Scoring is Not Enough: Addressing Gaps in Utility-Fairness Trade-offs for Ranking

Shubham Singh
University of Illinois Chicago
Chicago, IL, USA
ssing57@uic.edu

Ian A. Kash
University of Illinois Chicago
Chicago, IL, USA
iankash@uic.edu

Mesrob I. Ohannessian
University of Illinois Chicago
Chicago, IL, USA
mesrob@uic.edu

Abstract

Scoring functions are used to represent the relevance of individual documents. In modern recommendation systems, they are often learned from data and play a pivotal role in ranking sets of documents in a way that maximizes utility to a user's query. With the recent interest in algorithmic fairness, the success of scoring has naturally led to methods that learn scores that simultaneously trade off fairness and utility. In this work, we show that in stark contrast with utility-centric objectives, scoring is sub-optimal in achieving all utility-fairness trade-offs. We establish this with a series of counter-examples with a generic fairness formulation. We show that the issue persists whether we have a deterministic scoring function or a randomized one, or whether we measure fairness at the scope of a single query or across multiple queries. On the positive side, we experimentally demonstrate that semi-greedy post-processing has the potential to achieve much better trade-offs, often approaching the ideal of exhaustive post-processing in a computationally tractable way.

CCS Concepts

• **Information systems** → **Retrieval models and ranking**; • **Computing methodologies** → *Ranking*.

Keywords

Ranking, Scoring, Fairness, Trade-Offs, Plackett-Luce, Greedy

ACM Reference Format:

Shubham Singh, Ian A. Kash, and Mesrob I. Ohannessian. 2018. Scoring is Not Enough: Addressing Gaps in Utility-Fairness Trade-offs for Ranking. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Ranking is a basic task that has far-reaching contemporary applications. Recommendation systems have provided a key impetus to understanding ranking, motivated by presenting documents in response to a query, in such a way as to maximize engagement. Yet, ranking emerges in many diverse settings. For example, one may want to rank sites to determine the order in which they are inspected or to rank medical tests to decide the order in which they are administered. A primary utility often quantifies how well these ranking tasks are performed. With the diversity of applications, however, we increasingly recognize the importance of trading-off utility with secondary desirable objectives, such as fairness. To achieve such trade-offs, the field has continued to adhere to a key paradigm: learning how to score items based on the query and then performing ranking via sorting by these scores. We review key relevant work in this area in Section 2.

In this paper, we challenge this paradigm. More precisely, through a series of analytical, simulation, and real examples, we show that scoring cannot cover the entire range of achievable trade-offs and often falls considerably short of it. The key insight that drives this challenge is that, unlike utility, secondary objectives and in particular fairness cannot be decomposed as the contribution of individual items. On the positive side, we empirically demonstrate that semi-greedy approaches along with simple relevance-learning more effectively cover these trade-offs.

To make this thesis clear, we focus on the single-user case and on a specific family of binary item-group fairness metrics as a secondary objective, measured either per-query or across-queries. Within this context, we give a precise definition of scoring and we abstract away the particulars of the learning algorithms. Informally, we define scoring as assigning to each item, based on its features, a numerical score. This assignment may be deterministic or, in the case of some models such as Plackett-Luce [22, 27], randomized. We replace learning with two abstractions: the true relevances and the group membership of items are known and the ranking procedure can be determined based on exact knowledge of how items combine to form queries. These assumptions can be thought of as operating in the infinite-data regime. In Section 3, we make these definitions and assumptions mathematically precise. Then, in Section 4, we give counterexamples showing that scoring is not enough to achieve optimal trade-offs. Some of these counterexamples are tractable synthetic examples and some are based on numerical optimization.

In Section 5, we consider a simple alternative to scoring: learning relevances without regard to the secondary objective and then, ex-post, accounting for it. Since such post-processing needs to explore the space of rankings, it can be intractable, except in very simple settings. We show that having recourse to near-greedy approaches can be an effective approximation that much more effectively covers achievable trade-offs. Learning-to-rank via scoring is categorized as an in-processing approach to tackle secondary objectives such as fairness as part of the learning process. Often, in-processing approaches are touted to offer more flexibility than post-processing.

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

The main contribution of our work is thus to show that the nature of scoring limits this flexibility when trading off a decomposable utility and a non-decomposable objective such as fairness. Figure 1 gives an overview of the key ideas and results of the paper.

2 Related Work

2.1 Scoring Functions

Traditionally, one of the most common approaches to ranking documents has been based on giving a score to the documents and then using the score to sort them [1]. A scoring function, often learned, produces the score using the document features and a user query. Using the scores, the documents are ranked in an order such that the most relevant documents are shown first [7, 19]. The scoring functions are evaluated using a utility objective, which is defined over cumulative relevances of ranked documents for a user. Prior works have explored learning methods in three scopes: (*i*) pointwise — scores are learned for each document against labeled relevances [13]; (*ii*) pairwise — scores are based on relative relevances for document pairs [4, 8]; (*iii*) listwise — scores are generated based on the complete list of documents presented [7, 9]. Recent works have focused more on the listwise scoring functions since they allow direct optimization over the complete ranked documents and are easily compatible with the performance measures [9]. One of the common techniques to train listwise learning-to-rank (LTR) methods uses the Plackett-Luce (PL) model to output a ranking based on the probability of selecting the next document conditioned on observed documents. The PL model uses probabilistic sampling, instead of a non-differentiable sorting function, making the model differentiable and suitable for gradient-based training methods. However, in practice, calculating gradients for the PL model requires sampling all possible permutations given a document set, making the problem computationally intractable [29]. To solve this, researchers have proposed several techniques to efficiently estimate unbiased gradients of the PL model [18, 20, 23, 25]. In particular, we use the low-variance gradient estimation technique proposed by Gadetsky et al. to compute PL model gradients for the listwise ranking implementation when comparing our alternative to scoring methods [11]. These are inspired by reinforcement learning, and others have also used similar approaches for learning scoring [12, 31].

2.2 Fairness in Ranking

As ranking systems have been adopted by socio-technical systems, fairness has become a crucial requirement. Although scoring functions are more suited for utility-based objectives, significant research has been centered around developing fairness-focused scoring functions that are used for ranking [2, 24, 30, 31]. One difference we note in this recent work is the definition and scope of fairness. For example, some researchers have investigated fairness between individual documents [5, 6, 28], whereas others have studied the fairness between documents belonging to groups [30, 31, 34]. Another criterion to measure fairness is the difference in representation for each ranking query [14] versus the difference in expected exposure across multiple queries [24, 26]. An extensive review by Zehlike et al. elaborates on the recent advances in fair ranking methods, datasets used and their applications [35, 36]. We focus on item-group fairness, though some of this work also considers fairness toward users, rather than documents/items, and more recent work also attempts to jointly be fair toward both users and items [16]. Ex-post approaches have not been too widespread, and we mention two of the most relevant ones here. In [14], fairness is also achieved ex-post, exactly based on alternations. In particular, there is no notion of trade-off, and the focus is rather on learning scores in a way that is aware of the post-processing. One of the closest ex-post approaches to the present paper is the greedy approach of [26], which however does not bring light to the fact that scoring is otherwise suboptimal.

