts fairness of the LTR output, by around 3% relative difference. FOE^S correctly sticks to the LTR output in this case. The message from comparing PPG and PL on the three tested datasets is that, PPG leads to slightly fairer performance than PL when optimized for EEL, while the two perform nearly the same when optimized for DTR.

Small number of sessions. When the number of sessions is limited (RQ5.2), we see a decisive advantage for PPG compared to PL in EEL. For EEL, on all three datasets, PPG converges at the first session, i.e., it finds a deterministic permutation for one session of each query which has a better fairness value than the expected fairness of

[5] We discuss the "PPG + intra" legend in Section 6.6.2.

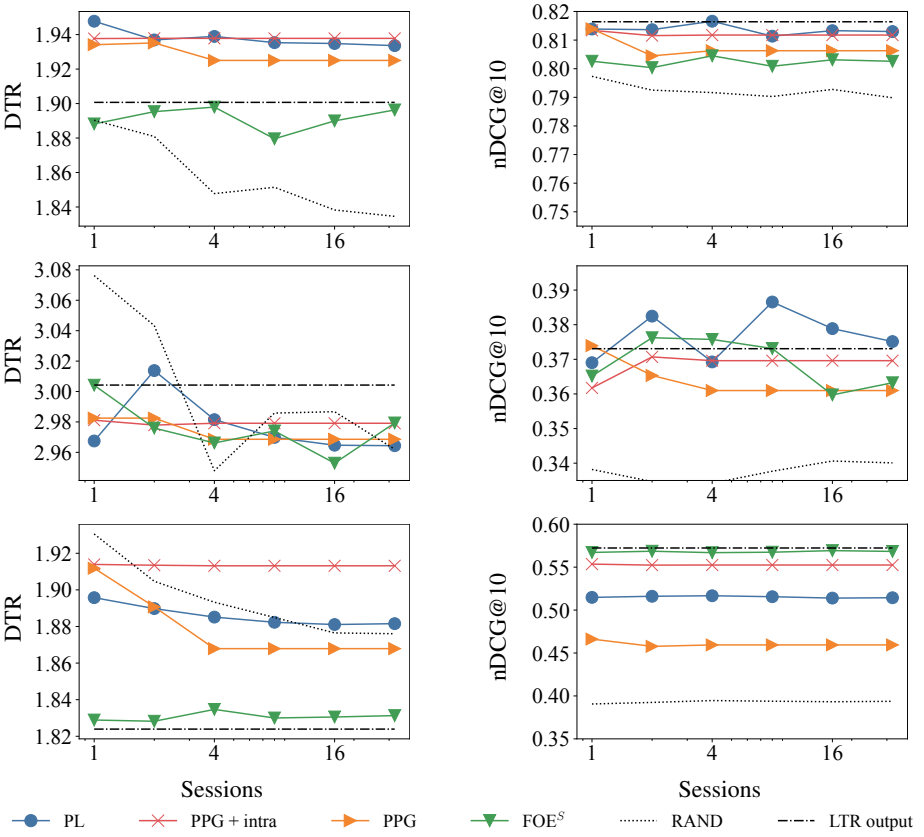


Figure 6.3: *LTR outputs as utility estimates.* Performance comparison of PL and PPG search for optimizing DTR fairness metric (lower is better). Top: TREC 2019; middle: TREC 2020; bottom: MSLR. In all cases PPG search achieves comparable results to the baselines (see Section 6.7 for more details).

PL-generated permutations after 32 sessions. In this sense, PL can be thought of as the **mean**$(\cdot)$ aggregator, whereas PPG is the **min**$(\cdot)$ aggregator; **mean**$(\cdot)$ needs more sessions to converge to the optimum value. For DTR, PPG needs 4 sessions to converge, while PL converges after 8 sessions. Therefore, the answer to (RQ5.2) on our tested datasets is that before a query is repeated many times, PPG is preferred to PL. The reason is that in PPG the reference permutation is set to the best sampled permutation, so in deterministic scenarios, i.e., with only one session per query, PPG is the go-to method.

6.6.2 Pairwise Constraints

To answer (RQ5.3) about the effectiveness of pairwise constraints, we set the intra-group fixed-ranking constraint on a PPG model as follows: The items in each fairness group

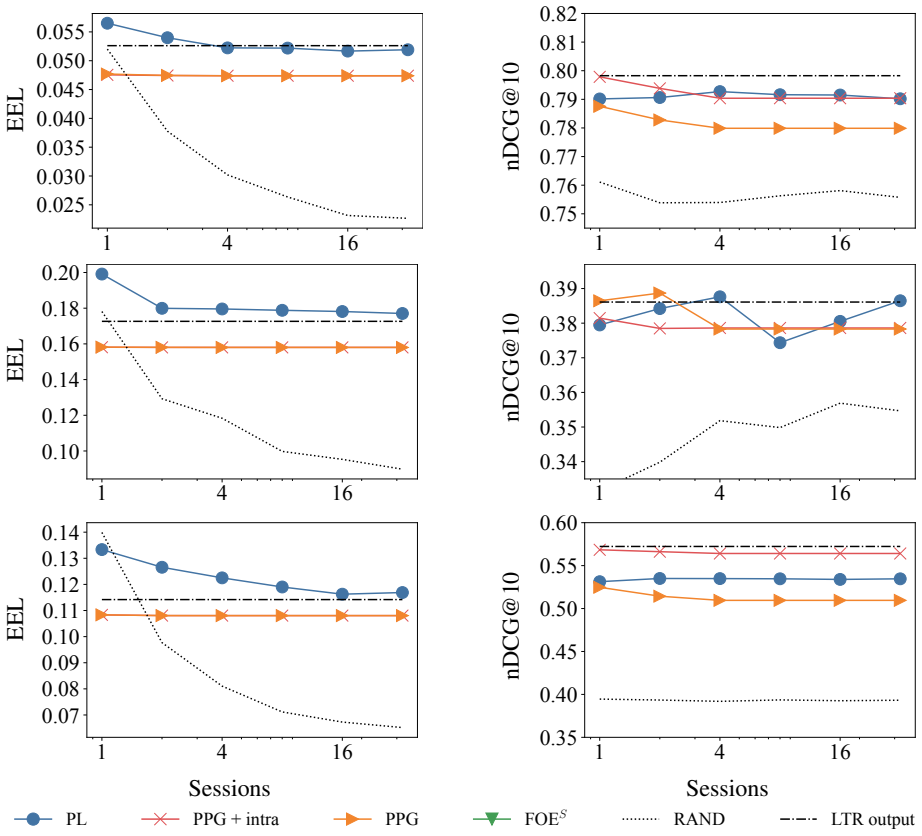


Figure 6.4: *LTR outputs as utility estimates.* Performance comparison of PL and PPG search for optimizing EEL fairness metric (lower is better). Top: TREC 2019; middle: TREC 2020; bottom: MSLR. In all cases PPG search achieves comparable results to the baselines (see Section 6.7 for more details).

are constrained to have the same ordering as the LTR output. This means that the weight of each edge whose endpoints belong to the same fairness group is initialized by zero and consequently will remain zero during training. This is a conservative way of not hurting the ranking performance too much, while searching for a fair ranking (see Section 6.3.4). The "PPG + intra" legend in Figure 6.3 and 6.4 shows the results. In this figure, we see that in all the cases, the ranking performance, measured by nDCG@10, is noticeably improved by adding the intra-group constraint over PPG. This ranking performance improvement comes with a negligible cost of fairness degradation: in the EEL case, "PPG" and "PPG + intra" have the same fairness performance, while for DTR, there is a slight degradation of less than 3% relative difference in fairness performance on all three datasets. So we can answer (RQ5.3) positively: using proper pairwise constraints *does* help to improve other measures, while optimizing for fairness. This possibility is very useful in scenarios where the true relevance labels are unknown and

