FairRankTune: A User-Friendly Toolkit Supporting Fair Ranking Tasks

KATHLEEN CACHEL, Worcester Polytechnic Institute, USA
ELKE RUNDENSTEINER, Worcester Polytechnic Institute, USA

We present FairRankTune, an open-source Python toolkit supporting end-to-end fair ranking workflows, analysis, auditing, and experimentation. FairRankTune provides researchers, practitioners, and educators with a self-contained module for generating ranked data, ranking strategies, and popular ranking-based fairness metrics. The central piece of FairRankTune, is the introduction of a group fairness-tunable ranked data generator, RankTune, that is the first to streamline the creation of custom fairness-relevant ranked datasets. RankTune advances existing typically informal data generation strategies. Its unique feature is a fairness-tuning mechanism for controlling the fairness of generated rankings with respect to any number of protected groups, while also providing the means to create multi-ranking datasets with similar and diverse fairness degrees. Our toolkit also offers fair ranking metrics and fairness-aware ranking strategies side-by-side so toolkit users can directly utilize these diverse fairness tools on the generated ranking data within one integrated platform. This not only closes the gap of limited publicly-available metric and algorithm implementations for fair ranking, but also removes the friction of working with disparate software. We illustrate the utility of our FairRankTune toolkit by demonstrating its use in constructing customized ranked datasets, comparing the RankTune generator against current data generation strategies, and via a case study analyzing differences in fair ranking metrics. FairRankTune simplifies fairness workflows for auditing and conducting research while promoting reproducibility and ease of use.

CCS Concepts: • **Information systems** → *Retrieval models and ranking*; • **Computing methodologies** → **Interactive simulation**.

Additional Key Words and Phrases: Fair Ranking, Algorithmic Fairness, Data Generation, Bias Measurement, Bias Mitigation

ACM Reference Format:
Kathleen Cachel and Elke Rundensteiner. 2023. FairRankTune: A User-Friendly Toolkit Supporting Fair Ranking Tasks. In *Non-archival Track at Equity and Access in Algorithms, Mechanisms, and Optimization (EAAMO '23), October 30– November 01, 2023, BOSTON, MA.* ACM, New York, NY, USA, 19 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

1.1 Background

Over the last several years, fairness in ranking has been an active research area due to the ubiquitous nature and societal impact of ranking-based tasks. A significant branch of fair ranking research focuses on *group-fairness* [5, 19, 25, 33, 46, 53, 55, 67–69], addressing how different demographic groups of candidate items are treated in a ranking system. In particular, the notion of *statistical parity* [48], meaning demographic groups receive a proportional share of favorable rank positions, has been integrated into a plethora of ranking systems [13, 25, 28, 55, 66, 68, 69] and formulated into a variety of metrics [5, 13, 19, 25, 33, 46, 53, 55, 67].

As this impactful area grows, a substantial obstacle faced by researchers is the unavailability of rich fairness-relevant ranked data [36, 47, 70]. This is in part due to the challenge of releasing real-world datasets for fairness research. Fairness analysis necessitates sensitive information such as gender, age, and race, which, when made public, can pose

Manuscript submitted to ACM

privacy risks. Legal restrictions often prevent platforms from collecting such information in the first place, along with restrictions on its use, sharing, and retention [17]. Moreover, despite the breadth of work in constructing fair ranking metrics [5, 19, 25, 33, 46, 53, 55, 67], few metrics have publicly available implementations [33, 53, 67]. Thus, there is a lack of a dedicated tool providing metrics for auditing, for research, and for educational purposes within one integrated package. Fairness-focused toolkits are almost exclusively for fair *classification* [4, 7, 18, 22, 29, 29, 31, 34, 49, 51, 59], and its corresponding metrics and algorithms – in contrast to our *ranking* objective. Thus, these toolkits are not directly applicable to users focused on fair ranking.

1.2 State-of-the-Art

Synthetic (generated) data has emerged as common approach for evaluating new fairness-enhanced technologies [2, 10, 26, 27, 45]. Generated data eliminates the need for costly laborious data collection, while presenting opportunities to study targeted custom scenarios. For this reason, in the recent fair ranking literature researchers created their own ranked data for experimental analysis in their particular study [2, 10, 26, 27, 45]. The strategy generally employed in Akpinar et al. [2], Bower et al. [10], Ghazimatin et al. [26], Ghosh et al. [27], Nandy et al. [45] was to assign items a random score (from normal or uniform distributions), and then order the items in the final ranking by sorting by this score. This approach poses two disadvantages. First, the fairness of the ranking resulting from this strategy is hard to control since the ordering produced is entirely random, but is also potentially time-consuming. And similarly, it requires many attempts to get it "just right" since there is no a priori indication of how fair or unfair the generated ranking will be.