Our work studies the trade-offs between utility and fairness for scoring and non-scoring based ranking methods. It shows the inadequacy of the former in terms of achieving the best possible trade-off. We show that our arguments hold for fairness definitions across both single and multiple queries when considering group fairness.

3 Problem Setting

We introduce notions of "scorability" to formalize what scoring strives to achieve, without being bogged down by specific learning procedures. We use the following notation. Let the set of possible documents be denoted by $\mathcal{D}$ and individual documents be denoted by $d \in \mathcal{D}$. We associate with each document a relevance $\text{rel}(d) \in [0, 1]$. We assume that each query consists of a multiset D of m documents from $\mathcal{D}$, to allow repetition. Relevances, in general, depend on the query and the user placing the query. By forgoing this dependence, we are restricting to a specific case. However, when a limitation is present in a specific case it is also present in the general case in that instance. Thus the conclusions of the paper hold generally.

Deterministic and Random Ranking. A *deterministic ranking* σ_D is defined as a permutation of the m documents in D, in a way that may depend on D. $\sigma_D(i) = d$ is interpreted as placing document $d \in D$ at position i. We assume that no selection is made, that is all documents in the query are ranked. More precisely, for each D, σ_D is equivalent to a bijective map $[m] \to D$. We also allow for *randomized ranking*. In this case, σ_D is a random map defined through a distribution, conditional on D, over all $m!$ permutations. In other words, given D, σ_D is sampled from this conditional distribution. Although σ_D always depends on D, whenever D is explicitly specified we drop the subscript $_D$ and simply write σ.

Utility. Utility, in general, is a function from σ, D to $\mathbb{R}_+$. In most ranking applications, however, this utility decomposes as a sum. A document at position i delivers a utility to the querying user based on its relevance as well as a factor that depends only on position, captured by non-increasing weights w_i. The *utility* of a deterministic ranking can then be given as:

$$\mathcal{U}(\sigma, D) = \sum_i w_i \text{rel}(\sigma_D(i)). \tag{1}$$

Examples:

Recommendation systems If we use $w_i = 1/(\log_2 i + 1)$ to capture position bias, utility becomes the discounted

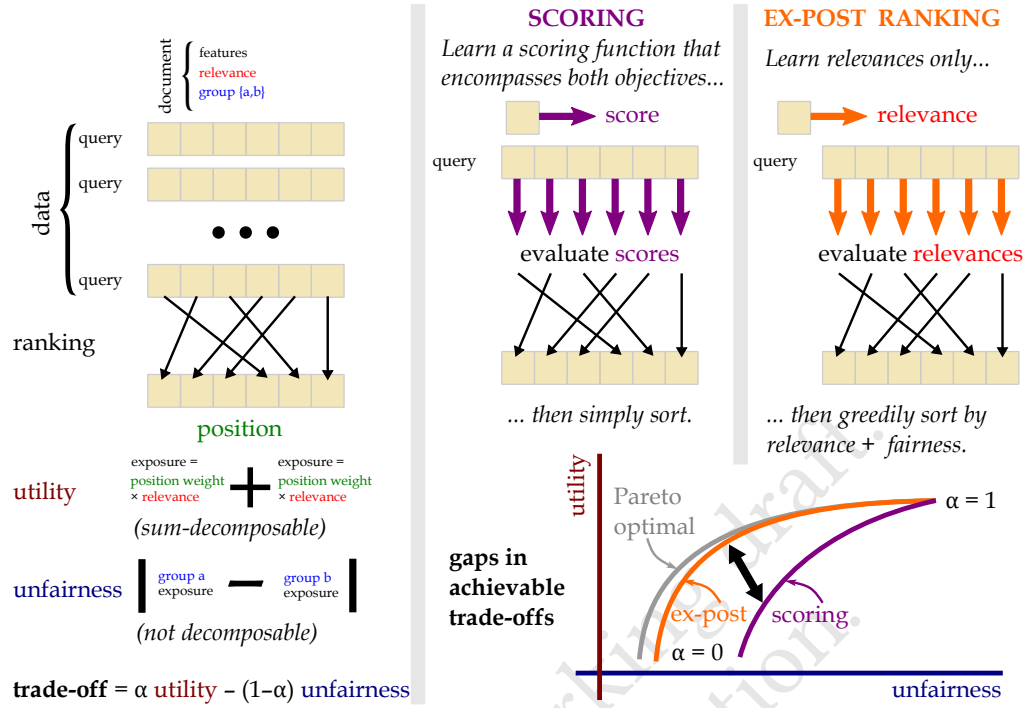

Figure 1: Overview of the paper, highlighting key ideas about the limitations of scoring functions for utility-fairness trade-offs.

cumulative gain (DCG). Here, $w_i\text{rel}$ represents an interaction probability, thus maximizing utility boosts interaction.

Inspection sites Documents could refer to inspections sites, with $\text{rel}(d)$ the probability of a failure occurring at site d, and σ_D the order in which sites are visited to correct failures. Then, choosing $w_i = -(i-1)/m$ and maximizing utility minimizes average uncorrected failure time.

For randomized ranking, because of its linearity, the notion of utility extends in a straightforward way to expected utility. We keep the notation the same, and distinguish the two instances based on context:

$$\mathcal{U}(\sigma, D) = \sum_i w_i \mathbf{E}\left[\text{rel}(\sigma_D(i)) | D\right]. \tag{2}$$

Note that the expectation is over random rankings, while the query D is fixed.

Unfairness. In addition to utility, we are concerned with giving documents a fair representation in the ranking. For the purpose of defining fairness, we think of $\mathcal{D}$ as being partitioned into two groups G_a and G_b, and use $g(d) \in \{a, b\}$ as the group-membership function. We capture fairness, or rather *unfairness*, by measuring the disparity of relative representation between the two groups:

$$\mathcal{V}(\sigma, D) = \left| \frac{1}{|G_a|} \sum_{i:g(\sigma_D(i))=a} w_i \text{rel}(\sigma_D(i)) - \frac{1}{|G_b|} \sum_{i:g(\sigma_D(i))=b} w_i \text{rel}(\sigma_D(i)) \right| \tag{3}$$

By convention, if either of the groups is not present in D, we assume fairness is vacuously achieved and set $\mathcal{V} = 0$.

For example, if the documents are interview candidates, utility captures consideration probability, and the groups refer to two racial groups, this disparity would be the difference of per-candidate consideration across these groups. In the earlier example of inspection sites, if G_a and G_b are two geographical inspection zones, this would be the disparity in failure time correction across zones. Other variants of (3) can be justified in various contexts, e.g., normalizing by the total relevance of each group instead of their sizes, using only w in the sum instead of rel, etc. We choose (3) because it is a common disparity measure and gives a concrete instance to illustrate the phenomenon that we aim to shed light on. This phenomenon, however, is likely not limited to this choice, as we discuss later.