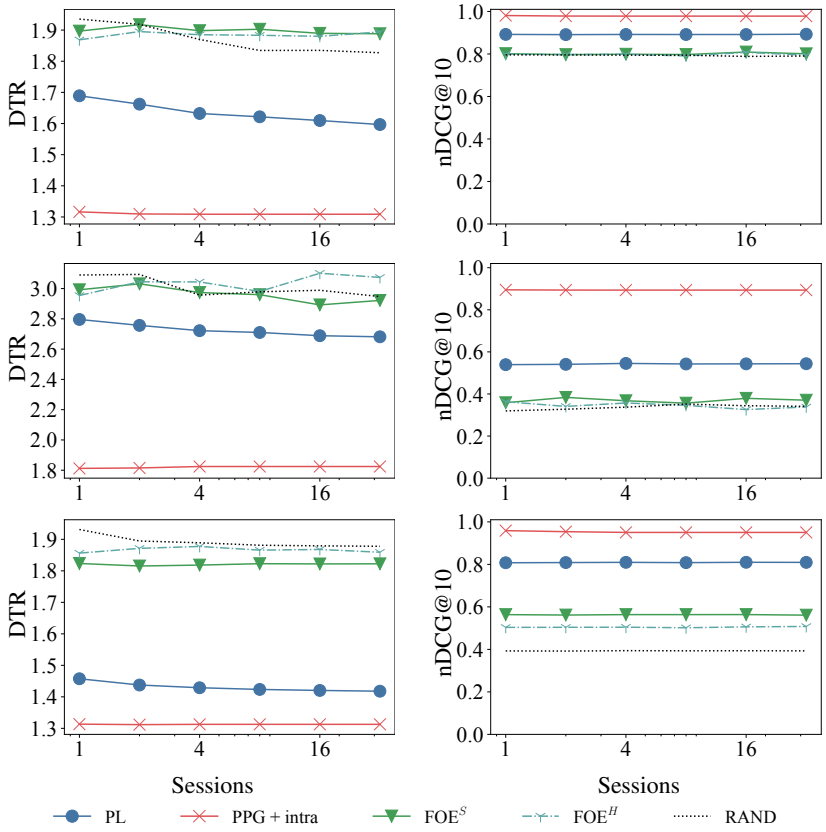


Figure 6.5: *True relevance labels as utility estimates.* Performance comparison of PL and PPG search for optimizing DTR fairness metric. Top: TREC 2019; middle: TREC 2020; bottom: MSLR (see Section 6.7 for more details).

the exact ranking performance cannot be measured. Imposing intra-group fixed-ranking constraints is a conservative way of not hurting our best estimation of the ideal ranking.

6.6.3 Accurate Utility Estimates

One advantage of post-processing for fairness optimization is the possibility of improving future fairness for the torso queries. In this section, we investigate the effectiveness of different fairness optimization methods for specialized queries where highly confident, accurate utility estimates are available through tabular models.

Figures 6.5 and 6.6 show the comparison of different fairness optimization methods when accurate relevance labels are known, for optimizing DTR and EEL, respectively. Here, we only include the results of "PPG + intra", as we have shown its superiority to the simple "PPG" in Section 6.6.2. For brevity, we only mention "PPG" instead of "PPG + intra" in the following. The gap between the performance of PPG to other

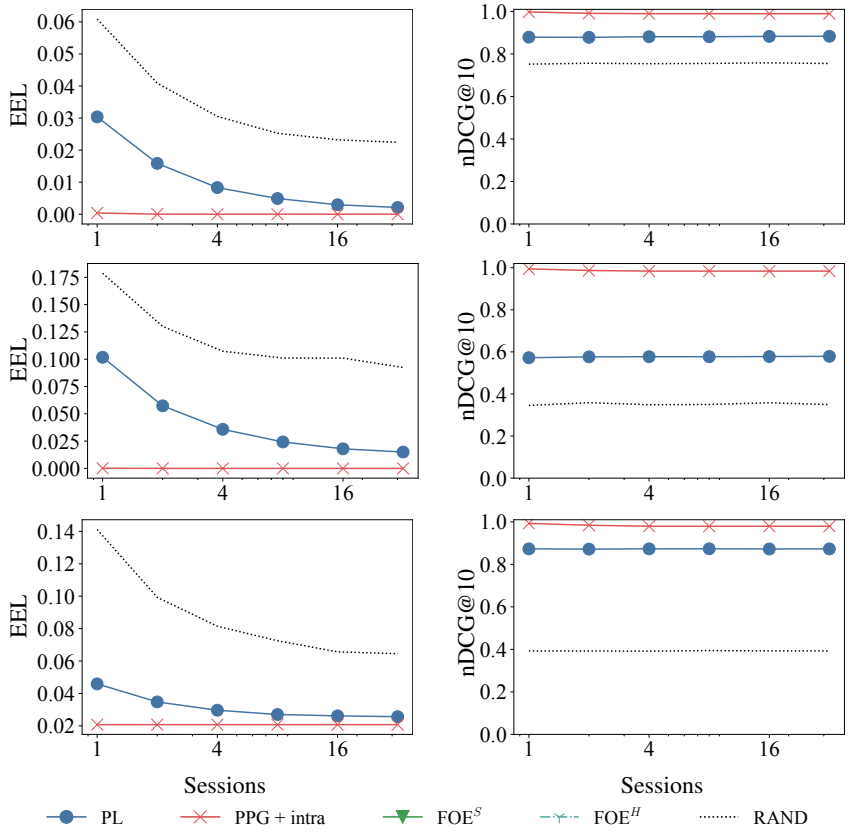


Figure 6.6: *True relevance labels as utility estimates.* Performance comparison of PL and PPG search for optimizing EEL fairness metric. Top: TREC 2019; middle: TREC 2020; bottom: MSLR (see Section 6.7 for more details).

baselines is huge, for both fairness and ranking. PPG is the clear winner in all the tested datasets and both fairness metrics. For EEL, PPG is able to achieve the optimal value of 0. For DTR, PPG comes very close to the optimal value of 1 on the TREC 2019 and MSLR datasets. Compared with FOE^H and FOE^S, we observe that knowledge of accurate utility estimates helps PL and PPG more and causes them to perform better. This is in contrast with our observation with noisy LTR outputs in Section 6.6.1.

In terms of the ranking performance, we observe that the intra-group fixed-ranking has helped PPG to maintain the ranking near ideal, while minimizing the fairness measure.

Table 6.1: Results on 4 sessions with LTR generated labels.

		Fairness		NDCG@10	
	Model	DTR$_\downarrow$	EEL$_\downarrow$	DTR$_\uparrow$	EEL$_\uparrow$
TREC 2019	FOE^S	**1.898**	-	0.805	-
	FOE^H	1.944	-	0.803	-
	PL	1.939	0.052	**0.817**	**0.793**
	PPG	1.925	**0.047**	0.806	0.780
	PPG + intra	1.938	**0.047**	0.812	0.790
	LTR output	1.901	0.053	0.816	0.798
	RAND	1.848	0.030	0.792	0.754
TREC 2020	FOE^S	**2.966**	-	**0.376**	-
	FOE^H	3.047	-	0.371	-
	PL	2.981	0.180	0.369	**0.388**
	PPG	2.969	**0.158**	0.361	0.378
	PPG + intra	2.979	**0.158**	0.370	0.379
	LTR output	3.004	0.173	0.373	0.386
	RAND	2.948	0.118	0.334	0.352
MSLR-qs	FOE^S	**1.835**	-	**0.567**	-
	FOE^H	1.842	-	0.526	-
	PL	1.885	0.123	0.517	0.535
	PPG	1.868	**0.108**	0.459	0.510
	PPG + intra	1.913	**0.108**	0.552	**0.564**
	LTR output	1.824	0.114	0.572	0.572
	RAND	1.893	0.081	0.395	0.392

6.7 Results for Small Number of Sessions

Table 6.1 and 6.2 contain the results of all models on 4 sessions with LTR outputs as utility estimates and knowledge of the true relevance labels, respectively. In the case of a limited number of sessions and LTR outputs as utility estimates (Table 6.1), PPG has a slight advantage compared to PL when optimizing for both EEL and DTR in all three tested datasets. In this case, FOE^S, specifically designed for DTR optimization, leads to the fairest ranking in all three datasets. When the true relevance labels are known (Table 6.2), PPG search outperforms the baselines by a noticeable margin (20%, 33%, and 8% relative decrease for DTR and 100%, 100%, and 30% relative decrease for EEL, compared to the runner up, in TREC 2019, TREC 2020, and MSLR, respectively), obtaining the fairest ranking policies while achieving NDCG scores close to the ideal ranking. Note that the scores reported in Table 6.1 and 6.2 are from two separate experiments which explains the small differences in the results for the RAND baseline.