To the best of our knowledge, there is only one fairness-focused ranked data generator, the Unfair Ranking Generator (`URG`) [67]. `URG` creates rankings based on parameter $f \in [0, 1]$. f adjusts the placement of items from the "protected group", compared to the "unprotected group", from being at the bottom of the ranking ($f = 0$) to being at the top ($f = 1$). Thus, a given f value does not behave consistently across items sets; it may generate fair or unfair rankings depending on the protected group's size. Moreover, `URG` does not support generating rankings with > 2 groups, an increasing issue as the fairness community moves away from assuming the presence of only two groups (e.g., only male or female).

Beyond data generation, `FARE` [1] [33] and `FairSearch` [2] [71] are the only available tools supporting fair ranking workflows. `FARE` presents three fair ranking metrics, with all restricted to only binary demographic groups and assumes the existence of a ground-truth ideal ranking. `FairSearch` is a companion Python tool for the works proposing FA*IR [68] and DELTR [69] ranking methods. Unfortunately, both algorithms only support two demographic groups. The current limited landscape for fairness-tunable data-generation [67] and fair ranking toolkits [33, 71] means practitioners, researchers, and students lack not only advanced user-friendly ranked data generation capabilities, but also a dedicated set of tools providing complimentary implementations of contemporary multi-group fair ranking metrics and methods.

1.3 Our Proposed Toolkit

To close the above implementation gap and reduce frictions posed by current tools, we present the `FairRankTune` toolkit. We aim to help researchers, practitioners, educators and students by providing an end-to-end fair ranking toolkit supporting data generation, bias-measurement, and bias-mitigation.

In this work, we propose a new dedicated tunable-fairness data generation strategy, called the `RankTune` tool, which we include in `FairRankTune`. `RankTune` provides the following practical capabilities: `RankTune` (1.) can produce ranked

[1] https://pypi.org/project/fare/
[2] https://pypi.org/project/fairsearchcore/

data along the entire statistical parity fairness spectrum, (2) can produce multiple rankings with the same degree of fairness, (3) provides a fairness-tuning mechanism that offers a consistent usage-pattern and interpretation across diverse item sets, and (4) supports multiple, not just binary, groups. We demonstrate the use of `RankTune` to create a fairness-relevant multi-winner voting dataset (multiple voter rankings) from a single-ranking law student dataset. We also empirically compare `RankTune` to the above data generation strategies from the literature, highlighting how it advances fairness-relevant ranked data generation capabilities and ease-of-use.

`FairRankTune` offers a `Metrics` library encompassing implementations of several fairness metrics for analyzing rankings. A key innovation of the `Metrics` library is providing toolkit users multiple choices for *how* to calculate a given top-level metric. For instance, for group exposure [55], a popular fairness criteria [27, 55, 69], `FairRankTune` offers seven ways of calculating a top-level exposure metric (e.g., min-max ratios, max absolute difference, L-2 norms of per-group exposures, etc.). This provides enhanced modularity and lowers the barrier for researchers to conduct additional studies, including comparing these formulations as well as conducting user studies on their interpretability. Additionally, we provide implementations of fair re-ranking strategies [23, 25] allowing researchers to test new fairness metrics on post-mitigation rankings and against the existing metrics in the `Metrics` library. Thus `FairRankTune` also provides educators with a standalone framework, where students can use `FairRankTune` as a test-bed to generate data, apply re-ranking strategies, and assess the fairness of the outcome.

Contributions. The resulting easy-to-use open-source Python toolkit will benefit individual practitioners as well as the fairness research community at large by providing a standardized metric library, access to customizable ranked data sets designed for fairness tasks, and the promotion of reproducibility. Contributions of this work include:

- `FairRankTune` is an open-source Python library: https://pypi.org/project/FairRankTune/. It has three main components: data generation - `RankTune` (Section 2.1), the `Metrics` library (Section 2.2), and the bias-mitigation `Rankers` module (Section 2.3).
- We present the first multi-group fairness-tunable algorithm for generating ranked data offered as the `RankTune` tool. We empirically compare it with available data generators, `URG` [67] and strategies from recent experimental setups [2, 10, 26, 27, 45], demonstrating `RankTune`'s capabilities and user-friendliness (Section 3).
- Using `FairRankTune` we conduct a case study performing the first comparative empirical analysis of statistical parity metrics in multi-group rankings. We experimentally observe, that comparing certain metrics can be misleading under specific conditions such as across datasets and different numbers of ranked items (Section 4).
- We discuss use cases of `FairRankTune`, and identify areas in which our toolkit's capabilities provide unique research opportunities (Section 5).

