Understanding and Mitigating the Effect of Outliers in Fair Ranking

Fatemeh Sarvi
AIRLab, University of Amsterdam
Amsterdam, The Netherlands
f.sarvi@uva.nl

Maria Heuss
University of Amsterdam, Amsterdam
The Netherlands
m.c.heuss@uva.nl

Mohammad Aliannejadi
University of Amsterdam, Amsterdam
The Netherlands
m.aliannejadi@uva.nl

Sebastian Schelter
University of Amsterdam, Amsterdam
The Netherlands
s.schelter@uva.nl

Maarten de Rijke
University of Amsterdam, Amsterdam
The Netherlands
m.derijke@uva.nl

ABSTRACT

Traditional ranking systems are expected to sort items in the order of their relevance and thereby maximize their utility. In fair ranking, utility is complemented with fairness as an optimization goal. Recent work on fair ranking focuses on developing algorithms to optimize for fairness, given position-based exposure. In contrast, we identify the potential of outliers in a ranking to influence exposure and thereby negatively impact fairness. An outlier in a list of items can alter the examination probabilities, which can lead to different distributions of attention, compared to position-based exposure. We formalize outlierness in a ranking, show that outliers are present in realistic datasets, and present the results of an eye-tracking study, showing that users' scanning order and the exposure of items are influenced by the presence of outliers. We then introduce OMIT, a method for fair ranking in the presence of outliers. Given an outlier detection method, OMIT improves fair allocation of exposure by suppressing outliers in the top-k ranking. Using an academic search dataset, we show that outlierness optimization leads to a fairer policy that displays fewer outliers in the top-k, while maintaining a reasonable trade-off between fairness and utility.

CCS CONCEPTS

• **Information systems** → *Learning to rank.*

KEYWORDS

Fair ranking; Outliers

ACM Reference Format:
Fatemeh Sarvi, Maria Heuss, Mohammad Aliannejadi, Sebastian Schelter, and Maarten de Rijke. 2022. 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), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3488560.3498441

1 INTRODUCTION

The primary goal of a ranker as used in a search engine or recommender system is to optimize the list in order to satisfy user needs by sorting items in their order of relevance to the query [22]. Recently there has been a growing concern about the unfairness towards minority groups caused by this simplistic assumption [5, 6]. Several studies have proposed approaches to achieve fair ranking policies. The goal is to ensure that the protected groups receive a predefined share of visibility. Exposure-based methods [6, 24, 26, 31–33] quantify the expected amount of attention each individual or group of items receives from users, where attention is typically related to the item's position and based on the observation that users are more likely to click on items presented at higher positions [5, 19].

But item position is not the only factor that affects exposure [13]. Inter-item dependencies also play a key role [8]. E.g., consider a user who is trying to buy a phone. When searching on an e-commerce platform, if an item in the list is on promotion and has a "Best Seller" badge, this can be distracting so that it gets more attention from the user, irrespective of its position in the list; the item would stand out even more if it is the only one with this feature.

We hypothesize that inter-item dependencies have an effect on examination probability and exposure of items. We focus on the case of having an *outlier* in the ranking and aim to understand and address its effect on user behavior. We hypothesize that exposure received by an item is influenced by the existence of an outlier in the list, and assume that this effect should be considered while allocating exposure to protected groups in a fair ranking approach.

We define *outliers* as items that observably deviate from the rest. The properties and method with which we identify outliers in a set of items are dependent on the task. The properties are observable item features that can be presentational in nature or correspond to ranking features used to produce the ranked list. E.g., in the e-commerce search example, if only one item in a result page has a "Best Seller" tag, it is an outlier based on this presentational feature.

To begin, we perform an **exploratory analysis on the TREC Fair Ranking dataset**. We observe that a large number of outliers exist in the rankings, where we use multiple outlier detection techniques to identify outliers based on the papers' citations as they can make an item more attractive and catchy than others.

Next, we perform an **eye tracking study**, where we measure the attention that each item in a ranked list gets through eye tracking, so as to show that users can actually perceive outliers in rankings.

We find that attention is more focused on outlier items. The scanning order and exposure received by each item may be influenced by the existence of outliers. Unlike other types of bias studied in search and recommendation [19, 27, 28, 37, 40], our eye-tracking study reveals that outlierness comes from inter-item dependencies. It affects the examination propensities for items around the outlier in a way that is less dependent on the position and based on relationships between items presented together. However, it is translated to bolded keyword matches in the title and abstracts, which can be calculated for each item separately, independent of its neighbors. Attractiveness does not alter the examination model based on position bias, and only results in relatively more clicks on items when they are presented with more bolded matched keywords [40].

While allocating fair exposure to protected items or groups in a fair ranking solution, we should account for the effect of outliers. We **propose an approach to account for the existence of outliers in rankings without damaging the utility or fairness of the ranking** by mitigating outlierness, called OMIT. OMIT jointly optimizes (1) user utility, (2) item fairness, and (3) fewer outliers in top-k positions as a convex optimization problem that can be solved optimally through linear programming. Via its solution, we derive a stochastic ranking policy using Birkhoff-von Neumann (BvN) decomposition [7]. OMIT reduces the number of outliers at top-10 positions on the TREC 2020 dataset by 80.66%, while maintaining the NDCG@10, compared to a state-of-the-art ranking baseline.

The main contributions of this work are as follows: (1) we introduce, study and formalize the problem of outlierness in ranking and its effects on exposure distribution and fairness; (2) we run an extensive eye-tracking user study in two search domains to support our hypothesis about the existence of an effect of outliers on items' exposure; (3) inspired by our analysis, we propose OMIT, an efficient approach that mitigates the outlierness effect on fairness; (4) we compare OMIT to competitive baselines on two TREC datasets in terms of fairness, outlierness, and utility; OMIT is able to remove outliers while balancing utility and fairness; and (5) we make the data from our eye-tracking study plus the code that implements our baselines and OMIT publicly available.

2 BACKGROUND

Exposure and utility. Consider a single query q, that we will often leave out for notational simplicity, for which we want to rank a set of documents $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$. Suppose we are given document utilities $\mathbf{u} \in \mathbb{R}^N$, where u_i is a proxy for the relevance of document d_i for q. Let $\mathbf{v} \in \mathbb{R}^N$, be the *attention vector*, where v_j denotes how much attention a document gets at position j, and which is decreasing with the position. This vector encodes the assumed position bias, e.g., $v_j = 1/\log(1 + j)$.

We require a probabilistic ranking in the form of a doubly stochastic document-position matrix $\mathbf{P} \in [0, 1]^{N\times N}$ where P_{ij} denotes the probability of putting document d_i at position j. Such a matrix can be decomposed into a convex combination of permutation matrices, which allows us to sample a concrete ranking [32].

The *exposure* of a document d_i under ranking $\mathbf{P}$ denotes the expected attention that this document will get. Using the position based attention vector v, this can be modeled as a function of the ranking and position bias: $\text{Exposure}(d_i|\mathbf{P}) = \sum_{j=1}^{N} P_{ij}\, v_j$.