Table 6.2: Results on 4 sessions with true relevance labels.

	Model	Fairness		NDCG@10	
		DTR$_\downarrow$	EEL$_\downarrow$	DTR$_\uparrow$	EEL$_\uparrow$
TREC 2019	FOE^S	1.898	-	0.799	-
	FOE^H	1.885	-	0.799	-
	PL	1.632	0.008	0.892	0.881
	PPG	1.320	**0.000**	0.881	0.900
	PPG + intra	**1.309**	**0.000**	**0.978**	**0.989**
	Ideal ranking	1.668	0.014	1.000	1.000
	RAND	1.870	0.031	0.794	0.754
TREC 2020	FOE^S	2.974	-	0.368	-
	FOE^H	3.044	-	0.357	-
	PL	2.722	0.036	0.545	0.577
	PPG	2.141	**0.000**	0.574	0.819
	PPG + intra	**1.825**	**0.000**	**0.894**	**0.984**
	Ideal ranking	2.538	0.023	1.000	1.000
	RAND	2.957	0.107	0.337	0.349
MSLR-qs	FOE^S	1.818	-	0.564	-
	FOE^H	1.877	-	0.504	-
	PL	1.429	0.030	0.810	0.873
	PPG	1.395	**0.021**	0.619	0.821
	PPG + intra	**1.313**	**0.021**	**0.950**	**0.979**
	Ideal ranking	1.548	0.027	1.000	1.000
	RAND	1.889	0.081	0.394	0.392

6.8 Conclusion

In this chapter, we have defined probabilistic permutation graph (PPG), a novel representation for permutation distributions. We used PPG as a substitute for Plackett-Luce (PL) in black-box optimization of fairness metrics, using the REINFORCE algorithm. Unlike PL, which is represented by pointwise logits, PPG is constructed by a reference permutation together with pairwise inversion probabilities. The reference permutation of PPG is very useful in deterministic scenarios.

Our experiments show that PPG search, compared to state-of-the-art post-processing fairness optimization methods, is more robust in finding a deterministic fair permutation for one session, while having comparable performance for expected fairness over larger numbers of sessions. This means that PPG search can be considered as an answer to RQ5. The pairwise inversion probabilities also allow us to impose pairwise constraints that can control other objectives while optimizing for fairness. We experimentally verified the effectiveness of such pairwise constraints on controlling the ranking performance. Finally, we have shown that in scenarios such as tabular search, where high-quality

estimates of utility are available, PPG performs outstandingly well, with a considerable gap to other state-of-the-art fairness optimization methods.

As future work, it would be interesting to test the effectiveness of PPG on other fairness metrics as well as other general permutation optimization problems. Another possible line of work is to apply PPG for an in-processing method as a substitute for PL.

This concludes the final research chapter of this thesis. Next, we will conclude the thesis by summarizing the answers to our research questions and proposing possible directions for future work.

7 Conclusions

In this thesis, we re-examined a number of existing assumptions in unbiased and fair counterfactual learning to rank (CLTR). In preceding chapters, in terms of unbiased CLTR we have discussed: (i) adapting inverse propensity scoring (IPS) for cascade-based models (CBM) of user browsing; (ii) a generalization to IPS to also correct for trust bias in addition to position bias; and (iii) breaking the cyclic dependency between bias parameter and relevance estimation. Then, in terms of fairness of ranking we have seen: (iv) the impact of group membership bias on the quality and fairness of ranking; and (v) a novel representation for permutation distribution that allows for fairness optimization both in deterministic and stochastic ranking systems. In this chapter, we review the main findings of these topics and propose directions for future work.

7.1 Main Findings

In this section, we revisit the research questions that were posed in Chapter 1 and summarize the most important findings.

RQ1 How to go beyond position-based model (PBM) and consider user cascading behavior when using IPS for position bias correction?

Through a number of semi-synthetic experiments, we confirmed that when the user behavior follows a cascade model, PBM-IPS is not helpful for position bias correction, in that it has a gap toward the full information case. Then, we derived closed-form formulas for click propensities in three widely used cascade-based click models, to fill in the gaps of the PBM-IPS performance on cascade-based clicks and have experimentally shown the effectiveness of our CM-IPS to correct for position bias.

RQ2 How to effectively correct for trust bias in user click data?

The answer to this question is not "IPS" as we have proven that no IPS estimator can correct for trust bias. Consequently, we have introduced the novel affine correction (AC) method, which both reweights clicks and penalizes items for being displayed at ranks where the users' trust is high. We proved that the AC is unbiased w.r.t. both position bias and trust bias, thus it is the first CLTR method that can deal with both of these

biases simultaneously. We have confirmed our theoretical findings by semi-synthetic experiments on two LTR datasets, showing that with AC the ranking performance can reach to full information case, while with IPS it cannot.

RQ3 Is it possible to break the cyclic dependency between relevance and bias parameters when correcting for position and trust bias?

We proposed a novel correction method, mixture-based correction (MBC), as an answer to this question. Unlike IPS and AC methods, in which the unbiasedness relies on accurate bias parameters estimation, the unbiasedness proof of MBC does not rely on knowledge of relevance. We have shown that the cyclic dependency in the existing methods that use regression-based EM (rbEM), leads to at least three practical limitations: (i) Severe sensitivity to the choice of the regression function; (ii) EM not necessarily converging towards the zero gradient; and, (iii) Low efficiency due to repeated use of the regression function. MBC solves all of these limitations as a side benefit. We have also inspected our findings regarding IPS from RQ1 for the case of MBC. We have observed that when clicks adhere to cascading models, while PBM is assumed by the correction methods, both MBC and AC will remain biased, but MBC is more robust, in the sense that its ranking performance is affected less compared to AC.

RQ4 What is the impact of group bias on the quality and fairness metrics in a ranking and how to correct for this bias, without substituting equality for equity?

We have theoretically shown that, in the presence of group bias, NDCG and DTR change linearly with the fraction of affected relevant items that are still as attractive as the non-affected relevant items, while the change in EEL w.r.t. group bias has a logarithmic form. We also confirmed these effects on the quality and fairness of ranking through extensive experiments on various datasets. Regarding the bias correction part, we argued that correcting for group bias, which is a type of *content-based* bias, is not as easy as *context-based* types of bias such as position and trust bias. To measure group bias, assumptions based on fairness constraints should be made about the utility distribution of different groups. However, such assumptions can potentially make the equity-based notion of fairness meaningless. We proposed to solve this issue by amortized correction for the group bias, as global equality does not contradict local equity. We have experimentally confirmed that our correction method, when its assumptions are met, is able to fully recover the scores suffering from group bias, in the sense that the ranking and fairness metrics after correction achieve the values of the full information case.

RQ5 Is it possible to have a single general fairness optimization method that performs well for both stochastic and deterministic rankings?