For randomized ranking, some authors (e.g., [31]) aggregate the disparity over the random rankings within the absolute value in (3). This can be thought of as corresponding to a long session in which multiple rankings may be produced for a single query. However, since (3) measures per-query unfairness and since during a single query only a single ranking is sampled, we adhere to the more natural choice of aggregating outside of the absolute value, i.e., a short session:

$$\mathcal{V}(\sigma, D) = \mathbf{E}\left[\left| \frac{1}{|G_a|} \sum_{i:g(\sigma_D(i))=a} w_i \text{rel}(\sigma_D(i)) - \frac{1}{|G_b|} \sum_{i:g(\sigma_D(i))=b} w_i \text{rel}(\sigma_D(i)) \right| \middle| D \right], \tag{4}$$

Again, the expectation is over random rankings while the query D is fixed, and we keep the notation the same and disambiguate based on context. Readers interested in the effect of session length, albeit in a slightly different setting, can be referred to [32].

Trade-offs. With the primary objective of utility and the secondary objective of fairness, we are interested in striking a good trade-offs between utility (1) and unfairness (3). Trade-offs at level α can be defined through the combined objective function:

$$\mathcal{T}_\alpha(\sigma, D) = \alpha\, \mathcal{U}(\sigma, D) - (1-\alpha)\, \mathcal{V}(\sigma, D), \tag{5}$$

where the minus in the second term refers to the fact that we strive to increase utility, but to decrease unfairness.

Deterministic Scoring. Scoring, in a typical learning-to-rank framework, produces a function $\text{score}(d) : \mathcal{D} \to \mathbb{R}$ using either pairwise or list-wise data, as overviewed in Section 2. At query-time, $\text{score}(d)$ is computed for each document in the query D and the ranking is produced by sorting the documents in decreasing score order. The score-induced ranking can then be defined as:

$$\sigma_{D,\text{score}} = \text{sort}_\downarrow(D, \text{score}). \tag{6}$$

To avoid ties, we assume score never assigns distinct documents in $\mathcal{D}$ the same score (if it does, we arbitrarily break the tie by perturbing by an infinitesimal number). With limited data, score typically does not have direct access to rel, but rather depends on features of the document, potentially including its group membership. To narrow down on the issue at hand, consider instead the infinite-data regime where score knows exactly $\text{rel}(d)$ as well as $\text{g}(d)$. Since this information is sufficient to evaluate both (1) and (3), and thus (5), there is no loss of capability in restricting score to be an arbitrary function of these two attributes. Thus, moving forward, $\text{score}(d)$ is always of the form

$$\text{score}\left(\text{rel}(d), \text{g}(d)\right) : [0,1] \times \{a, b\} \to \mathbb{R}.$$

Randomized Scoring. Deterministic scoring is widely used, but a softer variant gives more flexibility. In this case, $\text{score}(d)$ is a random variable drawn independently from other documents in D, from a distribution f_d on $\mathbb{R}$ that depends uniquely on d. In the infinite-data regime, f_d depends uniquely on $\text{rel}(d)$ and $\text{g}(d)$. In this case, the score-induced ranking of (6) is a randomized ranking. The main case of randomized scoring that we consider is the Plackett-Luce (PL) model, a commonly used model for list-wise ranking [11, 14, 23, 25, 31, 33]. The best way to relate PL to randomized scoring is to note that it is equivalent to learning a deterministic score $h(d)$ first, and then to randomize it by adding independent standard Gumbel noise to each:

$$\text{score}_{\text{PL}}(d) = h(d) + \text{Gumbel}.$$

This then induces a randomized ranking:

$$\sigma_{D,\text{PL}} = \text{sort}_\downarrow\left(D, \text{score}_{\text{PL}}\right).$$

As proposed by Plackett [27] and Luce [22], an alternative way of describing PL is as a sequential generation of the ranking itself, through the probability of a document being ranked at the next position i, conditioned on the remaining documents up till that point.

$$\mathbb{P}\left(\sigma_{D,\text{PL}} = \sigma | D\right) = \prod_{i=1}^{m} \underbrace{\left(\frac{\exp(h(\sigma(i)))}{\sum_{j=i}^{m} \exp(h(\sigma(j)))}\right)}_{\mathbb{P}(\sigma(i)|D\setminus(\sigma(k))_{k<i})}. \tag{7}$$

In learning to rank, typically, h is parametrized using, for example, a linear regressor or neural network that depends on the features of d. The resulting ranking sampling is often referred to as a stochastic "ranking policy" π_D. In the infinite-data regime, we can take h to be simply a function of the relevance and group membership, $h(d) = h(\text{rel}(d), \text{g}(d))$.

4 Suboptimality of Scoring

The central question of this work is whether, to strike good trade-offs between utility (1) and unfairness (3), i.e., achieve a low value of trade-off at a desired level α (5), it is sufficient to rely on scoring functions.

4.1 Deterministic Case

We start by address deterministic scores. We may have varying levels of expectations of what a scoring function should accomplish. The most stringent of these is to expect the score to rank each D the best way possible. We can formalize this as follows.

Definition 1 (Strong Scorability). *A pair of utility and unfairness functions $(\mathcal{U}, \mathcal{V})$ is called* strongly α-scorable *if $\forall$ problem instances $(\mathcal{D}, \text{rel}, \text{g})$ $\exists$ deterministic* score *such that $\forall D = \{d_1, d_2, ..., d_m\}$, ranking them by score using $\sigma_{\text{score}} = \text{sort}_\downarrow(D, \text{score})$ achieves*

$$\mathcal{T}_\alpha\left(\sigma_{\text{score}}, D\right) = \max_\sigma \alpha \cdot \mathcal{U}(\sigma, D) - (1-\alpha)\mathcal{V}(\sigma, D) \tag{8}$$

By allowing score to depend on the problem instance, Definition 1 abstracts away the notion that the scoring function may be learned given sufficient data from this instance. What makes this requirement stringent or "strong" is that we ask for optimality of the trade-off for every query D. We now provide the first evidence that we should not take scoring for granted.

Theorem 1 (Counterexample to Strong Scorability). *Let $\mathcal{U}$ and $\mathcal{V}$ be given by Eqs.* (1) *and* (3) *with some monotonically non-increasing non-constant w. Then, for any $\alpha < 1$, there exists a problem instance $(\mathcal{D}, \text{rel}, \text{g})$ such that $\forall$ deterministic scoring functions* score *that can depend on the instance, there exists D such that Eq.* (8) *fails. Therefore, this choice of $(\mathcal{U}, \mathcal{V})$ is not strongly scorable.*

Demanding that a scoring function achieve the optimal trade-off for every query D may be seen as too strong, since the counterexample explicitly constructs a hard query D to counteract each choice of score. Does the issue persist if we relax this definition? Consider instead the following notion.