The *expected utility* U of a ranking $\mathbf{P}$ is the sum of the documents' utilities weighted by the exposure given to them by $\mathbf{P}$:

$$U(\mathbf{P}) = \sum_{i=1}^{N} u_i \ \text{Exposure}(d_i|\mathbf{P}) = \sum_{i=1}^{N}\sum_{j=1}^{N} u_i\, P_{ij}\, v_j = \mathbf{u}^T\mathbf{P}\mathbf{v}. \qquad (1)$$

Without fairness considerations, a utility-maximizing ranking can be found by sorting the documents in descending order of utility.

Group fairness. Suppose now that the documents $\mathcal{D}$ can be partitioned into two disjoint sets $\mathcal{D}_{dis}$ and $\mathcal{D}_{priv}$, where documents in $\mathcal{D}_{dis}$ belong to a historically disadvantaged group (e.g., publications from not so well established institutes), and those in $\mathcal{D}_{priv}$ belong to the privileged group (e.g., publications from well-established institutes). We want to ensure a certain notion of fairness in the ranking. We want to avoid *disparate treatment* of the different groups. We use the *disparate treatment ratio* [32], which measures how unequal the exposure given to the disadvantaged group (in relation to the corresponding utility of the disadvantaged group) is compared to the corresponding ratio of the privileged group, as:

$$\text{dTR}(\mathcal{D}_{dis}, \mathcal{D}_{priv}|\mathbf{P}) = \frac{\sum_{d_i\in\mathcal{D}_{dis}} \text{Exposure}(d_i|\mathbf{P}) / \sum_{d_i\in\mathcal{D}_{dis}} u_i}{\sum_{d_p\in\mathcal{D}_{priv}} \text{Exposure}(d_p|\mathbf{P}) / \sum_{d_p\in\mathcal{D}_{priv}} u_p}. \qquad (2)$$

Note that dTR is 1 if the groups are treated *fairly* and smaller than 1 if the ranking is unfair towards the disadvantaged group. We often encounter disparate treatment when only optimizing for the expected utility of a ranking [5, 6, 32]. We can find a utility maximizing ranking $\mathbf{P}$ that avoids disparate treatment by solving the following optimization problem [32]:

$$\begin{aligned} \mathbf{P} &= \arg\max\nolimits_{\mathbf{P}} \mathbf{u}^\top\mathbf{P}\mathbf{v} && (\textit{expected utility})\\ \text{such that } \mathbb{1}^\top\mathbf{P} &= \mathbb{1}^\top && (\textit{row stochasticity})\\ \mathbf{P}\mathbb{1} &= \mathbb{1} && (\textit{column stochasticity})\\ 0 \le \mathbf{P}_{ij} &\le 1 && (\textit{valid probabilities})\\ \mathbf{f}^\top\mathbf{P}\mathbf{v} &= 0, && (\textit{dTR constraint}) \end{aligned} \qquad (3)$$

where $\mathbb{1}$ denotes a vector and $\mathbf{f}$ is the vector constructed to encode the avoidance of disparate treatment, with

$$f_i = \frac{\mathbb{1}_{d_i\in\mathcal{D}_{dis}}}{|\mathcal{D}_{dis}|\ \sum_{d_s\in\mathcal{D}_{dis}} u_s} - \frac{\mathbb{1}_{d_i\in\mathcal{D}_{priv}}}{|\mathcal{D}_{priv}|\ \sum_{d_p\in\mathcal{D}_{priv}} u_p}, \qquad (4)$$

where $\mathbb{1}_{d_i\in\mathcal{D}_{dis}} = 1$ if document d_i is in the disadvantaged group and 0 otherwise (and analogously for $\mathbb{1}_{d_i\in\mathcal{D}_{priv}}$) [32].

Degrees of outlierness. Outliers are items that deviate from the rest of the data [18]. They can be interesting observations or suspicious anomalies. Either way, they are considered noise that can affect the statistical analysis. We describe three outlier detection methods that we will use later in the paper. Let $x = \{x_1, \ldots, x_n \mid x_i \in \mathbb{R}\}$ be a set of values for which we want to identify outliers.

Median Absolute Deviation (MAD). Although it is common practice to use Z-scores to identify possible outliers, this can be misleading (particularly for small sample sizes) due to the fact that the maximum Z-score is at most $(n-1)/\sqrt{n}$. Iglewicz and Hoaglin [17] recommend using the modified Z-score: $M_i = 0.6745(x_i - \tilde{x})/MAD$, where MAD is the median absolute deviation and $\tilde{x}$ is the median of x. These authors recommend that modified Z-scores with an absolute value of greater than 3.5 be labeled as potential outliers.

Median K-Nearest Neighbor (MedKNN). This model [3] uses the K-Nearest Neighbor algorithm to define a distance-based outlier detection method. For each point x_i we have a value $w_k(x_i)$ as the weight calculated from the k nearest neighbors; outliers are the points with the largest values of w_k. We use Median K-Nearest Neighbor, which computes $w_k(x_i)$ as the *median* distance of x_i to its k neighbors. To find the k nearest neighbors, the method linearizes the search space and uses *Hilbert space-filling curve* to search efficiently; the method scales linearly in the dimensionality and the size of the data.

Copula-Based Outlier Detection (COPOD). COPOD [21] is a novel outlier detection method based on estimating tail probabilities using empirical copula. COPOD uses empirical cumulative distribution functions (ECDFs) to compute tail probabilities. These tail probabilities estimate the probability of observing a point at least as extreme as x_i for each data point. If x_i is an outlier, the probability of observing a point as extreme as x_i should be small. It means that this point has a rare occurrence. This method is deterministic and efficient, and scalable to high dimensionality data.

3 OUTLIERS IN RANKING

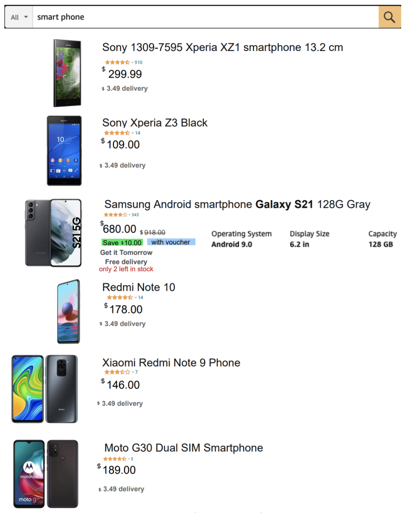
A common assumption is that exposure is a function of position [6, 24, 26, 32, 36]. We argue that this assumption holds only if ranked items can be deemed similar, meaning that no item is perceived as an outlier. Below we introduce outliers in the context of ranking. We then determine that outliers are present in rankings in realistic datasets. We also report on an eye-tracking study that shows that the presence of outliers in ranked lists impacts user behavior.

A definition of outliers in ranking. For a ranked list, we define outliers as items that stand out among the window of items that the user can see at once, drawing the user's attention. Outlier items often have (visible) characteristics that distinguish them from their neighbors. E.g., consider Fig. 1, which shows a result page for the query "smart phone". The result page view consists of 6 products, each presented with characteristics such as title, image, and price. The item at position three deviates from the other items in terms of several visual characteristics; it has more details, some promotive tags, and bold keywords. Other features, such as more positive reviews, or a higher price, may also influence the user's perceived relevancy. In this example, the third item can be considered as an outlier according to such visual characteristics.