We have introduced a new representation for the distribution of permutations, called probabilistic permutation graph (PPG), constructed by pairwise inversion probabilities instead of the pointwise logits in the Plackett-Luce (PL) representation. Our experiments show that our PPG search method, compared to state-of-the-art post-processing fairness optimization methods, is more robust in finding a deterministic fair permutation for one session, while having comparable performance for expected fairness over larger

numbers of sessions. The pairwise inversion probabilities also allow us to impose pairwise constraints that can control other objectives while optimizing for fairness. We experimentally verified the effectiveness of such pairwise constraints on controlling the ranking performance. Finally, we have shown that in scenarios such as tabular search, where high-quality estimates of utility are available, PPG performs outstandingly well, with a considerable gap to other state-of-the-art fairness optimization methods.

7.2 Future Work

In this thesis, we have provided several solutions to make unbiased and fair CLTR methods applicable in wider scenarios, i.e., RQ2 and RQ5, or with more realistic assumptions, i.e., RQ1, RQ3, and RQ4, compared to existing work. Here, we list a number of limitations and potential future directions for these solutions.

In Chapter 2 we derive formulas for three cascade-based models and our experiments covered the results for one of them. There are other more complex user browsing models in the literature [e.g., 17, 19, 78, 80] for which extending the IPS formulation may or may not be straightforward. Exploring the interplay of more complex click models and IPS or AC could be and interesting future direction.

Our affine correction (AC) method to correct for trust bias in Chapter 3 is more prone to the high variance problem compared to IPS. The reason is that the denumerator in AC is strictly less than the denumerator in IPS for all of the ranking positions. Furthermore, as a result of penalizing incorrect clicks, corrected clicks in AC can have negative values in contrast to IPS. While in terms of bias in the presence of trust bias AC is favored to IPS, the aforementioned characteristics of AC may show high variances in practice, especially for a low number of training clicks. As a future direction, variance reduction solutions, similar to propensity clipping [119], could prove effective for the AC method. Recent work in [94] is an example of this line where AC is extended to the doubly-robust estimator for CLTR with a provable lower variance than AC itself.

In Chapter 4, our mixture-based correction (MBC) solution to break the cyclic dependency between bias parameters and relevance estimation is based only on the click-through rate (CTR), and the content features of the query and items are not used for correction. Adding features to the correction process can be an interesting area that is already initiated by two-tower models [136] as well as decomposed ranking debiasing [142].

Our analysis regarding group bias in Chapter 5 considers feedback from users in the form of judgements, e.g., clicks, job hires, etc. Increasingly, group bias can affect more complex signals such as the tone or content of answers in a conversation. With the prevalent use of pre-trained large language models (LLM), an interesting topic would be to study the impact of group biased text, as the pre-training data or the examples for in-context learning, on the quality and fairness of LLMs for downstream tasks. User studies similar to [67, 107] with the aim of modeling group bias more precisely than what we did using a multiplicative factor, could also lead to valuable lessons for the community.

In Chapter 6 we presented an approximate but efficient method for sampling from a probabilistic permutation graph (PPG) distribution. One line of theoretical continuation

would be to devise efficient sampling methods with tighter error bounds than our proposed method. We used PPG for post-processing fairness optimization. Another future direction is to apply PPG as an in-processing method as a substitute for PL. More generally, as ranking fairness optimization is a special case of permutation optimization where PL is a well-established representation, our work can be extended to see how PPG representation compares to PL in terms of finding the optimum permutation in general permutation optimization.

In this thesis, we have revisited several assumptions present in current unbiased and fair counterfactual learning to rank methods. These insights can serve as a foundation for questioning other prevalent assumptions in this field. For instance, we can question the assumption of uniform examination, which assumes that users from various groups examine items with the same probabilities. Additionally, we can think of the relevance uniformity assumption, which considers interactions from all user groups as equally informative and accurate. Looking ahead, we can expand the application of the ideas explored in this thesis beyond information retrieval. This may involve ranking video segments for video summarization based on user views or ranking user reviews for opinion mining. Furthermore, a future direction involves investigating the extent to which we can influence or alleviate user biases without violating privacy concerns. For example, we can explore the impact of combining various contextual factors, such as item position and visual effects, to encourage a more equitable distribution of examination probabilities. Alternatively, we can analyze the effectiveness of displaying more results from minority groups at the top of the ranking in reducing group bias over time, and also its potential damages to user satisfaction.

Bibliography

[1] A. Agarwal, K. Takatsu, I. Zaitsev, and T. Joachims. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 5–14. ACM, 2019. (Cited on pages 1, 20, 23, and 34.)

[2] A. Agarwal, X. Wang, C. Li, M. Bendersky, and M. Najork. Addressing trust bias for unbiased learning-to-rank. In *The World Wide Web Conference*, pages 4–14. ACM, 2019. (Cited on pages 2, 3, 13, 20, 23, 24, 28, 31, 32, 33, 34, 35, 38, 42, 43, 44, 46, 47, 48, 49, 50, 58, and 68.)

[3] A. Agarwal, I. Zaitsev, X. Wang, C. Li, M. Najork, and T. Joachims. Estimating position bias without intrusive interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 474–482. ACM, 2019. (Cited on pages 20, 23, 34, and 39.)

[4] Q. Ai, K. Bi, C. Luo, J. Guo, and W. B. Croft. Unbiased learning to rank with unbiased propensity estimation. In *Proceedings of the 41st International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 385–394. ACM, 2018. (Cited on pages 2, 11, 12, 13, 14, 16, 20, 23, 28, 32, 34, 45, 46, 47, 48, 49, 50, and 64.)

[5] Q. Ai, X. Wang, S. Bruch, N. Golbandi, M. Bendersky, and M. Najork. Learning groupwise multivariate scoring functions using deep neural networks. In *SIGIR*, pages 85–92, 2019. (Cited on page 16.)

[6] A. Arampatzis, S. Robertson, and J. Kamps. Score distributions in information retrieval. In *Advances in Information Retrieval Theory*, pages 139–151. Springer, 2009. (Cited on pages 42 and 45.)

[7] A. Asudeh, H. Jagadish, J. Stoyanovich, and G. Das. Designing fair ranking schemes. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD, pages 1259–1276, 2019. (Cited on page 86.)

[8] R. Baeza-Yates. Bias on the web. *Communications of the ACM*, 61(6):54–61, 2018. (Cited on page 86.)

[9] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999. (Cited on page 1.)

[10] S. Baswana, P. P. Chakrabarti, S. Chandran, Y. Kanoria, and U. Patange. Centralized admissions for engineering colleges in India. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, pages 323–324, 2019. (Cited on page 74.)

[11] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016. (Cited on page 84.)

[12] A. Berny. Selection and reinforcement learning for combinatorial optimization. In *International Conference on Parallel Problem Solving from Nature*, pages 601–610. Springer, 2000. (Cited on page 87.)

[13] A. J. Biega, K. P. Gummadi, and G. Weikum. Equity of attention: Amortizing individual fairness in rankings. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pages 405–414, 2018. (Cited on pages 67, 68, and 86.)

[14] A. J. Biega, F. Diaz, M. D. Ekstrand, and S. Kohlmeier. Overview of the TREC 2019 fair ranking track. In *The Twenty-Eighth Text REtrieval Conference (TREC 2019) Proceedings*, 2019. (Cited on pages 74 and 96.)

[15] A. J. Biega, F. Diaz, M. D. Ekstrand, S. Feldman, and S. Kohlmeier. Overview of the TREC 2020 fair ranking track. In *The Twenty-Ninth Text REtrieval Conference Proceedings (TREC 2020)*, 2021. (Cited on page 65.)

[16] G. Birkhoff. *Lattice Theory*. AMS, 1940. (Cited on page 97.)

[17] K. Bisht and S. Susan. Weighted ensemble of neural and probabilistic graphical models for click prediction. In *2021 the 5th international conference on information system and data mining*, pages 145–150, 2021. (Cited on page 109.)