Definition 2 (Weak Scorablility). *A pair of utility and unfairness functions $(\mathcal{U}, \mathcal{V})$ is called* weakly α-scorable *if $\forall$ problem instances $(\mathcal{D}, \text{rel}, \text{g})$ and distributions $\mathbb{D}$ on $\mathcal{D}$ $\exists$ a deterministic* score *such that if the documents in D are sampled $\sim_{\text{i.i.d.}} \mathbb{D}$, then ranking them by score using $\sigma_{\text{score}} = \text{sort}_\downarrow(D, \text{score})$ achieves*

$$\begin{aligned}&\mathbf{E}_D\left[\mathcal{T}_\alpha\left(\sigma_{\text{score}}, D\right)\right] = \\ &\quad \max_{\sigma_D} \alpha \cdot \mathbf{E}_D\left[\mathcal{U}(\sigma, D)\right] - (1-\alpha)\mathbf{E}_D\left[\mathcal{V}(\sigma, D)\right].\end{aligned} \tag{9}$$

It is important to note that the maximization in Eq. (9) is over σ_D, the space of mappings from D to permutations, and not over a fixed permutation.

Theorem 2 (Counterexample to Weak Scorability). *Let $\mathcal{U}$ and $\mathcal{V}$ be given by Eqs.* (1) *and* (3) *with some monotonically non-increasing non-constant w. Then, there exists a problem instance* $(\mathcal{D}, \text{rel}, \text{g})$ *and a distribution $\mathbb{D}$ on $\mathcal{D}$, such that for all scoring functions* score, *Eq.* (9) *fails. Therefore, this choice of $(\mathcal{U}, \mathcal{V})$ is not weakly scorable.*

The reason the apparently weaker definition 2 does not change the fortunes of scorability is because even though we are asking for scoring to achieve the best average trade-off, this trade-off can still be optimized query-by-query. This is because to optimize $\mathbf{E}_D[f(\sigma)]$ we can optimize $\mathbf{E}[f(\sigma)\|D]$ for each D, yielding σ_D.

4.2 Randomized Case

Having identified that deterministic scorability fails for the family of utility and unfairness objectives considered, we now demonstrate that randomization does not necessarily help circumvent the trade-off gap. In particular, we focus on the widely used Plackett-Luce model, we construct a specific tractable counterexample instance where the Pareto frontier can be calculated, then numerically optimize the randomized scoring and demonstrate its suboptimal trade-offs.

We build a counterexample where the best Plackett-Luce model still suffers from a gap. We let weights to be the standard position bias $w_i = 1/(\log_2(i) + 1)$. We choose $\mathbb{D}$ to have $k = 5$ documents, with the following composition $(\text{rel}(d), \text{g}(d)) = (1, a), (2, b), (3, a), (4, b), (5, a)$. We sample, i.i.d., queries D consisting of $m = 8$ documents. This is the same as the synthetic data set described in Section 6, which also contains details about how the PL model is solved. Primarily, to calculate gradients directly is intractable and one often resorts to the log-trick, as in the REINFORCE algorithm, However, a variance reduction technique proposed by Grathwohl et al. [15] has been demonstrated to work well for training PL models [11], and is particularly reliable for this simple counterexample.

The results are included as part of Section 6, in Figure 2. The navy (top navy curve) is the exact (exhaustive) ex-post optimal solution. We can see clearly that the PL model (bottom purple curve) is unable to achieve good trade-offs in comparison. In Section 5, we propose and explain the alternatives approaches plotted in the same figure.

4.3 Across-Query Fairness

The notion of unfairness given by Eq. (3) is per-query. Namely, we measure the discrepancy between groups at the level of individual queries. While this is reasonable in many applications, e.g., suggesting male and female applicants in response to a single job posting, there are other instances where fairness is better measured across queries, e.g., whether certain movie categories are reaching a wide enough audience. We now demonstrate that even in that setting, scoring can be suboptimal.

We assume, as in the case of weak scorability, that queries are formed based on a distribution $D \sim \mathbb{D}$. The notion of utility is mostly unchanged, except that we now average over all queries:

$$\overline{\mathcal{U}}(\sigma, \mathbb{D}) = \mathbf{E}_{D\sim\mathbb{D}}[\mathcal{U}(\sigma, \mathcal{D})]. \tag{10}$$

The notion of unfairness, however, is more radically affected. Instead of aggregating representation per query, we aggregate it across queries:

$$\overline{\mathcal{V}}(\sigma, D) = \left| \mathbf{E}_{D\sim\mathbb{D}} \left[\frac{1}{m\mathbb{P}(G_a)} \sum_{i:\text{g}(\sigma_D(i))=a} w_i \text{rel}(\sigma_D(i)) - \frac{1}{m\mathbb{P}(G_b)} \sum_{i:\text{g}(\sigma_D(i))=b} w_i \text{rel}(\sigma_D(i)) \right] \right| \tag{11}$$

where

$$\mathbb{P}(G) = \frac{1}{m}\mathbf{E}_{D\sim\mathbb{D}}\left[\sum_i \mathbb{1}\{\sigma_D(i) \in G\}\right].$$

Definition 3 (Across-Query Scorability). *A pair of across-query utility and unfairness functions is called scorable if $\forall\alpha$, $\forall\mathbb{D}$, $\exists score_{\alpha,\mathbb{D}}$ such that if $\mathcal{D} = \{d_1, d_2, ..., d_m\} \sim$ i.i.d. from $\mathbb{D}$ are sorted using $\sigma = sort(\mathcal{D}, score_{\alpha,\mathbb{D}})$ achieves*

$$\max_{\sigma} \quad \alpha \cdot \overline{\mathcal{U}}(\sigma, \mathbb{D}) - (1-\alpha) \cdot \overline{\mathcal{V}}(\sigma, \mathbb{D})$$

Counterexample to Across-Query Scorability. To demonstrate that even in the across-query setting scorability is not to be taken granted, we construct a specific example where we can tractably solve for the optimal ranking.

In particular, we sample a query $D = \{\{d_1, \ldots, d_m\}\}$ from document distribution $\mathbb{D}$ i.i.d. (with replacement), where $\mathbb{D}$ is a categorical distribution of k documents. We assume $m \geq k$. The number of all possible multisets is given $(\binom{k}{m})$. Within each multiset, the number of possible permutations is given by $\binom{m}{m_1,\ldots,m_k}$. Recall that the maximization in Definition (3) determines, for every multiset D, the particular permutation representing the way they would be ranked. Unlike the per-query instance where the optimization could be solved for each query separately, the across-query optimization needs to take into account all queries simultaneously.