Formally, we define outliers in ranking as follows. Consider a ranked list of N items in $\mathcal{D} = \{d_1, d_2, \dots, d_N\}$, that has been produced in response to a query. We call an observable characteristic of an item d in a ranked list an *observable item feature*. These features can be purely presentational in nature, like the bold keywords in Fig. 1, or correspond to ranking features used by the search engine to produce the ranked list, e.g., the average user rating.

Definition 3.1 (Degree of outlierness). Let g be an observable item feature, and M be one of the outlier detection methods discussed in Section 2. The *degree of outlierness* of an item d_i in the ranked list $[d_1, \dots, d_N]$ is the value calculated by M for $g(d_i)$ in the context of $\{g(d_1), \dots g(d_N)\}$, that determines how much $g(d_i)$ differs from the other elements of the set. We write $\mathrm{M}(g(d_i)|\mathcal{D})))$ for this value.

Definition 3.2 (Outliers in ranking). We say that according to M, item d_i is an *outlier in the ranked list* $[d_1, \dots, d_N]$ *for feature* g, if


All smart phone

Sony 1309-7595 Xperia XZ1 smartphone 13.2 cm
310
$ 299.99
$ 3.49 delivery

Sony Xperia Z3 Black
14
$ 109.00
$ 3.49 delivery

Samsung Android smartphone **Galaxy S21** 128G Gray
343
$ 680.00 $ ~~918.00~~
Save $10.00 with voucher
Get it Tomorrow
Free delivery
only 2 left in stock
Operating System Android 9.0
Display Size 6.2 in
Capacity 128 GB

Redmi Note 10
14
$ 178.00
$ 3.49 delivery

Xiaomi Redmi Note 9 Phone
7
$ 146.00
$ 3.49 delivery

Moto G30 Dual SIM Smartphone
3
$ 189.00
$ 3.49 delivery

Figure 1: E-commerce example used in our eye-tracking user study. A result page with one outlier at position 3, identified by more descriptive fields, higher price, and colored tags.

Table 1: Descriptive statistics of TREC Fair Ranking Track 2020 data.

	Train	Test
#queries	200	200
#unique papers	4,649	4,693
% of clicks in product pages	0.169	0.170

M identifies $g(d_i)$ as an outlier in the set $\{g(d_1), \dots, g(d_N)\}$.

Note that detecting an item as an outlier in a ranking depends on the context in which we see the item. Throughout the paper, we consider the full ranked list of items as the context in which we detect outliers. In Section 6 we study varying sizes for the context.

Moreover, it is possible to use multiple observable features to detect the outliers. For example, we can consider image size as g_1 and price as g_2, and then use any combination of these two feature values to present item d_i.

Below, when we refer to an item d being an outlier in a given ranked list, we assume that it is clear from the context what outlier detection method M and observable item feature g are being used.

Do outliers in ranking exist? To determine whether outliers are present in rankings in datasets, we take a retrieval test collection, compute feature values for one of the (potentially observable) rankings features appropriate for the collection, and determine whether there are outliers among the top-20 documents returned for the test queries (using ListNet as the ranker, see Section 5). For the experiments in Section 6 we use the academic search dataset provided by the TREC Fair Ranking track.[1] It contains information about papers and authors extracted from the Semantic Scholar Open Corpus.[2] It comes with queries and relevance judgments; see Table 1 for some descriptive statistics.

We used the number of citations as observable feature g for this dataset as they can make an item more attractive than others (when reported). In the remainder of the section, we report the analysis only on the TREC 2020 data, as we observed similar trends in both

[1]https://fair-trec.github.io/
[2]http://api.semanticscholar.org/corpus/

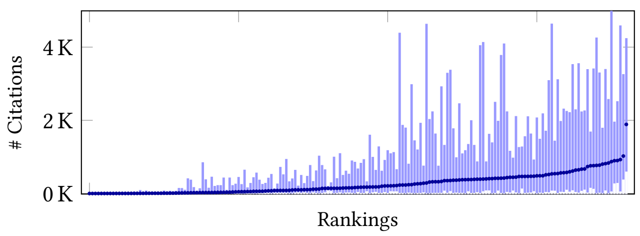


Figure 2: Distribution of the number of citations of top-20 papers returned for test queries in the TREC Fair Ranking track 2020 data.

datasets. Fig. 2 shows the mean, maximum, and minimum of papers' citation counts for all search sessions in the data. There is a high variance between mean and maximum citations, which implies that the data is outlier-prone based on this feature. We plot outlier counts for each position in the top-20. Fig. 3 depicts the number of relevant and non-relevant outliers detected by the outlier detection methods introduced in Section 2 at different positions. The stacked bars show that in spite of the attractiveness of outliers, most of these items are irrelevant, judging by the click data. In total, 88.5%, 89.8%, and 90.1% of the outliers are irrelevant when MedKNN, COPOD, and MAD are employed as the outlier detection method, respectively. The average percentage of irrelevant documents in the top 20 positions in the dataset is 83.3%. This suggests that by pushing these items to lower positions we can improve the degree of outlierness without jeopardizing utility.

Do users perceive outliers in the ranking? Outliers are present in realistic datasets, but do they impact user behavior? Prior studies stress the importance of relationships between ranked items [30], but it is unknown how an outlier in a ranking affects the examination probability. To address this gap, we conduct an eye-tracking study. We ask participants to interact with search engine result pages, as they normally would, and find the items that they prefer and think are relevant. We track their eye movements via an online webcam-based eye tracking service.[3] We use two scenarios, e-commerce, and scholarly search. We focus on a list view; in both scenarios, participants are able to see all items in one page. For each scenario, we include two result pages, one without an outlier item and one with (as in Fig. 1). In the absence of outliers we expect participants to follow the position bias examination assumption [19]; in the presence of outliers, we expect that users' attention is diverted towards them.

We recruit 40 university students and staff for both scenarios. In the instructions, we describe the overall goal of the research and ask participants to read the instructions carefully. We describe what webcam-based eye tracking is and that the eye-tracking service will ask them for access to their webcam. We instruct participants to first read and understand the query and then start scanning the result page as if they submitted the query themselves.

For reporting, we consider four eye-tracking measures based on participants' eye fixations: (1) fixation count (the number of fixations within an area; more fixation means more visual attention); (2) time spent (shows the amount of time that participants spent on average looking at an area); (3) Time To First Fixation (TTFF; the amount of time that it takes participants on average to look at one area for the first time); and (4) revisit count (indicates how many times on average participants looked back at the area) [15].

[3]RealEye, https://realeye.io.

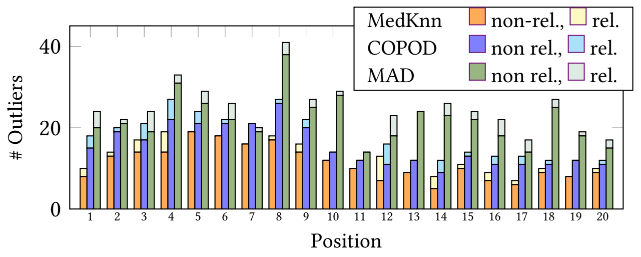


Figure 3: Number of outliers at each position w.r.t. different outlier detection methods, considering the number of citations as the observable feature. Each stacked bar shows the number of irrelevant and relevant outliers.