[18] H. Bonab, M. Aliannejadi, A. Vardasbi, E. Kanoulas, and J. Allan. Cross-market product recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 110–119, 2021.

[19] A. Borisov, I. Markov, M. de Rijke, and P. Serdyukov. A neural click model for web search. In *Proceedings of the 25th International Conference on World Wide Web*, pages 531–541. International World Wide Web Conferences Steering Committee, 2016. (Cited on page 109.)

[20] M. Brownstein. Implicit bias. In *Stanford Encyclopedia of Philosophy*. 2017. (Cited on page 66.)

[21] C. J. Burges. From RankNet to LambdaRank to LambdaMART: An overview. Technical Report MSR-TR-2010-82, Microsoft, 2010. (Cited on pages 49 and 50.)

[22] L. E. Celis, D. Straszak, and N. K. Vishnoi. Ranking with fairness constraints. In *ICALP*, pages 28:1–28:15, 2018. (Cited on page 86.)

[23] L. E. Celis, A. Mehrotra, and N. K. Vishnoi. Interventions for ranking in the presence of implicit

bias. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, pages 369–380, 2020. (Cited on pages 64, 66, 67, 68, 72, 73, 74, and 86.)

[24] P. Chandar and B. Carterette. Estimating clickthrough bias in the cascade model. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1587–1590, 2018. (Cited on pages 14 and 45.)

[25] O. Chapelle and Y. Chang. Yahoo! Learning to rank challenge overview. In *Proceedings of the Learning to Rank Challenge*, volume 14 of *Proceedings of Machine Learning Research*, pages 1–24. PMLR, 25 Jun 2011. (Cited on pages 1, 16, 22, 33, 49, and 74.)

[26] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th International Conference on World Wide Web*, pages 1–10. ACM, 2009. (Cited on pages 3, 12, and 44.)

[27] D. Chen, W. Chen, H. Wang, Z. Chen, and Q. Yang. Beyond ten blue links: Enabling user click modeling in federated web search. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, pages 463–472, 2012. (Cited on page 42.)

[28] X. Chen and Y. Yang. Cutoff for exact recovery of gaussian mixture models. *arXiv preprint arXiv:2001.01194*, 2020. (Cited on page 48.)

[29] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Morgan & Claypool Publishers, 2015. (Cited on pages 11, 12, 13, 15, 17, 43, 47, 48, and 67.)

[30] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*, pages 87–94, 2008. (Cited on pages 2, 12, 19, and 22.)

[31] X. Dai, J. Hou, Q. Liu, Y. Xi, R. Tang, W. Zhang, X. He, J. Wang, and Y. Yu. U-rank: Utility-oriented learning to rank with implicit feedback. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020. (Cited on page 45.)

[32] F. Diaz, B. Mitra, M. D. Ekstrand, A. J. Biega, and B. Carterette. Evaluating stochastic rankings with expected exposure. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, pages 275–284, 2020. (Cited on pages 4, 64, 65, 74, 84, 86, 96, and 97.)

[33] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011. (Cited on page 34.)

[34] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 331–338, 2008. (Cited on pages 47 and 55.)

[35] B. Dushnik and E. W. Miller. Partially ordered sets. *American Journal of Mathematics*, 63(3):600–610, 1941. (Cited on pages 85 and 95.)

[36] J. Ehrhardt, T. Spinde, A. Vardasbi, and F. Hamborg. Omission of information: Identifying political slant via an analysis of co-occurring entities. *Schmidt, Wolff (Eds.): Information between Data and Knowledge*, pages 80–93, 2021.

[37] M. D. Ekstrand, G. McDonald, A. Raj, and I. Johnson. Overview of the TREC 2021 fair ranking track. In *The Thirtieth Text REtrieval Conference (TREC 2021) Proceedings*, 2022. (Cited on page 70.)

[38] V. Emelianov, N. Gast, K. P. Gummadi, and P. Loiseau. On fair selection in the presence of implicit variance. In *Proceedings of the 21st ACM Conference on Economics and Computation*, EC '20, pages 649–675, 2020. (Cited on pages 66, 67, 72, and 73.)

[39] A. Fabris, A. Purpura, G. Silvello, and G. A. Susto. Gender stereotype reinforcement: Measuring the gender bias conveyed by ranking algorithms. *Information Processing & Management*, 57(6):102377, 2020. (Cited on page 64.)

[40] Z. Fang, A. Agarwal, and T. Joachims. Intervention harvesting for context-dependent examination-bias estimation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 825–834, 2019. (Cited on pages 20, 23, and 34.)

[41] C. FitzGerald and S. Hurst. Implicit bias in healthcare professionals: A systematic review. *BMC medical ethics*, 18(1):1–18, 2017. (Cited on page 66.)

[42] M. A. Fligner and J. S. Verducci. Multistage ranking models. *Journal of the American Statistical Association*, 83(403):892–901, 1988. (Cited on page 87.)

[43] A. Gadetsky, K. Struminsky, C. Robinson, N. Quadrianto, and D. Vetrov. Low-variance black-box gradient estimates for the Plackett-Luce distribution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10126–10135, 2020. (Cited on pages 5, 84, 87, and 97.)

[44] S. C. Geyik, S. Ambler, and K. Kenthapadi. Fairness-aware ranking in search & recommendation systems with application to linkedin talent search. In *Proceedings of the 25th ACM SIGKDD Inter-*

national Conference on Knowledge Discovery & Data Mining, pages 2221–2231, 2019. (Cited on page 86.)

[45] T. Grainger. *AI Powered Search*. Manning Publications, 2021. (Cited on page 88.)

[46] A. Grotov and M. de Rijke. Online learning to rank for information retrieval: SIGIR 2016 tutorial. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 1215–1218, 2016. (Cited on page 1.)

[47] A. Grotov, A. Chuklin, I. Markov, L. Stout, F. Xumara, and M. de Rijke. A comparative study of click models for web search. In *CLEF*, pages 78–90. Springer, 2015. (Cited on page 12.)

[48] E. J. Gumbel. *Statistical Theory of Extreme Values and Some Practical Applications: A Series of Lectures*, volume 33. US Government Printing Office, 1954. (Cited on page 87.)

[49] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y.-M. Wang, and C. Faloutsos. Click chain model in web search. In *International World Wide Web Conference*, pages 11–20. ACM, 2009. (Cited on pages 3 and 13.)

[50] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 124–131. ACM, 2009. (Cited on pages 3, 12, 16, 17, 47, and 55.)

[51] S. Hajian, F. Bonchi, and C. Castillo. Algorithmic bias: From discrimination discovery to fairness-aware data mining. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 2125–2126, 2016. (Cited on page 64.)

[52] M. Heuss, F. Sarvi, and M. de Rijke. Fairness of exposure in light of incomplete exposure estimation. In *SIGIR 2022: 45th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 759–769. ACM, July 2022. (Cited on pages 86 and 96.)

[53] K. Hofmann, S. Whiteson, and M. de Rijke. Balancing exploration and exploitation in learning to rank online. In *Advances in Information Retrieval*, pages 251–263. Springer, April 2011. (Cited on page 49.)

[54] Z. Hu, Y. Wang, Q. Peng, and H. Li. Unbiased LambdaMART: An unbiased pairwise learning-to-rank algorithm. In *The World Wide Web Conference*, pages 2830–2836. ACM, 2019. (Cited on page 25.)

[55] R. Jagerman, H. Oosterhuis, and M. de Rijke. To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions. In *Proceedings of the 42nd International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 15–24. ACM, 2019. (Cited on pages 32, 33, 48, 49, 50, and 74.)

[56] 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. (Cited on page 23.)

[57] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 154–161. ACM, 2005. (Cited on pages 2, 3, 20, 23, 43, and 64.)