To solve this, we propose the following linear relaxation. A multiset is equivalently identified via the corresponding histogram $h = Hist(D)$. Let $J(h)$ be the set of all permutations within the histogram h, and let $j \in J(h)$ represent the indices of individual permutations. In other words, j's in all h's collectively define σ. Each choice j results in a contribution to utility, which we denote by $\mathcal{U}_{h,j}$, as well as a contribution to the unfairness term within the absolute values, which we denote by $\mathcal{V}_{h,j}$. These contributions can be computed offline. If we relax the selection of the permutation to a convex combination over permutations given by coefficient $x_{j|h}$, akin to randomized sampling, the maximization of the trade-off can be written as a linear program as follows:

	Optimal Scoring		Linear Program	
α	Utility	Unfairness	Utility	Unfairness
0	11.75	0	11.16	0.00
0.1	11.75	0	13.21	0.00
0.2	13.27	0.33	13.21	0.00
0.3	13.27	0.33	13.21	0.00
0.4	13.27	0.33	13.21	0.00
0.5	13.27	0.33	13.21	0.00
0.6	13.36	0.45	13.21	0.00
0.7	13.36	0.45	13.21	0.00
0.8	13.36	0.45	13.21	0.00
0.9	13.36	0.45	13.21	0.00
1	13.36	0.45	13.36	0.45

Table 1: Across-query utility and unfairness: suboptimality of scoring vs. direct optimization.

$$
\begin{aligned}
\text{maximize} \quad & \alpha \sum_h p(h) \sum_{j \in J(h)} x_{j|h}\, \mathcal{U}_{h,j} - Z \\
\text{subject to} \quad & (1-\alpha) \sum_h p(h) \sum_{j \in J(h)} x_{j|h}\, \mathcal{V}_{h,j} \le Z \\
& -(1-\alpha) \sum_h p(h) \sum_{j \in J(h)} x_{j|h}\, \mathcal{V}_{h,j} \le Z \\
& \sum_{j \in J(h)} x_{j|h} = 1, \quad \forall h \\
& x_{j|h} \ge 0, \quad \forall h, j \\
& Z \ge 0\,.
\end{aligned}
$$

We choose a uniform distribution for $\mathbb{D}$, which then results each sequence to be equally likely. The probability $p(h)$ for a histogram h is then computed using the multinomial coefficient, representing the number of ways to arrange elements according to the counts in h:

$$p(h) = \frac{m!}{h_1!\, h_2!\, \ldots\, h_k!} \left(\frac{1}{k}\right)^m .$$

As for the PL model, we choose $k = 5$, $m = 8$, with the following composition of documents $(\mathrm{rel}(d), \mathrm{g}(d)) = (1, a), (2, b), (3, a), (4, b), (5, a)$. Weights are chosen to be the standard position bias $w_i = 1/(\log_2(i) + 1)$. This is the same as the synthetic data set described in Section 6. To compare with scoring, we optimize over all possible deterministic scorings by considering all $k!$ different orderings that they induce on the documents. The results are in Table 1, highlighting a clear gap between scoring and the optimal trade-offs. It is worth noting that despite the linear program being a relaxation, it predominantly produces pure strategies, i.e., a single ranking choice per multiset.

5 Alternatives to Scoring

Having seen the shortcomings of scoring in various context, be it deterministic, randomize, per query D or across queries drawn from a distribution $\mathbb{D}$, we now turn our attention to potential alternatives to overcome these shortcomings. In particular, we explore and propose ex-post methods that can achieve better trade-off values than scoring functions. In what follows, we assume that we know the true relevance of the documents or that can accurately learn them.

In Section 6, we compare these approximate ex-post approaches to exact brute-force *post-processing*, that evaluates all the permutations for a sampled document set and finds the ranking that achieves the best fairness trade-offs for the sampled set.

5.1 Greedy Approach

We design a greedy approach that creates a ranking that achieves better trade-off values than scoring methods. The first step in the greedy approach is to split the document set by groups, $\{a, b\}$, and within each group, sort the documents in decreasing order of their relevance. The group-wise sorted document sets let the greedy approach decide on the most suitable document for a better utility-fairness trade-off by considering just the most relevant remaining document from each group. This is similar to the approach followed by [26].

The greedy approach starts with an empty list of ranked documents. For each position, i, we add the most relevant document from each group, $\{a, b\}$, and calculate the trade-off for a given α. The document that achieves the maximum trade-off value is ranked at position i and removed from the group-wise sorted document sets. The trade-off value at any position i is computed for the previously selected documents and the one considered at position i, given as:

$$\sigma_D(i) = \arg \max_{g \in \{a,b\}} f\left(\sigma([0, \ldots, i-1]) \cup \{d_g^*\}\right), \qquad (12)$$

where d_g^* is the document with the highest relevance from group g, and f represents the (truncated) trade-off function. We repeat this process for all positions, and the final result is a ranked list of all documents.

In theory, since we make the best decision at each position for a given trade-off value, the greedy approach should achieve the best possible trade-off for the complete ranking. However, in reality, the unfairness of a ranking depends on the relative positions of other documents, and the one-step decision in the greedy approach is too myopic and can get stuck in a local optimum. This is another reason why greedy performs better at higher α values, where utility is given more weight in the trade-off.

5.2 Beam Search

The second approach we propose is adapted from beam search [21], commonly used in NLP procedures [10]. Beam search is similar to the greedy approach we mentioned in 5.1 with a modification in the number of documents the algorithm considers from each group. Precisely, the algorithm takes B and L as inputs, which denote beam size and look-ahead respectively and are used to determine how many ranked list choices it keeps in memory with the highest trade-off values.

We initialize a list of size B with the most relevant document from each group. For subsequent positions $i > 1$, we collect the top-L most highly relevant documents from each group in a round-robin way, append them to the documents in each list to create temporary ranking orders, and calculate their trade-offs. Suppose the list of rankings in the beam is given as $\{\sigma_D^1, \ldots, \sigma_D^B\}$ such that

$1 \leq b \leq B$ and the documents to be evaluated are written as $\{d_g^\ell \mid g \in \{a, b\}, \ell = 1, \ldots, L\}$, then updating each ranking in B could be written as:

$$\sigma_D^b = \arg\max_{1 \leq \ell \leq L} f\left(\sigma_D^b([0, \ldots, i-1]) \cup \{d_g^\ell\}\right) \tag{13}$$

The number of rankings to be considered at each step is thus $B \times L$, and we sort them in decreasing order of their trade-off values and only keep the top-B lists to maintain our beam size. We repeat this process until all the remaining documents in the set have been ranked and the ranking with the best trade-off is returned. We describe the complete beam search for ranking in Algorithm 1.