1.4 Related Work

Bias Mitigation and Measurement in Ranking Research in fair ranking has expanded rapidly in recent years, including numerous surveys [21, 36, 47, 72, 73]. While there are many ethical and sociotechnical considerations in designing fair ranking systems, we can predominately categorize recent advances as bias measurement [5, 19, 25, 33, 46, 53, 55, 67] or bias mitigation technologies [5, 15, 16, 25, 28, 42, 55, 56, 61, 68, 69]. Bias measurement focuses on formulating metrics to measure and quantify unfairness concerns [5, 19, 25, 33, 46, 53, 55, 67]. Our toolkit makes it easy to compare multiple metrics and conduct audits. Additionally, researchers can contrast the design of novel metrics using the metrics offered in `FairRankTune` as comparative metrics. On the other hand, bias mitigation considers the design of fairness-enhanced algorithms and fairness interventions. At a high level, fair approaches can be described as

pre, in, or post-processing. Pre-processing address bias issues in the data prior to a downstream task like ranking or classification [14, 22], but do not produce rankings. In-processing algorithms tend to be learning-to-rank methods that ensure fairness at model training time [16, 42, 56, 61, 69]. Post-processing methods take an initial ranking and re-rank items to create a fair ranking [5, 15, 25, 28, 55, 68]. Our toolkit `FairRankTune` facilitates testing the design of all such novel fair ranking methods by providing researchers with a suite of metrics, baseline strategies, and a tool to generate custom test-bed datasets.

1.5 Practitioner and Researcher Tools

Recent work has developed open-sourced fairness toolkits [3, 4, 7, 18, 22, 29, 29, 31, 34, 49, 51, 59]. As pointed out earlier, these software packages are geared toward fair *classification* - not ranking. Unlike our work, they thus do not support generating fairness-tunable ranked data or analyzing and creating fair rankings. Examples include `Aequitas` [51] a toolkit for auditing classification models for different fairness concerns, `LiFT` [59] a Spark-based tool for monitoring fairness within in-production (industrial) machine learning pipelines developed by LinkedIn, and `AI-Fairness-360` [4] a toolkit containing fair-classification methods, and metrics maintained by IBM.

Within ranking systems, there are many software packages simulating and helping researchers study sociotechnical issues [9, 24, 38, 41, 43, 52]. In contrast to our work, they do not focus on contemporary fairness notions as conceptualized by the algorithmic fairness community. Instead, they support studying diverse issues such as popularity bias, polarization, and misinformation. Examples include `SIREN` [9] a tool for studying filter bubbles, `T-RECS` [38] a simulation environment for polarization, and (mis) information, and `MARS-Gym` [52] facilitates the simulation of marketplace recommendations.

2 THE FAIRRANKTUNE TOOLKIT

We now present the the `FairRankTune` toolkit and its main components. Section 2.1 introduces our fairness-tunable ranked data generation tool, `RankTune`; describing its novel data generation strategy. Section 2.2 overviews the metrics library included in `FairRankTune`, along with `FairRankTune`'s innovative aspects for conducting research and facilitating learning. Section 2.3 describes the fair ranking strategies included in the `Rankers` module of `FairRankTune`, while Section 2.4 demonstrates the use of `FairRankTune` for creating and analyzing a multi-winner voting dataset.

2.1 RANKTUNE Data Generator

The first component of `FairRankTune` is the new fairness-tunable ranked data generator `RankTune`.

2.1.1 Notation. We use the symbol $X = x_1, x_2, ..., x_n$ to represent items to be ordered in a ranked list τ. $\tau(x_i)$ denotes the ordinal position of item x_i in the ranking τ. Items, sometimes called documents, candidates, or providers, belong to a group defined by a shared protected attribute value, such as, gender = "woman". We represent the groups associated with the items as $G = g_1, g_2, ..., g_m$. We use $D_X = (p_{X:g_1}, ..., p_{X:g_m})$ to represent the distribution of groups in the item set X, where the proportion of each group is $p_{X:g_m} = |g_m|/|X|$. For example, $D_X = (0.2, 0.3, 0.5)$ indicates that g_1 corresponds to 20% of items in X, and g_2 and g_3 to 30% and 50% of X, respectively.