[58] 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. (Cited on pages 1, 2, 11, 12, 13, 14, 16, 20, 22, 23, 28, 32, 33, 34, 41, 42, 46, 47, 48, 49, 50, 64, 66, 68, 72, 74, 85, and 96.)

[59] C. Jolls and C. R. Sunstein. The law of implicit bias. *Calif. L. Rev.*, 94:969, 2006. (Cited on page 66.)

[60] S. Katariya, B. Kveton, C. Szepesvari, and Z. Wen. DCM bandits: Learning to rank with multiple clicks. In *International Conference on Machine Learning*, pages 1215–1224, 2016. (Cited on pages 68 and 88.)

[61] M. Kay, C. Matuszek, and S. A. Munson. Unequal representation and gender stereotypes in image search results for occupations. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3819–3828, 2015. (Cited on pages 4, 64, 66, and 67.)

[62] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017. (Cited on page 49.)

[63] Ö. Kırnap, F. Diaz, A. Biega, M. Ekstrand, B. Carterette, and E. Yilmaz. Estimation of fair ranking metrics with incomplete judgments. In *Proceedings of the Web Conference 2021*, pages 1065–1075, 2021. (Cited on page 96.)

[64] J. Kleinberg and M. Raghavan. Selection problems in the presence of implicit bias. In *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018. (Cited on pages 64, 66, 67, 68, 72, and 73.)

[65] J. Kleinberg, S. Mullainathan, and M. Raghavan. Inherent trade-offs in the fair determination of risk scores. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 43:1–43:23. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. (Cited on page 84.)

[66] K. Krieg, E. Parada-Cabaleiro, G. Medicus, O. Lesota, M. Schedl, and N. Rekabsaz. Grep-BiasIR: A dataset for investigating gender representation-bias in information retrieval results. *arXiv preprint arXiv:2201.07754*, 2022. (Cited on pages 67 and 73.)

[67] K. Krieg, E. Parada-Cabaleiro, M. Schedl, and N. Rekabsaz. Do perceived gender biases in retrieval results affect relevance judgements? *arXiv preprint arXiv:2203.01731*, 2022. (Cited on pages 4, 64, 67, and 109.)

[68] M. R. Kumar. Rti complaint, 2009. Decision No. CIC/SG/C/2009/001088/5392, Complaint No. CIC/SG/C/2009/001088. (Cited on page 74.)

[69] B. Kveton, C. Szepesvari, Z. Wen, and A. Ashkan. Cascading bandits: Learning to rank in the cascade model. In *International Conference on Machine Learning*, pages 767–776, 2015. (Cited on page 68.)

[70] P. Lagrée, C. Vernade, and O. Cappé. Multiple-play bandits in the position-based model. In *Advances in Neural Information Processing Systems*, pages 1597–1605, 2016. (Cited on page 88.)

[71] T. Lattimore and C. Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020. (Cited on page 68.)

[72] D. Lefortier, P. Serdyukov, and M. de Rijke. Online exploration for detecting shifts in fresh intent. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, page 589–598, 2014. (Cited on page 1.)

[73] C. Li, H. Feng, and M. de Rijke. A contextual-bandit approach to online learning to rank for relevance and diversity. *CoRR*, 2019. (Cited on page 1.)

[74] C. Li, B. Kveton, T. Lattimore, I. Markov, M. de Rijke, C. Szepesvári, and M. Zoghi. Bubblerank: Safe online learning to re-rank via implicit click feedback. In *Uncertainty in Artificial Intelligence*, pages 196–206. PMLR, 2020. (Cited on pages 68, 85, and 88.)

[75] M. Li, A. Vardasbi, A. Yates, and M. de Rijke. Repetition and exploration in sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2532–2541, 2023.

[76] S. Li, T. Lattimore, and C. Szepesvári. Online learning to rank with features. In *International Conference on Machine Learning*, pages 3856–3865. PMLR, 2019. (Cited on page 1.)

[77] S.-S. Liaw and H.-M. Huang. An investigation of user attitudes toward search engines as an information retrieval tool. *Computers in Human Behavior*, 19(6):751–765, 2003. (Cited on page 1.)

[78] J. Lin, W. Liu, X. Dai, W. Zhang, S. Li, R. Tang, X. He, J. Hao, and Y. Yu. A graph-enhanced click model for web search. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 1259–1268, 2021. (Cited on page 109.)

[79] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009. (Cited on pages 1, 3, 11, and 22.)

[80] Y. Liu, X. Xie, C. Wang, J.-Y. Nie, M. Zhang, and S. Ma. Time-aware click model. *ACM Transactions on Information Systems (TOIS)*, 35(3):1–24, 2016. (Cited on page 109.)

[81] Y. Lu and H. H. Zhou. Statistical and computational guarantees of Lloyd's algorithm and its variants. *arXiv preprint arXiv:1612.02099*, 2016. (Cited on page 48.)

[82] R. D. Luce. *Individual Choice Behavior*. John Wiley, 1959. (Cited on page 87.)

[83] K. S. Lyness and M. E. Heilman. When fit is fundamental: Performance evaluations and promotions of upper-level female and male managers. *Journal of Applied Psychology*, 91(4):777, 2006. (Cited on page 66.)

[84] C. J. Maddison, D. Tarlow, and T. Minka. A* sampling. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. (Cited on page 87.)

[85] F. J. Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951. (Cited on page 72.)

[86] T. Mavridis, S. Hausl, A. Mende, and R. Pagano. Beyond algorithms: Ranking at scale at booking.com. In *ComplexRec-ImpactRS@RecSys*, 2020. (Cited on page 88.)

[87] R. M. McConnell and J. P. Spinrad. Modular decomposition and transitive orientation. *Discrete Mathematics*, 201(1):189–241, 1999. (Cited on page 89.)

[88] G. J. McLachlan and K. E. Basford. *Mixture models: Inference and applications to clustering*, volume 38. M. Dekker New York, 1988. (Cited on page 47.)

[89] A. Mehrotra, B. S. R. Pradelski, and N. K. Vishnoi. Selection in the presence of implicit bias: The advantage of intersectional constraints. In *2022 ACM Conference on Fairness, Accountability, and*