Algorithm 1: Beam Search for Utility-Fairness Trade-off

Input: Relevances rel's, Group memberships g's, Sampled indices `indices`, Beam size B, Look-ahead size L, Iterations N
Output: Trade-offs for varying α
foreach $\alpha \in [0, 1]$ *with step size* 0.1 **do**
  $f_\alpha = \alpha \cdot \mathcal{U}(\sigma, D) - (1 - \alpha) \cdot \mathcal{V}(\sigma, D)$;
  foreach *iteration* $i = 1, 2, \ldots, N$ **do**
    Get relevance rel and group membership g for queried documents D;
    Sort D based on rel in descending order;
    Group sorted D into dictionary `desc_group_dict` based on g;
    Initialize the beam with top-ranked items from each group;
    Set `beam_candidates` to the B best initial choices based on their trade-offs;
    for *each position* $j = 1, 2, \ldots, |\text{rel}| - 1$ **do**
      Initialize empty sets for `trade-off_list` and `seek_indices`;
      for *each beam candidate* $w \in$ `beam_candidates` **do**
        Collect remaining unprocessed documents for each group from `desc_group_idx_dict`;
        Limit the number of documents from each group to the look-ahead size L;
        Combine remaining documents into a single list;
        for *each document* `index` $\in$ `remaining` **do**
          Compute trade-off for appending `index` to beam candidate w;
          Append the trade-off and the updated beam to `trade-off_list` and `seek_indices`;
      Select the top B beams with the lowest trade-off values;
      Update `beam_candidates` accordingly;
    Select the final beam `beam_idx` as the candidate with the lowest trade-off;
    Compute final utility and unfairness for the selected beam;

6 Evaluation

We evaluate both scoring via listwise Plackett-Luce ranking and our ex-post approaches (greedy and beam search) on one synthetic dataset and one real-world dataset. We show that, across all ranges of α, our approaches achieve considerably better trade-offs than scoring. In the case of synthetic data, the results are also compared to the optimal trade-off, which ex-post approaches but scoring does not.

6.1 Data

Our evaluation is done on two datasets — a synthetic dataset and the German Credit dataset [17]. To construct the synthetic dataset, we select $k = 5$ unique documents with relevances $\{1, ..., 5\}$ and alternating group membership $\{a, b, a, b, \cdots\}$. For each query D on the k documents, we sample $m = 8$ random rankings. We sample 50,000 such queries to constitute a synthetic training set. The small values of k and m are intended to make the exact exhaustive ex-post optimal solution tractable in this case.

We also adapt the German Credit dataset for a ranking task (as done in [31]) by following a similar process to the synthetic dataset. The German Credit dataset contains 1,000 rows of 20 attributes comprising an individual's creditworthiness and true relevances. Relevances are binary in this case, i.e., creditworthy (rel=1) or not (rel=0). Following the methodology of Singh and Joachims [31], each query ranks 10 documents with a ratio of 4:1 for creditworthy to non-creditworthy individuals. We perform an 80-20 train-test split and sample 10,000 queries from the train split and 2,000 from the test split. We use the protected attribute, sex, as the group membership to calculate unfairness.

6.2 Experiments

We design the experiments to measure how trade-offs between utility and unfairness are navigated by different methods at different α values.

Ex-post — The *ex-post* ranking iterates over all possible permutations given the documents in a query and finds the optimal given an α value. Naturally, this is an expensive operation and takes $O(n!)$ time. Therefore, we perform this evaluation only for the synthetic dataset with a smaller query size of $m = 8$. Using this baseline, we get the best possible ranking for each query, which allows us to contextualize the performance of the other approaches.

Greedy and Beam Search — These proposed alternative approaches are elaborated in Section 5 and in Algorithm 1. They are approximate ex-post solutions that require knowledge of relevances, either exact or learned.

PL model — We use PL-RELAX, the low-variance gradient estimator for the Plackett-Luce distribution, proposed by Gadetsky et al. [11], to train on our datasets. The proposed estimator is unbiased and offers low variance compared to the other gradient estimators, such as REINFORCE. The PL model is a listwise ranking algorithm that learns a real-number preference over the documents and samples a ranking that optimizes a loss function. In our case, the loss function is the trade-off between utility and unfairness Eq. (5).

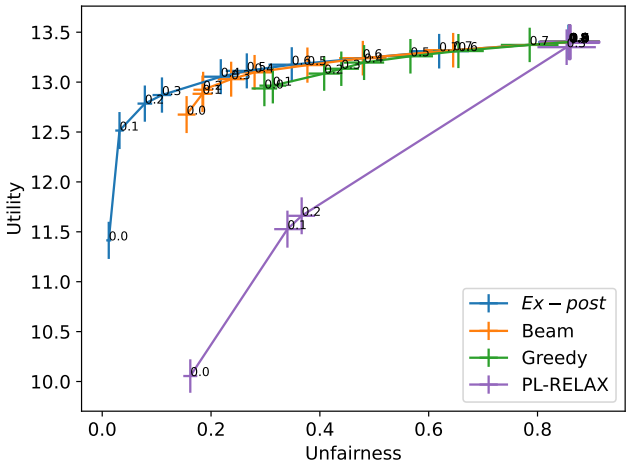

Figure 2: Utility and unfairness values for $\alpha = \{0, 0.1, \ldots, 1\}$ achieved by the brute-force *ex-post* (blue), beam search (orange), greedy (green) and the trained PL model (purple) on the synthetic dataset.

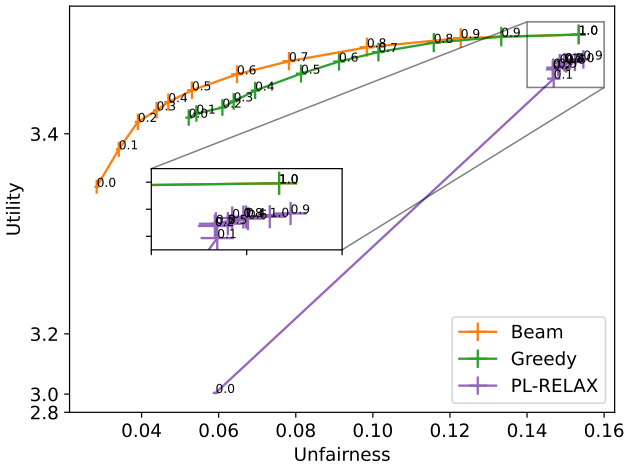

Figure 3: Utility (in log-scale) and unfairness values for $\alpha = \{0, 0.1, \ldots, 1\}$ achieved by the beam search (orange), greedy (green), and the trained PL model (purple) on the German Credit dataset.

6.3 Results

Synthetic Dataset. We plot the results for the *ex-post*, beam search, greedy, and the PL-model on the synthetic dataset in Fig. 2. Each line corresponds to the trade-offs offered by one of the approaches for $\alpha \in \{0, \ldots, 1\}$. The greedy approach performs better than the PL model with gradient estimates (shown in purple, labeled PL-RELAX) and for $\alpha > 0.5$ gets close to the Pareto frontier achieved by the *ex-post* approach. One reason that explains the better performance of the greedy approach at higher α values is as follows. When we prefer utility over fairness, since our choice of documents across groups is sorted in a decreasing order by relevance, the greedy approach gets close to finding optimal rankings without much exploration.