2.1.2 Underlying Core Idea. `RankTune` is a *fairness-tunable* ranked data generation method. It constructs a ranking τ by placing items into τ from top to bottom. The idea behind `RankTune` is that to construct a "fair" ranking, each time we place an item in the generated ranking, the likelihood of placing an item $\in g_i$ should be equal to g_i's proportion of the items (i.e., if $p_{x:g_i} = 0.2$, meaning g_i is 20% of X then g_i should have a 20% chance of being placed). Then, on the other side of the spectrum, if we want a completely "unfair" ranking, we should place items into τ such that groups are ordered

Algorithm 1 Pseudocode of Core RankTune Data Generation Methodology

Input: Item set X and groups G, unfairness tuning parameter ϕ.
Output: Ranking τ.

1: $\tau \leftarrow []$
2: $D_X \leftarrow (p_{X:g_1}, ..., p_{X:g_m})$ // Each group's proportion of X
3: $g_{min} \leftarrow$ group id of smallest group
4: $\phi_{scaled} \leftarrow (1-\phi) * (1 - p_{X:g_{min}}) + p_{X:g_{min}}$
5: $D_{target} \leftarrow (D_X / (1 - p_{X:g_{min}}) * (1 - \phi_{scaled})$
6: $D_{target}(g_{min}) \leftarrow \phi_{scaled}$ // so $\sum D_{target} = 1$
7: $lowCount \leftarrow 0$
8: $lowBound$ & $upperBound \leftarrow$ empty arrays of size $|G|$
9: **for each** $g_i \in G$ **do** // Map Target Distribution to $[0, 1]$
10: $lowBound[g_i] \leftarrow lowCount$
11: $upperBound[g_i] \leftarrow lowCount + D_{target}(g_i)$
12: $lowCount \leftarrow lowCount + D_{target}(g_i)$
13: **while** each group has unplaced items **do**
14: $r \leftarrow random([0, 1])$
15: $g_{2p} \leftarrow$ s.t. $lowBound[g_{2p}] < r < upperBound[g_{2p}]$
16: $\tau.append($next item from group $g_{2p})$
17: $G_{rem} \leftarrow$ sorted (smallest to largest) groups with unplaced items
18: **for each** $g_i \in G_{rem}$ **do** // place items by increasing group size
19: **for each** $x_j \in g_i$ **do**
20: $\tau.append(x_j)$
21: **return** τ

by increasing size from small to large. In this way, smaller groups would get bigger proportions of favorable positions, which maximally violates statistical parity fairness [32]. These are the two ends of the statistical parity spectrum.

To generate rankings along this spectrum, RankTune samples a random number r in the $[0, 1]$ interval each time it places an item. We design this interval to have "regions" that map to groups. In this way, the unfairness tuning parameter ϕ controls representativeness, i.e., how fairly each group is represented in the ranking. Specifically, when $\phi = 1$, then each group is fairly represented. Thus each group's region is equal to the group's proportion of X (fair). As ϕ increases, the fair representation of each group degrades because regions are distorted in such a way that smaller groups have larger regions compared to their proportion of X (unfair). The fairness tuning parameter ϕ is used to create the regions prior to placing any items into ranking τ.

2.1.3 Description of RankTune Data Generation Algorithm. Algorithm 1 displays the pseudocode for the core algorithmic strategy of the RankTune tool. This algorithm operates in three stages: (1) *creating group regions stage*, the assignment of groups to regions in the $[0, 1]$ interval using ϕ; (2) *pseduo-random item placement stage*, the repeated sampling of the $[0, 1]$ interval to place items into the to-be-constructed ranking and (3) *filling-out stage*, once a group has no items remaining, the rest of the items are placed.

Creating Group Regions: In Algorithm 1, lines 1 - 12 create group regions in the to-be-sampled $[0, 1]$ interval. This begins by identifying the smallest group g_{min}, or in the case of multiple such groups, it chooses a random group among those of the smallest size (line 3). Next, ϕ is scaled to a new value ϕ_{scaled} which represents g_{min}'s artificially adjusted new proportion of the item set (line 4). This new proportion can be the same (case of $\phi = 1$, i.e., fair) or larger when $\phi = 0$ resulting in $\phi_{scaled} = 1$ meaning the smallest group is certainly at the top of the ranking. Then in line

5, the original distribution of groups D_x is adjusted to a new group distribution D_{target}. That is, all other groups are proportionally scaled down to accommodate g_{min}'s new proportion of the item set ϕ_{scaled} (line 6). Then, lines 6-12 take the target group distribution and map each group's proportion of X into representing a corresponding region of the $[0, 1]$ space. Each group's proportion of the $[0, 1]$ interval is represented via a lower bound (*lowBound*) and upper bound (*upperBound*). This stage sets the fairness of the placement procedure.