Transparency, FAccT '22, pages 599–609, 2022. (Cited on page 68.)
[90] R. Mehrotra, J. McInerney, H. Bouchard, M. Lalmas, and F. Diaz. Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness & satisfaction in recommendation systems. In *Proceedings of the 27th ACM International Conference on Information & Knowledge Management*, pages 2243–2251, 2018. (Cited on page 86.)
[91] P. Molenberghs. The neuroscience of in-group bias. *Neuroscience & Biobehavioral Reviews*, 37(8): 1530–1536, 2013. (Cited on page 67.)
[92] M. Morik, A. Singh, J. Hong, and T. Joachims. Controlling fairness and bias in dynamic learning-to-rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pages 429–438, 2020. (Cited on pages 68 and 86.)
[93] H. Oosterhuis. Computationally efficient optimization of Plackett-Luce ranking models for relevance and fairness. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1023–1032, 2021. (Cited on pages 87, 96, and 97.)
[94] H. Oosterhuis. Doubly robust estimation for correcting position bias in click feedback for unbiased learning to rank. *ACM Transactions on Information Systems*, 41(3):1–33, 2023. (Cited on page 109.)
[95] H. Oosterhuis and M. de Rijke. Differentiable unbiased online learning to rank. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1293–1302. ACM, 2018. (Cited on pages 1, 11, and 48.)
[96] H. Oosterhuis and M. de Rijke. Policy-aware unbiased learning to rank for top-k rankings. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 489–498. ACM, 2020. (Cited on pages 2, 19, 20, 23, 28, 32, 33, 34, 39, 46, 47, 48, 49, and 50.)
[97] H. Oosterhuis and M. de Rijke. Robust generalization and safe query-specializationin counterfactual learning to rank. In *Proceedings of the Web Conference 2021*, pages 158–170, 2021. (Cited on pages 64 and 88.)
[98] H. Oosterhuis and M. de Rijke. Unifying online and counterfactual learning to rank: A novel counterfactual estimator that effectively utilizes online interventions. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 463–471. ACM, 2021. (Cited on page 50.)
[99] Z. Ovaisi, R. Ahsan, Y. Zhang, K. Vasilaky, and E. Zheleva. Correcting for selection bias in learning-to-rank systems. *arXiv preprint arXiv:2001.11358*, 2020. (Cited on pages 2, 19, 20, 23, and 28.)
[100] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975. (Cited on page 87.)
[101] P. Pobrotyn, T. Bartczak, M. Synowiec, R. Bialobrzeski, and J. Bojar. Context-aware learning to rank with self-attention. *arXiv preprint arXiv:2005.10084*, 2020. (Cited on page 75.)
[102] T. Qin and T. Liu. Introducing LETOR 4.0 datasets. *CoRR*, abs/1306.2597, 2013. (Cited on pages 1, 22, 33, 49, 74, and 96.)
[103] Z. Qin, S. J. Chen, D. Metzler, Y. Noh, J. Qin, and X. Wang. Attribute-based propensity for unbiased learning in recommender systems: Algorithm and case studies. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2359–2367. ACM, 2020. (Cited on page 45.)
[104] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th International Conference on Machine Learning*, pages 784–791, 2008. (Cited on page 68.)
[105] A. Raj and M. D. Ekstrand. Measuring fairness in ranked results: An analytical and empirical comparison. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 726–736, 2022. (Cited on page 64.)
[106] A. Raj, C. Wood, A. Montoly, and M. D. Ekstrand. Comparing fair ranking metrics. *arXiv preprint arXiv:2009.01311*, 2020. (Cited on page 84.)
[107] A. Raj, B. Mitra, N. Craswell, and M. Ekstrand. Patterns of gender-specializing query reformulation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2241–2245, 2023. (Cited on page 109.)
[108] Y. Saito and T. Joachims. Fair ranking as fair division: Impact-based individual fairness in ranking. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pages 1514–1524, 2022. (Cited on page 82.)
[109] M. Sanderson. Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4(4):247–375, 2010. (Cited on pages 1 and 22.)
[110] P. Sapiezynski, W. Zeng, R. E Robertson, A. Mislove, and C. Wilson. Quantifying the impact of user

attentionon fair group representation in ranked lists. In *International World Wide Web Conference*, pages 553–562, 2019. (Cited on page 86.)

[111] F. Sarvi, M. Heuss, M. Aliannejadi, S. Schelter, and M. de Rijke. Understanding and mitigating the effect of outliers in fair ranking. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, page 861–869, 2022. (Cited on pages 74, 86, 96, and 97.)

[112] F. Sarvi, A. Vardasbi, M. Aliannejadi, S. Schelter, and M. de Rijke. On the impact of outlier bias on user clicks. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 18–27, 2023.

[113] H. Schutze, C. D. Manning, and P. Raghavan. *Introduction to information retrieval*. Cambridge University Press, 2008. (Cited on page 1.)

[114] A. Singh and T. Joachims. Fairness of exposure in rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 2219–2228, 2018. (Cited on pages 2, 4, 64, 65, 68, 84, 86, 87, 96, and 97.)

[115] A. Singh and T. Joachims. Policy learning for fairness in ranking. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. (Cited on pages 4, 68, 84, 86, 87, and 97.)

[116] D. Sorokina and E. Cantu-Paz. Amazon search: The joy of ranking products. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 459–460, 2016. (Cited on pages 85 and 88.)

[117] J. Stoyanovich, K. Yang, and H. Jagadish. Online set selection with fairness and diversity constraints. In *21st International Conference on Extending Database Technology*, EDBT, pages 241–252, 2018. (Cited on page 86.)

[118] T. Sühr, S. Hilgard, and H. Lakkaraju. Does fair ranking improve minority outcomes? understanding the interplay of human and algorithmic biases in online hiring. In *Conference on Artificial Intelligence, Ethics and Society*, AIES '21, pages 989–999, 2021. (Cited on pages 4, 64, and 66.)

[119] A. Swaminathan and T. Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research*, 16(1):1731–1755, 2015. (Cited on pages 16, 36, and 109.)

[120] A. Trotman, J. Degenhardt, and S. Kallumadi. The architecture of ebay search. In *eCOM@ SIGIR*, 2017. (Cited on page 88.)

[121] A. Vardasbi, M. de Rijke, and I. Markov. Cascade model-based propensity estimation for counterfactual learning to rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pages 2089–2092, 2020. (Cited on pages 45, 48, and 68.)

[122] A. Vardasbi, H. Oosterhuis, and M. de Rijke. When inverse propensity scoring does not work: Affine corrections for unbiased learning to rank. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, pages 1475–1484, 2020. (Cited on pages 3, 44, 46, 47, 48, 49, 50, 58, 74, and 96.)

[123] A. Vardasbi, M. de Rijke, and I. Markov. Mixture-based correction for position and trust bias in counterfactual learning to rank. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 1869–1878, 2021. (Cited on pages 68, 72, 74, 85, and 96.)

[124] A. Vardasbi, M. de Rijke, and M. Dehghani. Intersection of parallels as an early stopping criterion. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, pages 1965–1974, 2022.

[125] A. Vardasbi, F. Sarvi, and M. de Rijke. Probabilistic permutation graph search: Black-box optimization for fairness in ranking. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 715–725, 2022. (Cited on pages 70 and 74.)

[126] A. Vardasbi, M. de Rijke, F. Diaz, and M. Dehghani. Group membership bias. *arXiv preprint arXiv:2308.02887*, 2023.

[127] A. Vardasbi, T. P. Pires, R. M. Schmidt, and S. Peitz. State spaces aren't enough: Machine translation needs attention. In *The 24th Annual Conference of the European Association for Machine Translation*, EAMT '23, 2023.

[128] S. Verma and J. Rubin. Fairness definitions explained. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, pages 1–7, 2018. (Cited on page 84.)

[129] M. Vlasceanu and D. M. Amodio. Propagation of societal gender inequality by internet search algorithms. *Proceedings of the National Academy of Sciences*, 119(29):e2204529119, 2022. (Cited on

pages 2 and 67.)

[130] X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 115–124. ACM, 2016. (Cited on pages 2, 11, 12, 20, 22, 23, 28, and 41.)

[131] 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. (Cited on pages 1, 2, 3, 11, 12, 13, 14, 15, 16, 20, 23, 32, 34, 39, 42, 44, 46, 47, 48, 58, 64, 66, and 72.)

[132] X. Wang, C. Li, N. Golbandi, M. Bendersky, and M. Najork. The LambdaLoss framework for ranking metric optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1313–1322. ACM, 2018. (Cited on pages 23 and 34.)

[133] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992. (Cited on pages 4, 84, and 87.)

[134] H. Wu, B. Mitra, C. Ma, F. Diaz, and X. Liu. Joint multisided exposure fairness for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, pages 703–714, 2022. (Cited on page 64.)

[135] H. Yadav, Z. Du, and T. Joachims. Policy-gradient training of fair and unbiased ranking functions. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1044–1053, 2021. (Cited on pages 74, 86, and 96.)

[136] L. Yan, Z. Qin, H. Zhuang, X. Wang, M. Bendersky, and M. Najork. Revisiting two-tower models for unbiased learning to rank. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2410–2414, 2022. (Cited on page 109.)

[137] K. Yang and J. Stoyanovich. Measuring fairness in ranked outputs. In *29th International Conference on Scientific and Statistical Database Management*, SSDBM, pages 1–6, 2017. (Cited on page 86.)

[138] Y. Yue, R. Patel, and H. Roehrig. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *International World Wide Web Conference*, pages 1011–1018. ACM, 2010. (Cited on pages 2 and 20.)