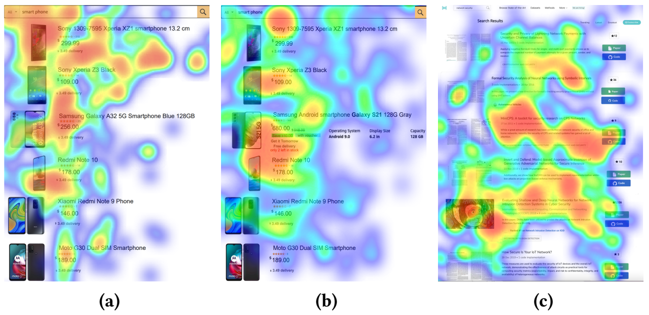
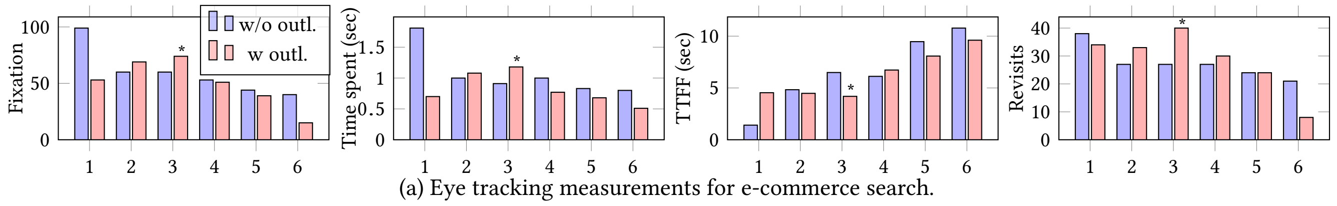


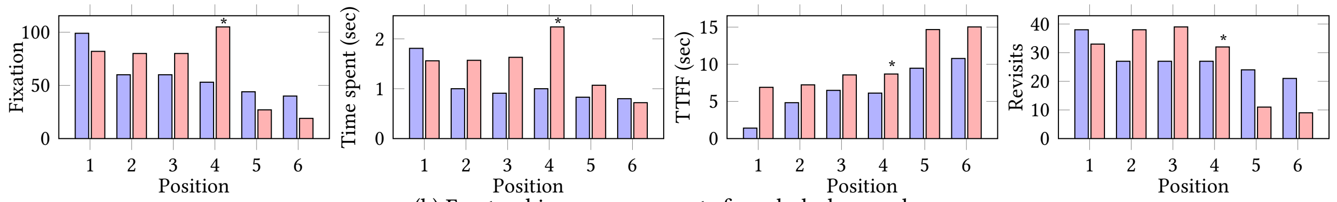
Figure 4: Examples used in the eye-tracking study. (a) Heatmap for a result list without outlier for the query "smart phone". Items at top ranks receive more attention, following the position bias assumption. (b) Heatmap for a similar page (the same list as in Fig. 1) but with one outlier, at position 3. Participants exhibit increased attention towards and around the outlier item. (c) Heatmap for a scholarly search example, with an outlier (at position 4).

Outliers in e-commerce. For this scenario, we mimick the Amazon Marketplace search result page.Fig. 1 depicts our example ranked list with an outlier. The third item in the list stands out from other items for different reasons, including price and sales-related tags (e.g., being on sale), as well as other information that is available for this item. Comparing Fig. 4a and 4b, we see that in the presence of an outlier, items at the top of the list receive less attention, contradicting the position bias assumption.

Fig. 5(a) reports the eye-tracking measures for the e-commerce scenario. We highlight the outlier in each list with an asterisk. In the no-outlier condition, participants exhibit linear behavior in terms of scanning the items. The highest number of fixations, time spent, revisits belongs to the items on the top of the list, and it decreases as we go down the list. TTFF demonstrates a linear behavior of the first fixation time, that is, most of the participants started scanning the results from top to bottom. The ranked list with an outlier exhibits very different measurements. Attention is more focused on the outlier item. Also, we see that from the TTFF values, the average time for the first fixation is the lowest for this item, suggesting that the scanning order and exposure are influenced by the existence of the outlier. This is also evident by comparing the heatmaps in Fig. 4a and 4b.



(a) Eye tracking measurements for e-commerce search.



(b) Eye tracking measurements for scholarly search.

Figure 5: Eye tracking measurements for each position, based on participants' eye fixations. The positions where outliers were shown are marked with an asterisk.

Outliers in scholarly search. In the second scenario, we study the effect of outliers on scholarly search result pages. To this end, we mimick the result page from PapersWithCode.[4]

To save space, we omit the eye fixation heatmap for the result page without outliers; it shows the familiar F-shape. Fig. 4c shows the eye fixation heatmap for the result page *with* an outlier item; the fourth item has a different thumbnail and a large number of GitHub stars, making this item stand out in the list. Similar to the e-commerce scenario, the eye fixations are very different from the F-shape typical for the no-outlier case. Fixations and time spent are the highest for the outlier item, suggesting that it draws lots of attention, and contradicts the cascade examination assumption.

However, different from the e-commerce case, we do not observe as big a difference in the TTFF values between the conditions with and without outliers, suggesting that the order of item scans is not affected as much as in the e-commerce example.

4 MITIGATING OUTLIERNESS IN FAIR LEARNING TO RANK

We now present a ranking algorithm for mitigating outlierness for fairness in ranking, called OMIT, that simultaneously accounts for item fairness and outlierness effect requirements. Based on the observations reported in the previous section we know that outliers can influence exposure and examination order, in a way that can be considered as a type of bias. We take a first step towards mitigating the outlierness phenomenon by proposing a remarkably simple, yet effective solution that removes outliers from the top positions where the distribution of exposure is most critical. Our solution aims at decreasing outlierness in the top-k positions, while retaining the ranking's utility and fairness with position-based assumptions.

OMIT is based on the linear programming method described in Section 2. In addition to optimizing for user utility while staying within the fairness constraints, our goal is to reduce the number of outliers in the top-k of all rankings. OMIT has two steps. In the first, we search for the marginal rank probability matrix that satisfies item group fairness by solving a linear program that optimizes both for user utility and fewer outliers in the top-k items with linear constraints for fairness. In the second, we derive a stochastic ranking policy from the solution of the linear programming method using the Birkhoff-von Neumann decomposition [7]; cf. also [32].