Pseudo-random Item Placement: Lines 13 - 17 construct ranking τ, from top to bottom (i.e. appending), by repeatedly sampling a random number r in the uniform $[0, 1]$ interval. In the prior stage, groups were mapped to regions of the interval. Thus, when r is sampled, we know what group to place (g_{2p}). Here, g_{2p} is the group for which $lowBound[g_{2p}] < r < upperBound[g_{2p}]$.

Filling Out Stage: Once `RankTune` has placed an entire group during the prior stage, lines 18 - 20 in Algorithm 1 fill the remaining positions in τ by placing groups according to increasing group size. `RankTune` then returns the ranking τ.

2.1.4 Assumptions underpinning RankTune. The underlying assumption of `RankTune` is that in order to generate rankings satisfying statistical parity fairness, the likelihood of a group receiving a positive outcome should be equal to that group's proportion of the candidate pool. Then "unfairness" can be added by distorting this proportional relationship between the likelihood of the group receiving the positive outcome and its proportion of the of candidate pool. We view the positive outcome to be the placement of a candidate into the generated ranking and each placement is made by sampling a random number in the uniform $[0, 1]$ interval. In the case of perfect statistical parity, the likelihood of placing a candidate from each group is equal to that group's proportion of all candidates. `RankTune` is driven by a stochastic process (using a random number generator to sample the $[0, 1]$ interval), thus we are unable to formally prove that the approach always generates a fair ranking. However, our empirical assessments and demonstrations in Section 3 and 5 respectively, indicate `RankTune` works well in practice.

2.1.5 Using RankTune. `RankTune` is a module in the `FairRankTune` toolkit. There are two primary functions for interacting with the tool, both shown in Listing 1. The first, `GenFromItems`, is to pass `RankTune` a specific set of items, `item_ids`, and their group identities, `group_ids`, along with `phi`, the unfairness tuning parameter, and `r_cnt`, how many rankings to generate. This can be used when working with a known item set. The second, `GenFromGroups`, does not require actual items to be passed in. Instead, the input is the distribution of groups, represented by array `group_proportions`, and how many items to rank, `num_items`, which saves effort in scenarios where what matters most is how many groups there are and how big they are, as opposed to specific items belonging to specific groups. Further, in Listing 1, `GenFromGroups` is shown generating 3 rankings. This illustrates that it will create different rankings with a very similar fairness degree modeled by `phi = 0.8`. Thus `GenFromGroups` returns an array of size 1000 x 3 called `rankings`; where each of the 3 returned rankings is captured by a column. The second return argument of both `GenFromItems` and `GenFromGroups` is an array representing the group identities of the ordered items in the generated ranking(s).

2.2 FairRankTune Metric Library

The second component of `FairRankTune` is the `Metrics` library. The `Metrics` library provides a unified interface to multiple statistical parity fairness metrics.

2.2.1 `FairRankTune` *metrics and support for their customization.* The fairness literature shows that most fair classification metrics are aggregation metrics that, through a mathematical formula, distill per-group metrics into one

```
import FairRankTune as frt
#Generate from known items
phi = 0.1 # representativeness tuning parameter
r_cnt = 1 #Generate 1 ranking
seed = 10 #For reproducibility
ranking_df, item_group_dict = frt.RankTune.GenFromGroups(item_group_dict, phi, r_cnt, seed)

#Generate from groups
#Generate a biased (phi = 0.1) ranking of 1000 items, with four groups of 100, 200, 300, and 400 items each.
group_proportions = np.asarray([.1, .2, .3, .4]) #Array of group proportions
num_items = 1000 #1000 items to be in the generated ranking
phi = 0.1 # representativeness tuning parameter
r_cnt = 1 #Generate 1 ranking
seed = 10 #For reproducibility
ranking_df, item_group_dict = frt.RankTune.GenFromGroups(group_proportions,
  num_items,phi, r_cnt, seed)

```

Listing 1. Usage example of RankTune data generation.