Compared to the greedy approach, the beam search performs closer to the *ex-post* trade-off values. Especially at lower α values, it can achieve lower unfairness while having a comparative utility to the greedy approach. Since the beam search evaluates trade-off values for more documents for ranking, it "explores" more permutations and, hence, can avoid making the suboptimal choice at the immediate next step by only considering the next most relevant documents, like in the greedy approach. Naturally, this exploration depends on the parameter L, and in our evaluation, we see the best results using $L = 1$. Beam search does not completely avoid choosing suboptimal documents at intermediate steps when unfairness dominates utility (lower α values), and finding the perfect fairness for ranking remains a combinatorial problem [3].

German Credit Dataset. Unlike the synthetic dataset, where we know the exact relevance of each document, we assume that greedy and beam search have no access to the true relevances. To learn the relevances from the dataset, we train a Logistic Regression model that maps features to relevances, rel $\in [0, 1]$. The probability of each individual being creditworthy (rel $= 1$) is treated as the target variable for this prediction. Beam search and greedy use these learned relevances to rank the documents in a query, making the scenario closer to a real-world instance where access to true relevances is not always given. For the PL-RELAX approach, we learn a linear model to map features to the Plackett-Luce coefficients. This gives PL the same representation power as the logistic regression used to learn relevances for greedy and beam search.

Fig. 3 shows the trade-off curves traced by the beam search, greedy, and PL-RELAX. Due to the larger size of the query in this case (m=10) compared to the synthetic dataset (m=8), we do not calculate the *ex-post* trade-offs.

Just like in the synthetic dataset, the beam search achieves the best trade-off at each α. Again, greedy achieves better trade-offs as α gets larger, but it maintains a gap with beam search until the highest α values. Despite all attempts to improve its convergence, PL-RELAX does not achieve as significant of a trade-off.

The experiments on both datasets show the gap between the trade-offs offered by a probabilistic scoring function from what is possible to achieve. Our proposed approximate ex-post approaches, on the other han d, achieve much better trade-offs for all α values, even in the absence of true relevances.

7 Conclusion

In this paper, we looked closely at the problem of learning to rank in the presence of a traditional utility objective as well as a secondary objective, namely fairness. We challenged the common paradigm of learning a score per query-document pair, by giving several counterexamples that show that this can fall short of achieving good trade-offs between the two objectives. The key insight is that, despite being an in-processing technique, scoring is limited by the fact that fairness is non-decomposable in contrast to utility. We showed that our approximate ex-post solutions, in the form of greedy and beam-search, are tractable alternatives with much better trade-off. They both, however, rely on learning relevances directly.

This opens the question of, if other properties are learned during in-processing instead of or in addition to relevances, whether they could promote even better trade-offs.

References

[1] Aman Agarwal, Xuanhui Wang, Cheng Li, Michael Bendersky, and Marc Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-Rank. In *The World Wide Web Conference on - WWW '19*. ACM Press, San Francisco, CA, USA, 4–14. doi:10.1145/3308558.3313697

[2] Abolfazl Asudeh, H. V. Jagadish, Julia Stoyanovich, and Gautam Das. 2019. Designing Fair Ranking Schemes. In *Proceedings of the 2019 International Conference on Management of Data*. ACM, Amsterdam Netherlands, 1259–1276. doi:10.1145/3299869.3300079

[3] Tolga Bektaş and Adam N. Letchford. 2020. Using ℓp-Norms for Fairness in Combinatorial Optimisation. *Computers & Operations Research* 120 (Aug. 2020), 104975. doi:10.1016/j.cor.2020.104975

[4] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H. Chi, and Cristos Goodrow. 2019. Fairness in Recommendation Ranking through Pairwise Comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2212–2220. doi:10.1145/3292500.3330745

[5] Asia J. Biega, Krishna P. Gummadi, and Gerhard Weikum. 2018. Equity of Attention: Amortizing Individual Fairness in Rankings. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 405–414. doi:10.1145/3209978.3210063

[6] Amanda Bower, Hamid Eftekhari, Mikhail Yurochkin, and Yuekai Sun. 2020. Individually Fair Rankings. In *International Conference on Learning Representations*.

[7] Christopher Burges, Robert Ragno, and Quoc Le. 2006. Learning to Rank with Nonsmooth Cost Functions. In *Advances in Neural Information Processing Systems*, Vol. 19. MIT Press.

[8] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*. Association for Computing Machinery, New York, NY, USA, 89–96. doi:10.1145/1102351.1102363

[9] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to Rank: From Pairwise Approach to Listwise Approach. In *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*. Association for Computing Machinery, New York, NY, USA, 129–136. doi:10.1145/1273496.1273513

[10] Markus Freitag and Yaser Al-Onaizan. 2017. Beam Search Strategies for Neural Machine Translation. In *Proceedings of the First Workshop on Neural Machine Translation*, Thang Luong, Alexandra Birch, Graham Neubig, and Andrew Finch (Eds.). Association for Computational Linguistics, Vancouver, 56–60. doi:10.18653/v1/W17-3207

[11] Artyom Gadetsky, Kirill Struminsky, Christopher Robinson, Novi Quadrianto, and Dmitry Vetrov. 2019. Low-Variance Black-box Gradient Estimates for the Plackett-Luce Distribution. doi:10.48550/arXiv.1911.10036 arXiv:1911.10036 [cs, stat]

[12] Yingqiang Ge, Xiaoting Zhao, Lucia Yu, Saurabh Paul, Diane Hu, Chu-Cheng Hsieh, and Yongfeng Zhang. 2022. Toward Pareto efficient fairness-utility trade-off in recommendation through reinforcement learning. In *Proceedings of the fifteenth ACM international conference on web search and data mining*. 316–324.

[13] Fredric C. Gey. 1994. Inferring Probability of Relevance Using the Method of Logistic Regression. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '94)*. Springer-Verlag, Berlin, Heidelberg, 222–231.

[14] Sruthi Gorantla, Eshaan Bhansali, Amit Deshpande, and Anand Louis. 2023. Optimizing Group-Fair Plackett-Luce Ranking Models for Relevance and Ex-Post Fairness. doi:10.48550/arXiv.2308.13242 arXiv:2308.13242 [cs]

[15] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. 2018. Backpropagation through the Void: Optimizing Control Variates for Black-Box Gradient Estimation. In *International Conference on Learning Representations*.

[16] Sophie Greenwood, Sudalakshmee Chiniah, and Nikhil Garg. 2024. User-item fairness tradeoffs in recommendations. *arXiv preprint arXiv:2412.04466* (2024).

[17] Hans Hofmann. 1994. Statlog (German Credit Data). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5NC77.