Step 1: Computing MRP matrix. Let $\mathcal{D}_q = \{d_1, \ldots, d_N\}$ be a set of items that we aim to rank for a given query q. Each ranking $\sigma(\cdot|q)$ corresponds to some permutation matrix P_σ that re-orders the elements of $\mathcal{D}_q$.[5] As discussed in Section 3, to determine which items are outliers, we use the domain specific observable item features $g(d_1), \ldots, g(d_N)$ that potentially impact the user's perception of an item. Using these characteristics as features, one can use any outlier detection method to determine which items should be considered outliers. The majority of outlier detection methods, including the ones we use (Section 3), find outliers by calculating scores that indicate the degree of outlierness. This results in a list of outlierness values $\mathrm{M}(g(d_1)|\mathcal{D}), \ldots, \mathrm{M}(g(d_n)|\mathcal{D})$, where $n \leq N$ is the size of the outlierness context that the algorithm takes into account while detecting the outliers. We define a vector $\mathbf{o}_\mathcal{D}$, that contains, for each document, the information whether it is an outlier with respect to the full ranked list:

$$\mathbf{o}_\mathcal{D} = \{o_i\}_i \quad \text{with } o_i = \begin{cases} \mathrm{M}(g(d_i)|\mathcal{D}), & \text{if } d_i \text{ is outlier in } \mathcal{D} \\ 0, & \text{else.} \end{cases} \tag{5}$$

We use $\mathbf{o}$ and $\mathbf{o}_\mathcal{D}$ interchangeably. Note that we are considering outliers in the context of the full list, $n = N$, i.e., the outlier detection algorithm takes the whole ranked list as input. We assume that items that are perceived as outliers in this context are likely to be perceived as outliers when seen in the smaller context of the top-k items. Below, we show that this heuristic works well in practice.

OMIT works by pushing outliers away from the top-k. Let P_σ be the permutation matrix corresponding to a ranking σ. The amount of outlierness that a ranking σ places in the top-k are equal to

$$\mathrm{Outlierness}_k^\mathcal{D}(\sigma|\,\mathrm{M}) = \sum_{i=0}^{k} (\mathbf{o}^T P_\sigma)_i = \mathbf{o}^T P_\sigma \mathbf{h}, \tag{6}$$

where $\mathbf{h} = (1, \ldots, 1, 0, \ldots, 0)$ is a vector containing 1 for the first k positions and 0 for all positions after that. Similarly, the expected outlierness, that is placed in the top-k by $\mathbf{P}$ is given by

$$\mathrm{Outlierness}_k^\mathcal{D}(\mathbf{P}|\,\mathrm{M}) = \mathbf{o}^T \mathbf{P}\mathbf{h}. \tag{7}$$

[4]https://paperswithcode.com

[5]For simplicity, we interchangeably use $\sigma(\cdot)$ and $\sigma(\cdot|q)$, as well as $\mathcal{D}$ and $\mathcal{D}_q$.

Algorithm 1 Outlier mitigation for fair ranking (OMIT)

```
Input: $\mathcal{D}_q$, M, $n$, $k$
Output: $\pi$
1: Create $\mathbf{o}_{\mathcal{D}}$ as Eq. 5 using $\mathcal{D}_q$ and M
2: $\mathbf{h} \leftarrow (h_1, \dots, h_n)$ such that $h_i = 1$ if $i \leq k$ else 0
3: $\mathbf{P} \leftarrow \arg\max_{\mathbf{P}} \mathbf{u}^T\mathbf{P}\mathbf{v} - \mathbf{o}^T\mathbf{P}\mathbf{h}$ such that $\mathbf{P}$ is doubly-stochastic
   and fair (Eq. 8)
4: $\pi \leftarrow$ BvN-Decomposition($\mathbf{P}$)
5: Return $\pi$
```

While optimizing for user utility we can use the expected outlierness to add an objective that will function as a regularization term, penalizing ranking policies with outliers in the top-k. We extend the linear programming approach described in Section 2 to solve:

$$\begin{aligned} \mathbf{P} = \arg\max_{\mathbf{P}} \mathbf{u}^T\mathbf{P}\mathbf{v} - \mathbf{o}^T\mathbf{P}\mathbf{h} \\ \text{such that } \mathbf{P} \text{ is doubly-stochastic} \\ \mathbf{P} \text{ is fair.} \end{aligned} \tag{8}$$

For item fairness, we adopt the disparate treatment constraints as described in Section 2. Both terms of the optimization objective, and the constraints for fairness and finding a doubly stochastic matrix are linear in N^2 variables. Hence, the resulting linear program can be solved efficiently using standard optimization algorithms [32].

Step 2: Constructing a stochastic ranking policy. The solution to the linear program $\mathbf{P}$ is a matrix indicating the probability of showing each item at any position. To generate actual rankings, we need to derive a stochastic ranking policy π from $\mathbf{P}$ and sample rankings σ to present to users. We follow [32] and use Birkhoff-von Neumann decomposition to compute π, which decomposes the doubly stochastic matrix $\mathbf{P}$ into the convex sum of permutation matrices $P = \theta_1 P_{\sigma_1} + \cdots + \theta_n P_{\sigma_n}$, with $0 \leq \theta_i \leq 1, \sum_i \theta_i = 1$ [7]. This results in at most $(N-1)^2 + 1$ rankings σ_i [23], corresponding to P_{σ_i}, that are shown to the user with probability θ_i, respectively.[6]

OMIT model summary. Algorithm 1 presents an overview of OMIT. OMIT takes as input the initial ranking D_q (optimized for utility), outlier detection method M, outlierness context size n, and the number of top items that we aim to remove outliers k. In line 1, $\mathbf{o}_{\mathcal{D}}$ is created for a given outlier detection technique and outlierness context size, followed by line 2 where we create the $\mathbf{h}$ list that takes into account the top of the list that we aim to mitigate outlierness. In line 3, we solve the linear program that jointly solves the fairness and outlierness problem and pass the stochastic ranking in line 4 to the BvN decomposition algorithm. Finally, we return the output of the BvN method as the output.

5 EXPERIMENTAL SETUP

We target the following research questions: (RQ1) How do different outlier detection methods affect OMIT's performance? (RQ2) How does our OMIT trade-off between utility, fairness, and outlierness, compared to baselines? (RQ3) We adopt the constraints proposed in [32] (called FOE) to optimize a ranked list for fairness and utility through linear programming, as described in Section 4. Given that OMIT adds additional constraints, is the overall linear program more effective when we treat the doubly stochastic matrix constraints as hard or soft constraints? (RQ4) How does changing the context of detecting outliers affect OMIT's outlierness improvement and utility? (RQ5) How does changing k affect OMIT's outlierness improvement and utility in the top-k positions?

Data. We use data from the TREC Fair Ranking 2019 and 2020 track (see Section 3). We make the group definitions over the two datasets consistent. As for the TREC 2019 data, we bin the original article groups into two classes. For the TREC 2020 data, we adopt the group definitions from the original data, that is, documents are assigned to two groups based on their authors' h-index. Moreover, we follow the TREC setup to generate query sequences, leading to multiple occurrences of the same query, using the provided frequencies. Specifically, we evaluate on a query sequence of size 10, 000, including all the queries in the evaluation data.

Evaluation metrics. We evaluate methods for fair learning to rank in the presence of outliers in terms of utility, item fairness, and outlierness. For utility and fairness, we use NDCG and dTR (see Eq. 2), respectively, as our metrics and report their expected values. To measure the expected outlierness of the policy $\mathbf{P}$ up to position n in the ranking, we use $\text{Outlierness}_n^{\mathcal{D}}(\mathbf{P}|\,\text{M})$ as defined in Eq. 7. Similarly we define the expected number of outliers up to position n in ranking for policy $\mathbf{P}$ as

$$\#\text{Outliers}_n^{\mathcal{D}}(\mathbf{P}|\,\text{M}) = \mathbf{o}_b^T\mathbf{P}\mathbf{h}, \tag{9}$$

where $\mathbf{o}_b^T = \mathbb{1}_{>0}(\mathbf{o}^T)$ is the binarized version of $\mathbf{o}^T$ where each outlier item is assigned 1, and all the rest are assigned 0.