Combo Variable	Formula
MinMaxRatio	$\min_g V / \max_g V$
MaxMinRatio	$\max_g V / \min_g V$
MaxMinDiff	$\max_g V - \min_g V$
MaxAbsDiff	$\max_g \lvert V - V_{mean} \rvert$
MeanAbsDev	$\frac{1}{G} \sum_g \lvert V - V_{mean} \rvert$
LTwo	$\lVert V \rVert_2^2$
Variance	$\frac{1}{G-1} \sum_g (V_g - V_{mean})^2$

Table 1. Formulas supported for combining per-group style metrics. $V = [V_1, ..., V_g]$ is an array of per-group metrics and G is the number of groups.

```
import FairRankTune as frt
ranking_df = pd.DataFrame(["Joe", "Jack", "Nick", "David", "Mark", "Josh", "Dave",
                           "Bella", "Heidi", "Amy"])
item_group_dict = dict(Joe= "M",  David= "M", Bella= "W", Heidi= "W", Amy = "W", Mark= "M",
                           Josh= "M", Dave= "M", Jack= "M", Nick= "M")

#Calculate EXP
EXPMaxMinDiff, exps_MaxMinDiff = frt.Metrics.EXP(ranking_df, item_group_dict, 'MaxMinDiff')
print("EXP (MaxMinDiff): ", EXPMaxMinDiff, "avg_exposures: ", exps_MaxMinDiff)

EXPMinMaxRatio, exps_MinMaxRatio = frt.Metrics.EXP(ranking_df, item_group_dict, 'MinMaxRatio')
print("EXP (MinMaxRatio): ", EXPMinMaxRatio, "avg_exposures: ", exps_MinMaxRatio)
```

Listing 2. Usage example of the Metrics library to calculate exposure two different ways on a RankTune generated ranking.

value [39]. This single value is typically reported as the fairness metric itself. For instance, in fair classification, we

can measure the true positive rate (TPR) for each group, and combine them into a single value by taking the min-max ratio of the TPRs. Thus, the per-group metric is TPR, which can be combined in any number of ways. We observe that the same conceptual idea is present in contemporary fair ranking metrics and specifically design `FairRankTune` with this additional modularity. To the best of our knowledge, comparing strategies for combining fair ranking per-group metrics into a single value has yet to be studied in the literature. `FairRankTune` simplifies the use of different such combination strategies. Table 1 displays the `FairRankTune` strategies for combining group-specific base metrics into a single value fair ranking metric.

Below, we briefly describe the fairness measures supported in `FairRankTune`. Listing 2 shows their example usage. Note that we intentionally focus on metrics that support multiple groups. We characterize measures in terms of their per-group metric and all but NDKL decompose into per-group metrics:

- **Exposure (EXP)** [19, 55]: the per-group metric is the group average exposure, whereby the exposure of item x_i in ranking τ is $exposure(\tau, x_i) = 1/log_2(\tau(x_i) + 1))$ and the average exposure for group g_j is $avgexp(\tau, g_j) = \sum_{\forall x \in g_j} exposure(\tau, x_i)/|g_j|$.
- **Attention Weighted Rank Fairness (AWRF)** [53]: the per-group metric is the group average attention, whereby the attention score for item x_i in ranking τ as $attention(\tau, x_i) = 100 \times (1-d)^{(\tau(x_i)-1)} \times d$, where d is a parameter representing the proportion of attention received by the first (top) ranked item.
- **Attribute Rank Parity (ARP)** [13]: The per-group metric is the average mixed pairs won by each group, calculated as $avgpairs(\tau, g_i) = \#\ mixedpairswon(g_i)/\#totalmixedpairs(g_i)$. Mixed pairs are pairs that contain items from two different groups.
- **Normalized-Discounted KL-Divergence (NDKL)** [25]: of ranking τ with respect to groups G is defined as:

$$NDKL(\tau, G) = \frac{1}{Z} \sum_{i=1}^{|X|} \frac{1}{log_2(i+1)} d_{KL}(D_{\tau_i} || D_X) \tag{1}$$