[139] B. Zdaniuk and J. M. Levine. Group loyalty: Impact of members' identification and contributions. *Journal of Experimental Social Psychology*, 37(6):502–509, 2001. (Cited on page 67.)

[140] M. Zehlike, F. Bonchi, C. Castillo, S. Hajian, M. Megahed, and R. Baeza-Yates. Fa*ir: A fair top-k ranking algorithm. In *Proceedings of the 26th ACM International Conference on Information & Knowledge Management*, pages 1569–1578, 2017. (Cited on page 86.)

[141] M. Zehlike, K. Yang, and J. Stoyanovich. Fairness in ranking: A survey. *arXiv preprint arXiv:2103.14000*, 2021. (Cited on page 86.)

[142] H. Zhao, J. Xu, X. Zhang, G. Cai, Z. Dong, and J.-R. Wen. Separating examination and trust bias from click predictions for unbiased relevance ranking. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, WSDM '23, page 913–921, 2023. (Cited on page 109.)

[143] M. Zoghi, T. Tunys, L. Li, D. Jose, J. Chen, C. M. Chin, and M. de Rijke. Click-based hot fixes for underperforming torso queries. In *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 2016. (Cited on pages 68, 85, and 88.)

[144] M. Zoghi, T. Tunys, M. Ghavamzadeh, B. Kveton, C. Szepesvari, and Z. Wen. Online learning to rank in stochastic click models. In *International conference on machine learning*, pages 4199–4208. PMLR, 2017. (Cited on pages 1 and 48.)

Summary

Online search engines are a means to satisfy users' information needs, but they can also form the knowledge of crowds. To keep up with the continuous changes in information as well as user taste, search engines learn from user interactions, and since user interactions come with bias, an important focus of research in this field lies in developing methods to correct for the bias of interactions. To ensure that sensitive groups and individuals are not hurt by the way of representing the results, a growing body of research focuses on making rankings fair.

In this thesis, we re-examine the assumptions of existing methods for bias correction and fairness optimization in ranking. Consequently, we propose methods that are more general than the existing ones, in the sense that they rely on less assumptions, or they are applicable in more situations. On the bias side, we first show that the click model assumption matters and propose cascade model-based inverse propensity scoring (IPS), on top of the existing position-based IPS. Next, we prove that the unbiasedness of IPS relies on the assumption that the clicks do not suffer from trust bias. When trust bias exists, i.e., users are more likely to click incorrectly on highly ranked items, we extend IPS and propose the affine correction (AC) method and prove that, in contrast to IPS, it gives unbiased estimates of the relevance. Finally, we show that the unbiasedness proofs of IPS and AC are conditioned on an accurate estimation of the bias parameters, and propose a bias correction method that does not rely on relevance estimation and, furthermore, is less sensitive to the users' true click behavior compared to IPS and AC. On the fairness side, we re-examine the implicit assumption that fair distribution of exposure leads to fair treatment by the users. Relying on the existing studies showing that the perceived group membership of an item affects users' judgments about its utility, we argue that fairness of exposure is necessary but not enough for a fair treatment and propose a correction method for this type of bias. Finally, we notice that the existing general post-processing framework for optimizing fairness of ranking metrics is based on the Plackett-Luce distribution, the optimization of which has room for improvement for queries with a small number of repeating sessions. To close this gap, we propose a new permutation distribution based on permutation graphs.

Samenvatting

Online zoekmachines zijn een middel om in de informatiebehoefte van gebruikers te voorzien, maar ze kunnen ook de kennis van de massa vormen. Om gelijke tred te houden met de voortdurende veranderingen in informatie en de smaak van gebruikers, leren zoekmachines van gebruikersinteracties, en aangezien gebruikersinteracties gepaard gaan met vooringenomenheid, ligt een belangrijke focus van onderzoek op dit gebied op het ontwikkelen van methoden om de vooringenomenheid van interacties te corrigeren. Om ervoor te zorgen dat gevoelige groepen en individuen geen schade ondervinden van de manier waarop de zoekresultaten worden weergegeven, richt een groeiend aantal onderzoeken zich op het eerlijk maken van ranglijsten.

In dit proefschrift onderzoeken we de aannames van bestaande methoden voor bias-correctie en eerlijkheidsoptimalisatie bij ranking. We stellen methoden voor die algemener zijn dan de bestaande, in de zin dat ze op minder aannames berusten, of dat ze in meer situaties toepasbaar zijn. Aan de bias-kant laten we eerst zien dat de aanname van het klikmodel ertoe doet en stellen we op cascademodellen gebaseerde inverse propensity scoring (IPS) voor, bovenop de bestaande positiegebaseerde IPS. Vervolgens bewijzen we dat de onbevooroordeeldheid van IPS berust op de veronderstelling dat de klikken geen last hebben van vertrouwensbias. Als er sprake is van vertrouwensbias, dat wil zeggen dat gebruikers eerder geneigd zijn om verkeerd te klikken op items met een hoge ranking, breiden we IPS uit en stellen we de affiene correctiemethode (AC) voor. We bewijzen dat deze, in tegenstelling tot IPS, onpartijdige schattingen van de relevantie geeft. Ten slotte laten we zien dat de onbevooroordeeldheidsbewijzen van IPS en AC afhankelijk zijn van een nauwkeurige schatting van de biasparameters, en stellen we een biascorrectiemethode voor die niet afhankelijk is van relevantieschatting en bovendien minder gevoelig is voor het echte klikgedrag van de gebruiker, vergeleken met IPS en AC. Wat eerlijkheid betreft, onderzoeken we opnieuw de impliciete veronderstelling dat een eerlijke verdeling van de blootstelling leidt tot een eerlijke behandeling door gebruikers. Op basis van bestaande onderzoeken die aantonen dat het waargenomen groepslidmaatschap van een item het oordeel van gebruikers over het nut ervan beïnvloedt, beargumenteren we dat eerlijkheid van de blootstelling noodzakelijk is, maar niet genoeg voor een eerlijke behandeling. We stellen een correctiemethode voor dit soort vooringenomenheid voor. Ten slotte merken we dat het bestaande algemene naverwerkingsraamwerk voor het optimaliseren van de eerlijkheid van rankingstatistieken gebaseerd is op de Plackett-Luce-distributie, waarvan de optimalisatie ruimte voor verbetering biedt voor zoekopdrachten met een klein aantal herhalende sessies. Om deze kloof te dichten, stellen we een nieuwe permutatieverdeling voor op basis van permutatiegrafen.

We re-examine the assumptions of existing methods for bias correction and fairness optimization in ranking. Consequently, we propose methods that are more general than the existing ones, in the sense that they rely on fewer assumptions, or they are applicable in more situations.

On the bias side, we first show that the click model assumption matters and propose cascade model-based inverse propensity scoring (IPS), on top of the existing position-based IPS. Next, we prove that the unbiasedness of IPS relies on the assumption that the clicks do not suffer from trust bias. When trust bias exists, i.e., users are more likely to click incorrectly on highly ranked items, we extend IPS and propose the affine correction (AC) method and prove that, in contrast to IPS, it gives unbiased estimates of the relevance. Finally, we show that the unbiasedness proofs of IPS and AC are conditioned on an accurate estimation of the bias parameters, and propose a bias correction method that does not rely on relevance estimation and, furthermore, is less sensitive to the users' true click behavior compared to IPS and AC.

On the fairness side, we re-examine the implicit assumption that fair distribution of exposure leads to fair treatment by the users. Relying on the existing studies showing that the perceived group membership of an item affects users' judgments about its utility, we argue that fairness of exposure is necessary but not enough for a fair treatment and propose a correction method for this type of bias. Finally, we notice that the existing general post-processing framework for optimizing fairness of ranking metrics is based on the Plackett-Luce distribution, the optimization of which has room for improvement for queries with a small number of repeating sessions. To close this gap, we propose a new permutation distribution based on permutation graphs.