Compared methods. To evaluate OMIT, we build several baseline methods, combining different options for each component of our model (initial ranking, fairness of exposure, outlier mitigation):

- **Initial ranking:** The initial ranking of all compared methods is generated using ListNet [9]. ListNet is a learning to rank model, optimizing for utility. We use it in our experiments to create initial ranked lists, $\mathcal{D}_q$, using the click data provided in the training set, with 30 maximum epoch, and a validation ratio of 0.3.
- **Fairness of exposure**: We use two variants of FOE [32] based on hard vs. soft doubly stochastic matrix constraints, and call them FOE^H and FOE^S, respectively.[7]
- **Outlier mitigation**: We employ two outlier mitigation techniques, namely, RO and OMIT. RO removes all the outlier items detected by M from the ranking, while OMIT is our proposed outlier mitigation method as described in Section 4.

We specify methods as combinations of the three components mentioned above. E.g., "ListNet + FOE^H + OMIT" uses the initial ranked list produced by ListNet, applying FOE fairness post-processing with hard constraints and the OMIT outlier mitigation model.

6 EMPIRICAL RESULTS

Effect of outlier detection method. We address **RQ1** by changing the outlier detection method, while keeping the other parts of the model fixed. Table 2 reports the results of using three different outlier detection methods in OMIT. For comparison, we also report the results of ListNet without outlier mitigation (row **a**) and report the relative improvements. All three outlier detection methods effectively reduce outlierness compared to the baselines. COPOD achieves the best results by reducing outlierness by 80.3% and 80.6%

[6]We used the implementation from https://github.com/jfinkels/birkhoff.

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

Table 2: Comparing loss in fairness and utility, with gains in outlierness for different outlier detection methods on the TREC 2019 and 2020 Fair Ranking data. Models used: (a) ListNet **and (b)** ListNet + FOE^S + OMIT. **Δ values denote the percentage of relative improvement compared to (a). * refers to statistically significant improvements compared to (a) using a two-tailed paired t-test** ($p < 0.05$).

			NDCG ↑		Fairness ↑	# Outliers ↓		Outlierness ↓	
	Model	Outl.	@5	@10	dTR	@10	Δ(%)	@10	Δ(%)
TREC 2019	(a)	COPOD	0.671	0.757	0.982	1.260	–	0.873	–
		MedKNN	0.671	0.757	0.982	0.507	–	0.432	–
		MAD	0.671	0.757	0.982	0.811	–	0.599	–
	(b)	COPOD	0.667	0.753	**0.995***	0.208*	**83.49**	0.172*	**80.29**
		MedKNN	**0.671**	0.756	0.991	0.102*	79.88	0.094*	78.24
		MAD	**0.671**	**0.757**	0.990	0.290*	64.24	0.205*	65.77
TREC 2020	(a)	COPOD	0.240	0.356	0.267	1.043	–	0.755	–
		MedKNN	0.240	0.356	0.267	0.783	–	0.602	–
		MAD	0.240	0.356	0.267	1.456	–	0.779	–
	(b)	COPOD	0.240	0.366*	**0.279***	0.178*	**82.93**	0.146*	**80.66**
		MedKNN	0.239	0.361	0.263	0.160*	79.56	0.133*	77.90
		MAD	**0.242**	**0.372***	**0.278***	0.430*	70.46	0.202*	74.10

in terms of Outlierness@10 on the TREC 2019 and 2020 data, respectively. In terms of dTR, COPOD outperforms MAD and MedKNN on both datasets where it increases dTR by 4.5% on TREC 2020. We see no significant difference in the utility achieved by the methods. Given that COPOD is parameter-free and scalable, we prefer it over the other two methods. For the remaining experiments, we choose COPOD as the outlier detection method.

Utility, fairness, and outlierness trade-offs. To answer **RQ2**, we turn to Table 2, which shows the results for OMIT and the baseline methods in terms of utility, fairness, and outlierness. Although ListNet is purely optimized for utility, it does not achieve the highest NDCG in all cases. This suggests that optimizing for fairness and outlierness could even improve the utility. As shown in Fig. 3 there is a high density of outliers among top positions that are mostly irrelevant. Therefore, when OMIT pushes these items to lower positions, it improves outlierness and utility measures simultaneously. Moreover, we see that mitigating outlierness does not cause any significant effect on dTR, showing that OMIT is capable of retaining position-based item fairness. Table 3 shows that OMIT effectively decreases the number of outliers in top-10 positions by at most 83.49% (and 82.93%) when used with FOE^S (row **g** in the table) on the TREC 2019 (and 2020) data. For the outlierness metrics, these values are 80.29% and 80.66%. From our data analysis, we observed a high density of non-relevant outlier items at the top of the list, indicating the high possibility of user distraction towards these non-relevant items, as suggested by our eye-tracking study.

Hard vs. soft constraint. We turn to **RQ3** and experiment with two variants of the FOE model, which differ in the constraint for generating a doubly stochastic matrix (see Eq. 8). This constraint is important since the BvN algorithm can guarantee to generate valid permutations only if the input is doubly stochastic. We observed that forcing the convex optimization to output such matrices can make the constraints too hard to satisfy even when only optimizing for fairness and utility. Hence, the algorithm cannot find an optimum solution for many of the queries. For example, FOE^H cannot

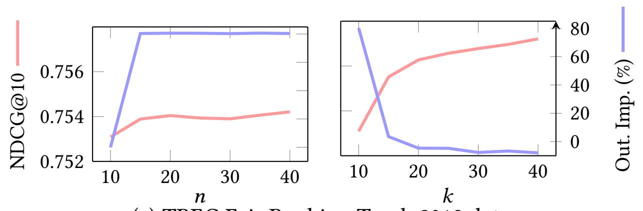
Table 3: Comparing loss in fairness and utility, with gains in outlierness for COPOD on the TREC 2019 and TREC 2020 Fair Ranking data. Models used: (a) ListNet; **(b)** ListNet+FOE^H; **(c)** ListNet+FOE^S; **(d)** ListNet+FOE^H+RO; **(e)** ListNet+FOE^S+RO; **(f)** ListNet + FOE^H + OMIT; **(g)** ListNet + FOE^S + OMIT. **Δ values denote the percentage of relative improvement compared to (a). Other conventions are the same as in Table 2.**

		NDCG ↑		Fairness ↑	# Outliers ↓		Outlierness ↓	
	Model	@5	@10	dTR	@10	Δ(%)	@10	Δ(%)
TREC 2019	(a)	0.671	0.757	0.982	1.260	–	0.873	–
	(b)	0.670	0.756	0.991	1.235	–	0.870	–
	(c)	**0.673**	**0.758**	0.982	1.225	–	0.852	–
	(d)	0.663	0.697	**0.996***	1.114	11.58	0.861	1.37
	(e)	0.667	0.700	0.965	1.072*	14.92	0.834	4.47
	(f)	0.672	0.757	**0.996***	1.080*	14.28	0.753*	13.74
	(g)	0.667	0.753	0.995*	0.208*	**83.49**	0.172*	**80.29**
TREC 2020	(a)	0.240	0.356	0.267	1.043	–	0.755	–
	(b)	0.237	0.355	0.249	1.073	–	0.780	–
	(c)	0.241	0.357	0.262	1.046	–	0.758	–
	(d)	**0.242**	0.362	**0.293***	1.143	−9.58	0.811	−7.41
	(e)	**0.242**	0.362	0.269	1.148	−10.06	0.817	−8.21
	(f)	0.237	0.359	0.282*	0.885*	15.15	0.645*	14.56
	(g)	0.240	**0.366***	0.279*	0.178*	**82.93**	0.146*	**80.66**