  where $d_{KL}(D_{\tau_i} || D_X)$ is the KL-divergence score of the group proportions of the first i positions in τ and the group proportions of the item set X and $Z = \sum_{i=1}^{|\tau|} \frac{1}{log_2(i+1)}$.
- **Exposure Utility (EXPU)** [55]: the per-group metric is the ratio of group average exposure and group average utility, whereby group average exposure is measured exactly as in EXP. Group average utility for group g_j is $avgutil(\tau, g_j) = \sum_{\forall x \in g_j} x_i^{util_\tau}/|g_j|$, where $x_i^{util_\tau}$ is the utility (or relevance score) for candidate x_i in ranking τ.
- **Exposure Realized Utility (EXPRU)** [55][3]: the per-group metric is the ratio of group average click-through rate and group average utility, whereby group average utility is measured exactly as in EXPU. The average click-through rate for group g_j is $avgctr(\tau, g_j) = \sum_{\forall x \in g_j} x_i^{ctr_\tau}/|g_j|$, where $x_i^{ctr_\tau}$ is the click-through rate for candidate x_i in ranking τ.
- **Exposure Rank Biased Precision Equality (ERBE)** [30][4]: the per-group metric is the group exposure, whereby the exposure of item x_i in ranking τ is $exposureRBP(\tau, x_i) = \gamma^{(1-\tau(x_i))}$ and the exposure for group g_j is $expRBP(\tau, g_j) = (1-\gamma) \sum_{\forall x \in g_j} exposureRBP(\tau, x_i)$. In the case of ERBE (and subsequent ERBP and ERBR metrics), Kirnap et al. [30] define exposure differently than Singh and Joachims [55]. Specifically, exposure is

[3]Singh and Joachims [55] used the names "disparate treatment" and "disparate impact" for EXPU and EXPRU, respectively. As pointed out by Raj and Ekstrand [50] this terminology, is inconsistent with the use of these terms in the broader algorithmic fairness literature, thus we have adopted the "exposure utility" and "exposure realized utility" terms introduced in [50].

[4]Kirnap et al. [30] use the terms "equality", "proportionality", and "proportionality to relevance" for ERBE, ERBP, ERBR respectively. To differentiate these exposure-based metrics from the similarly named ones in Singh and Joachims [55] we use the term "exposure rank biased precision" (ERB, for short) to highlight the Kirnap et al. [30] approach of combining exposure and the Rank Biased Precision metric.

inspired by the Rank Biased Precision (RBP) metric [44] and is calculated using the discount factor, γ, a decay parameter controlling the importance of higher ranks.

- **Exposure Rank Biased Precision Proportionality (ERBP)** [30]: the per-group metric is the group average exposure, whereby exposure is measured exactly as in ERBE. Group average exposure for group g_j is $avgexpRBP(\tau, g_j) = (1-\gamma)\sum_{\forall x \in g_j} exposureRBP(\tau, x_i)/|g_j|$.
- **Exposure Rank Biased Precision Proportionality to Relevance (ERBR)** [30]: the per-group metric is the ratio of group exposure and the number of items belonging to the given group that are relevant, whereby exposure is measured exactly as in ERBE. This ratio for group g_j is $expRBP2rel(\tau, g_j) = (1-\gamma)\sum_{\forall x \in g_j} exposureRBP(\tau, x_i)/|g_j^{rel}|$, where $|g_j^{rel}|$ is the count of relevant items in group g_j.

Moreover, our `Metrics` library also facilitates assessing individual fairness in rankings. Individual fairness asks that similiar items be treated similarly [20]. In the case of rankings, Biega et al. [6] provide the preeminent metric for measuring individual fairness:

- **Inequity of Amortized Attention (IAA)** [6]: measures the difference, via the L_1 norm between the cumulative attention and cumulative relevance of items in the ranking(s). Whereby the attention of an item x_i in ranking τ is $attention(\tau, x_i) = 1/log_2(\tau(x_i)+1))$ and the relevance of an item is a $[0-1]$-normalized score.

In subsequent sections, we employ all measures of statistical parity with specific formulations. For brevity, they are referenced by their abbreviation. Table 2 describes the specific formulation we use in this paper.

Abbreviation	Formulation	Range	More Fair
ER	Min-max ratio of avg. group exposures as in [69]	$(0, 1]$	1
EE-D	L_2 norm of avg. group exposure as in [19]	[0,∞)	0
AWRF	Min-max ratio avg. group attention as in [27]	$(0, 1]$	1
NDKL	Proportional group representation across rank prefixes as in [25]	[0,∞)	0
ARP	Max-min difference avg. mixed pairs as in [13]	[0,1]	0

Table 2. Overview of fair ranking metrics used in this work.