[18] Iris A. M. Huijben, Wouter Kool, Max B. Paulus, and Ruud J. G. van Sloun. 2022. A Review of the Gumbel-max Trick and Its Extensions for Discrete Stochasticity in Machine Learning. doi:10.48550/arXiv.2110.01515 arXiv:2110.01515 [cs, stat]

[19] Thorsten Joachims. 2002. Optimizing Search Engines Using Clickthrough Data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02)*. Association for Computing Machinery, New York, NY, USA, 133–142. doi:10.1145/775047.775067

[20] Wouter Kool, Herke van Hoof, and Max Welling. 2020. Estimating Gradients for Discrete Random Variables by Sampling without Replacement. *International Conference on Learning Representations, ICLR 2020* (2020).

[21] Bruce T Lowerre. 1976. THE HARPY SPEECH RECOGNITION SYSTEM. *Carnegie Mellon University* (1976).

[22] R. Duncan Luce. 1959. *Individual Choice Behavior*. John Wiley, Oxford, England. xii, 153 pages.

[23] Jiaqi Ma, Xinyang Yi, Weijing Tang, Zhe Zhao, Lichan Hong, Ed Chi, and Qiaozhu Mei. 2021. Learning-to-Rank with Partitioned Preference: Fast Estimation for the Plackett-Luce Model. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. PMLR, 928–936.

[24] Omid Memarrast, Ashkan Rezaei, Rizal Fathony, and Brian Ziebart. 2021. Fairness for Robust Learning to Rank. doi:10.48550/arXiv.2112.06288 arXiv:2112.06288 [cs, stat]

[25] Harrie Oosterhuis. 2021. Computationally Efficient Optimization of Plackett-Luce Ranking Models for Relevance and Fairness. doi:10.48550/arXiv.2105.00855 arXiv:2105.00855 [cs]

[26] Zohreh Ovaisi, Parsa Saadatpanah, Shahin Sefati, Mesrob Ohannessian, and Elena Zheleva. 2024. Fairness of Interaction in Ranking under Position, Selection, and Trust Bias. *ACM Transactions on Recommender Systems* (April 2024). doi:10.1145/3652864

[27] R. L. Plackett. 1975. The Analysis of Permutations. *Applied Statistics* 24, 2 (1975), 193. doi:10.2307/2346567 jstor:2346567

[28] Yuta Saito and Thorsten Joachims. 2022. 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)*. Association for Computing Machinery, New York, NY, USA, 1514–1524. doi:10.1145/3534678.3539353

[29] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. 2015. Gradient Estimation Using Stochastic Computation Graphs. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 3528–3536.

[30] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, London United Kingdom, 2219–2228. doi:10.1145/3219819.3220088

[31] Ashudeep Singh and Thorsten Joachims. 2019. Policy Learning for Fairness in Ranking. *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (2019), 11.

[32] Ali Vardasbi, Fatemeh Sarvi, and Maarten de Rijke. 2022. 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*. 715–725.

[33] Tian Xia, Shaodan Zhai, and Shaojun Wang. 2019. Plackett-Luce Model for Learning-to-Rank Task. doi:10.48550/arXiv.1909.06722 arXiv:1909.06722 [cs]

[34] Ke Yang and Julia Stoyanovich. 2017. Measuring Fairness in Ranked Outputs. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management (SSDBM '17)*. Association for Computing Machinery, New York, NY, USA, 1–6. doi:10.1145/3085504.3085526

[35] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in Ranking, Part I: Score-Based Ranking. *Comput. Surveys* 55, 6 (Dec. 2022), 118:1–118:36. doi:10.1145/3533379

[36] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in Ranking, Part II: Learning-to-Rank and Recommender Systems. *Comput. Surveys* 55, 6 (Dec. 2022), 117:1–117:41. doi:10.1145/3533380

A Proofs

A.1 Proof of Theorem 1

Proof. Consider the function

$$\phi(j) = \frac{1}{j}\sum_{i=1}^{j} w_j - \frac{1}{m-j}\sum_{i=j+1}^{m} w_j,$$

with $j = 1, \cdots, m-1$. We claim that ϕ cannot be constant, unless w itself is constant. Indeed, if $\phi(1) = \phi(m-1)$, it implies that $w_1 = w_m$, and since w is monotonic, it in turn implies that w is constant, leading to a contradiction. This particularly means that we can always choose some $p \in \{1, \cdots, m-1\}$ such that $|\phi(p)| > |\phi(m-p)|$.

Construct the problem instance as follows. Let $\mathcal{D}$ consist of two types of documents, with $(\text{rel}(d), \text{g}(d))$ = either $(1, a)$ or $(1, b)$. Note that, by choosing both types to have the same relevance, the utility of all rankings will be the same, and the trade-off will only depend on the unfairness term.

As previously discussed, we assume that $\text{score}(d)$ is a function of $(\text{rel}(d), \text{g}(d))$ and that it cannot assign the same value to the two distinct types of documents. W.l.o.g., assume $\text{score}(1, a) > \text{score}(1, b)$. Construct D to have p documents of type $(1, a)$ and $m-p$ of type $(1, b)$. This means that the score-induced ranking assigns the top p spots to documents from G_a and the remaining $m-p$ to G_b. Note that the unfairness of this ranking is equal to $|\phi(p)|$. This, by construction, is larger than $|\phi(m-p)|$, which is the unfairness we would have obtained had we placed the G_b documents first and G_a documents last. Therefore, for this choice of D, Eq. (8) fails and it follows that strong scorability does not hold. □

A.2 Proof of Theorem 2

Proof of Theorem 2. Construct the problem instance similarly to the proof of Theorem 1. Choose $\mathcal{D}$ to have two types of documents $(\text{rel}(d), \text{g}(d)) = (1, a)$ and $(\text{rel}(d), \text{g}(d)) = (1, b)$. Let $\mathbb{D}$ sample from each group equally. Let p be as before, and assume, w.l.o.g., that $\text{score}(1, a) > \text{score}(1, b)$. Since, through iterated expectations and the fact that we have full control of σ_D for each D, we have:

$$\max_{\sigma_D} \alpha \cdot \mathbf{E}_D\left[\mathcal{U}(\sigma, D)\right] - (1-\alpha)\mathbf{E}_D\left[\mathcal{V}(\sigma, D)\right] =$$
$$\sum_i w_i \mathbf{E}_D\left[\max_\sigma -(1-\alpha)\mathbf{E}\left[\mathcal{V}(\sigma, D)\right]\right] =$$
$$\sum_i w_i \mathbf{E}_D\left[\max_\sigma -(1-\alpha)\mathcal{V}(\sigma, D)\right]$$

Now note that with positive probability, D will have exactly p documents of G_a and $m-p$ documents of G_b. We know, from proof of Theorem 1, that in that case $\mathcal{V}(\sigma, D)$ is not minimal with the scoring-induced ranking. This implies that the overall expectation is not maximal. Therefore, Eq. (9) fails, and it follows that weak scorability does not hold. □

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009