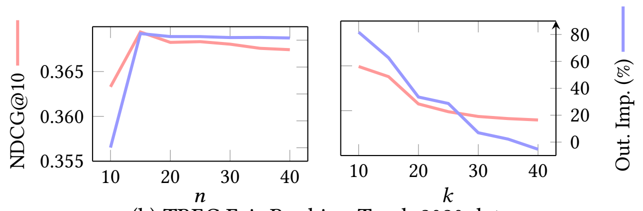
find solutions for 47%, and 46% of the queries on TREC 2019 and 2020 data, respectively. We return the original ranking as the output when FOE^H does not find an optimum solution. To fix this problem we implemented the constraint for doubly stochastic matrix as a soft constraint and we check for validity of the permutation matrices generated by the decomposition algorithm. Table 3 demonstrates the effectiveness of the soft variant of FOE (row **g** vs. **f**).

Effect of n. To answer **RQ4**, we examine the effect of the n parameter, which determines the outlierness context size. We change n from 10 to 40 items while keeping other parameters fixed, and observing the behavior of the model. This is important since the outlierness of an item depends on its context, e.g., an item can be considered as an outlier in the top-10 items, but may not be an outlier in the top-20 items. The two left plots in Fig. 6 show the outlierness improvements (compared to ListNet) and utility in terms of NDCG@10 for different values of n on the TREC 2019 and 2020 data. We see that Outlierness@10 improves for larger values of n, suggesting that determining outlierness of items in a bigger pool of items is more accurate and allows OMIT to mitigate the outliers in the top-10 positions more effectively.

Effect of k. To answer **RQ5**, we study the effect of changing k when optimizing for mitigating outlierness in top-k positions. We are interested in finding out how utility and outlierness are influenced when optimizing for mitigating outlierness for a longer list of top ranks ranging from 10 to 40. This mimics the cases where more items can be shown to a user, hence outliers in longer lists can be observed by the user. The left plots in Fig. 6 depict the results for both datasets. We observe that outlierness improvement drops for greater values of k since it is more challenging for the algorithm to push all outliers to lower positions. Fig. 3 shows that most outliers are located at top positions, so greater values of k do not necessarily translate to more outliers. The changes in utility scores of the lists



(a) TREC Fair Ranking Track 2019 data.



(b) TREC Fair Ranking Track 2020 data.

Figure 6: NDCG@10 and outlierness@10 improvement percentage for different values of k and different sizes of the window in which we detect the outliers.

are marginal, with a 0.5% increase and 1.6% decrease for larger values of k for the TREC 2019 and 2020 data, respectively. The difference in utility of the two datasets is due to the fact that there are more relevant outliers in TREC 2019 than in TREC 2020.

7 RELATED WORK

Bias in implicit feedback. Users' implicit feedback, such as clicks, can be a great source of relevance judgment that has been shown to help improve search quality [2]. Clicks may be misleading due to different types of bias, which causes the probability of a click to differ from the probability of relevance. Recent work on discovering and correcting for different types of bias in logged click data concerns position bias [19, 20], presentation bias [40], selection bias [29], trust bias [2, 35], popularity bias [1], and recency bias [12]. Inter-item dependencies affect the perceived relevance of items [13]. We introduce a phenomenon that is anchored in inter-item dependencies and may result in biased clicks. Our work differs from previously discussed types of bias by considering inter-item relationships. Showing items as outliers can make them more attractive to users, influencing their perceived relevance. Presentation bias [40] concerns a related effect; bold keywords in titles make some items appear more attractive. However, this definition of attractiveness is independent of other items in the list. Metrikov et al. [25] show that click-through rates can be manipulated by adding more images to top positions next to the ad slots on the search result page; they did not study the effect on item exposure or biased clicks. We focus on the effect of outliers, as an inter-item dependency on the examination probability and item exposure, and introduce it as a potential source of bias in click data.

Fair ranking. Following [42], we distinguish two ways of measuring fairness of rankings. Work on *probability-based* fairness determines the probability that a ranking is the result of a fair process [4, 10, 11, 16, 34, 39, 41]. *Exposure-based* methods determine the expected exposure for each item in the ranking and aim to ensure that this exposure is fairly distributed [6, 14, 24, 26, 31–33, 38]. Our work belongs to the second category. To estimate the expected exposure of each item or group, we need to take into account different types of bias that the user might have when observing system outputs. Previous work has mostly focused on position bias [6, 32, 38]. We emphasize the role of inter-item dependencies in the exposure that an item receives, which can be a source of unfairness when not considered in computing the expected exposure. We extend the re-ranking approach introduced in [32], to not only produce fair rankings but also avoid showing the outliers in the top-k.

An important effort for developing a benchmark for evaluating retrieval systems in terms of fairness is the *TREC Fair Ranking track* (see footnote 1). We expand the use of the track's resources to include the study of outliers in fair ranking.

8 CONCLUSION & FUTURE WORK

We introduced and studied a phenomenon related to fair ranking, that is, outlierness. We analyzed data from the TREC Fair Ranking track and found a significant number of outliers in the rankings. We hypothesized that the presence of outliers in a ranking impacts the way users scan the result page. We confirmed this hypothesis with an eye-tracking study in two scenarios: e-commerce and scholarly search. We proposed OMIT, an approach to mitigate the existence and effect of outliers in a ranked list. With OMIT, we introduced a ranking constraint based on the outlierness of items in a ranking and combined it with fairness constraints. We formulated the problem of outlier mitigation as a linear programming problem and produced stochastic rankings, given an initial ranking. Using OMIT one can reduce outliers in rankings without compromising user utility or position-based item fairness. We analyzed the effects of different outlier detection approaches and compared their results. Our experiments also showed that there is a trade-off between the depth of outlier detection and user utility. Now that we have established the impact of outliers in rankings, future work on fair ranking should consider the presence of outliers by default.

One limitation of our work is our focus on removing the bold outliers defined in the context of the whole list from the top-k positions. We plan to improve our model to mitigate the outlierness of all sliding windows of size k. In addition, we want to improve the performance for cases where outliers are relevant items, e.g., by considering alternative methods of outlier mitigation. Finally, a natural extension of our work could be to quantify the effect that an outlier in the presentation of the ranking can have on the examination probability distribution; this could open the door to unbiased learning to rank approaches that counter the outlier bias in logged user data.

DATA AND CODE

To facilitate reproducibility of our work, all code and parameters are shared at https://github.com/arezooSarvi/OMIT_Fair_ranking.

ACKNOWLEDGMENTS

This research was supported by Ahold Delhaize, the NWO Innovational Research Incentives Scheme Vidi (016.Vidi.189.039), and the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, https://hybrid-intelligence-centre.nl. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling Popularity Bias in Learning-to-rank Recommendation. In *RecSys*. 42–46.
[2] Aman Agarwal, Xuanhui Wang, Cheng Li, Michael Bendersky, and Marc Najork. 2019. Addressing Trust Bias for Unbiased Learning-to-rank. In *WWW*. 4–14.
[3] Fabrizio Angiulli and Clara Pizzuti. 2002. Fast Outlier Detection in High Dimensional Spaces. In *ECML PKDD*. 15–27.
[4] Abolfazl Asudeh, HV Jagadish, Julia Stoyanovich, and Gautam Das. 2019. Designing Fair Ranking Schemes. In *SIGMOD*. 1259–1276.
[5] Ricardo Baeza-Yates. 2018. Bias on the Web. *Commun. ACM* 61, 6 (2018), 54–61.
[6] Asia J Biega, Krishna P Gummadi, and Gerhard Weikum. 2018. Equity of Attention: Amortizing Individual Fairness in Rankings. In *SIGIR*. 405–414.
[7] Garrett Birkhoff. 1940. *Lattice Theory*. AMS.
[8] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A Neural Click Model for Web Search. In *WWW*. 531–541.
[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 *ICML*. 129–136.
[10] L Elisa Celis, Anay Mehrotra, and Nisheeth K Vishnoi. 2020. Interventions for Ranking in the Presence of Implicit Bias. In *FAT**. 369–380.
[11] L Elisa Celis, Damian Straszak, and Nisheeth K Vishnoi. 2018. Ranking with Fairness Constraints. In *ICALP*. 28:1–28:15.
[12] Ruey-Cheng Chen, Qingyao Ai, Gaya Jayasinghe, and W Bruce Croft. 2019. Correcting for Recency Bias in Job Recommendation. In *CIKM*. 2185–2188.
[13] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. *Click Models for Web Search*. Morgan & Claypool Publishers.
[14] Fernando Diaz, Bhaskar Mitra, Michael D Ekstrand, Asia J Biega, and Ben Carterette. 2020. Evaluating Stochastic Rankings with Expected Exposure. In *CIKM*. 275–284.
[15] Susann Fiedler, Michael Schulte-Mecklenbeck, Frank Renkewitz, and Jacob L Orquin. 2020. Guideline for Reporting Standards of Eye-tracking Research in Decision Sciences. *PsyArXiv preprint* (2020).
[16] Sahin Cem Geyik, Stuart Ambler, and Krishnaram Kenthapadi. 2019. Fairness-aware Ranking in Search & Recommendation Systems with Application to Linkedin Talent Search. In *SIGKDD*. 2221–2231.
[17] Boris Iglewicz and David Caster Hoaglin. 1993. *How to Detect and Handle Outliers*. ASQ Quality Press.
[18] Wen Jin, Anthony K. H. Tung, Jiawei Han, and Wei Wang. 2006. Ranking Outliers Using Symmetric Neighborhood Relationship. In *PAKDD*. 577–593.
[19] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *SIGIR*. 154–161.
[20] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-rank with Biased Feedback. In *WSDM*. 781–789.
[21] Zheng Li, Yue Zhao, N Botta, C Ionescu, and X COPOD Hu. 2020. COPOD: Copula-based Outlier Detection. In *ICDM*. 17–20.
[22] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.
[23] Marvin Marcus and Rimhak Ree. 1959. Diagonals of Doubly Stochastic Matrices. *The Quarterly Journal of Mathematics* 10, 1 (1959), 296–302.
[24] Rishabh Mehrotra, James McInerney, Hugues Bouchard, Mounia Lalmas, and Fernando Diaz. 2018. Towards a Fair Marketplace: Counterfactual Evaluation of the Trade-off between Relevance, Fairness & Satisfaction in Recommendation Systems. In *CIKM*. 2243–2251.
[25] Pavel Metrikov, Fernando Diaz, Sebastien Lahaie, and Justin Rao. 2014. Whole Page Optimization: How Page Elements Interact with the Position Auction. In *EC*. 583–600.
[26] Marco Morik, Ashudeep Singh, Jessica Hong, and Thorsten Joachims. 2020. Controlling Fairness and Bias in Dynamic Learning-to-rank. In *SIGIR*. 429–438.
[27] Maeve O'Brien and Mark T Keane. 2006. Modeling Result-list Searching in the World Wide Web: The Role of Relevance Topologies and Trust Bias. In *CogSci*. 1881–1886.
[28] Harrie Oosterhuis and Maarten de Rijke. 2021. Unifying Online and Counterfactual Learning to Rank: A Novel Counterfactual Estimator that Effectively Utilizes Online Interventions. In *WSDM*. 463–471.
[29] Zohreh Ovaisi, Ragib Ahsan, Yifan Zhang, Kathryn Vasilaky, and Elena Zheleva. 2020. Correcting for Selection Bias in Learning-to-rank Systems. In *WWW*. 1863–1873.
[30] Przemysław Pobrotyn, Tomasz Bartczak, Mikołaj Synowiec, Radosław Białobrzeski, and Jarosław Bojar. 2020. Context-aware Learning to Rank with Self-attention. *SIGIR eCom* (2020).
[31] Piotr Sapiezynski, Wesley Zeng, Ronald E Robertson, Alan Mislove, and Christo Wilson. 2019. Quantifying the Impact of User Attentionon Fair Group Representation in Ranked Lists. In *WWW*. 553–562.
[32] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. In *KDD*. 2219–2228.
[33] Ashudeep Singh and Thorsten Joachims. 2019. Policy Learning for Fairness in Ranking. In *NeurIPS*.
[34] Julia Stoyanovich, Ke Yang, and HV Jagadish. 2018. Online Set Selection with Fairness and Diversity Constraints. In *EDBT*. 241–252.
[35] Ali Vardasbi, Harrie Oosterhuis, and Maarten de Rijke. 2020. When Inverse Propensity Scoring does not Work: Affine Corrections for Unbiased Learning to Rank. In *CIKM*. 1475–1484.
[36] Lequn Wang and Thorsten Joachims. 2021. User Fairness, Item Fairness, and Diversity for Rankings in Two-Sided Markets. In *SIGIR*. 23–41.
[37] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to Rank with Selection Bias in Personal Search. In *SIGIR*. 115–124.
[38] Himank Yadav, Zhengxiao Du, and Thorsten Joachims. 2019. Policy-Gradient Training of Fair and Unbiased Ranking Functions. *arXiv preprint arXiv:1911.08054* (2019).
[39] Ke Yang and Julia Stoyanovich. 2017. Measuring Fairness in Ranked Outputs. In *SSDBM*. 1–6.
[40] Yisong Yue, Rajan Patel, and Hein Roehrig. 2010. Beyond Position Bias: Examining Result Attractiveness as a Source of Presentation Bias in Clickthrough Data. In *WWW*. 1011–1018.
[41] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. FA*IR: A Fair Top-k Ranking Algorithm. In *CIKM*. 1569–1578.
[42] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2021. Fairness in Ranking: A Survey. *arXiv preprint arXiv:2103.14000* (2021).