2.3 Rankers Module: Implementations of a Variety of Fair Ranking Algorithms

While the focus of `FairRankTune` is the `RankTune` (fairness-aware) data generation method and the accompanying `Metrics` library, we have also implemented and added to our toolkit an initial set of **fair ranking algorithms**. Specifically, the `Rankers` module allows users to use fair ranking methods all in the `FairRankTune` ecosystem. This allows our target audience of researchers, data scientists, and educators to work with state-of-the-art methods. For instance, these algorithms can be used as experimental baselines, in hands-on learning lessons, or as real-world fairness interventions. The `Rankers` module currently provides the DetContSort [25] and Epsilon-Greedy [23] algorithms, with others easily added in the future.

DetConstSort takes as input a ranking and re-ranks it to satisfy a given fair representation criteria. Epsilon-Greedy takes as input a ranking and repeatedly swaps pairs of items so that each item has probability ϵ of swapping with a random item below it. It does not require a specific notion of fairness or prior knowledge of group distributions. As neither algorithm has a publicly available implementation accompanying its introduction [23, 25], our implementation makes it easy for our target audience to use DetConstSort or Epsilon-Greedy in their work. Our documentation provides their BibTex references. Thus, users of `FairRankTune` can generate data and use a suite of fairness metrics, while also

Metric	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}
ER	0.8403	0.8408	0.8422	0.8417	0.8419	0.8404	0.8406	0.8411	0.8413	0.0014
AWRF	0.6255	0.6268	0.6271	0.6263	0.6267	0.6269	0.6256	0.6268	0.6271	0.9978
EE-D	0.1127	0.1127	0.1127	0.1127	0.1127	0.1127	0.1127	0.1127	0.1127	0.1111
NDKL	0.3673	0.3608	0.3596	0.3629	0.3576	0.3633	0.3662	0.3603	0.3588	0.0073
ARP	0.9563	0.9531	0.9519	0.9542	0.9533	0.9527	0.9560	0.9529	0.9519	0.9966

Table 3. Preference Profile of rankings, $\tau_1 - \tau_{10}$, using the *Law Students* dataset. Rankings are generated with RankTune and $phi = 0.9$ for modelling biased voters.

applying a bias mitigation method. We invite the community to contribute additional algorithms - and we intend to expand the offering of algorithms in the toolkit ourselves as well.

2.4 Demonstrating FairRankTune Toolkit Use and Utility

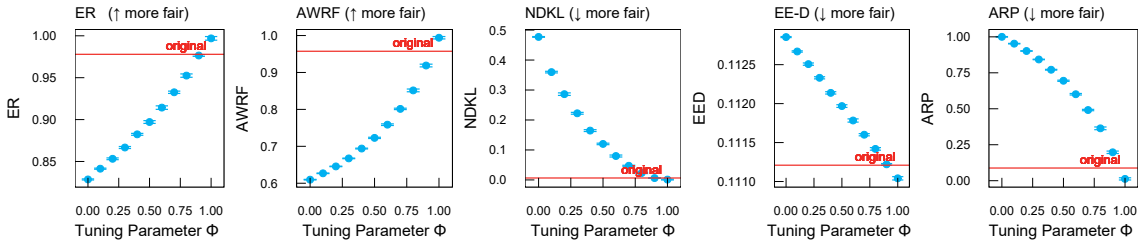


Fig. 1. Average metric values (with 95% confidence intervals) from RankTune constructed rankings using *Law Students* dataset. 10 rankings are generated per ϕ value. As ϕ increases, RankTune outputs increasingly unfairer alternate rankings - particularly compared to the original ranking. The red line "original" is the ranking induced by using the *LSAT score* feature of the dataset.

We now illustrate how pieces of the `FairRankTune` toolkit come together to create custom datasets and assess rankings. The `RankTune` generator can be used with group distributions or specific items, and here we utilize a known set of items. Specifically, we leverage the *Law Students* [62] dataset, which has been used for fair ranking experiments when only a *single* ranking is needed [69]. Yet, for many ranking tasks, as in multi-winner voting (MWV), more than one ranking is required. In multi-winner voting, many voters provide rankings of the same set of candidate items, and a subset of the most popular ones are chosen [11]. The rankings provided by voters are known as a preference profile. `RankTune` can easily create custom preference profiles from an existing dataset, such as *Law Students*, or from group distributions.

Popular approaches to generating synthetic data for voting scenarios include probabilistic rank generation methods such as the Mallows [40] and Plackett-Luce [37] models. However, they tend not to focus on the fairness properties of the rankings (with respect to ranked candidates) that they generate. In both models, voter rankings are generated by sampling from the distribution induced by the model. Specifically, the Mallows Model is an exponential location-spread model, in which the location is a central ranking among a set of voter rankings, and the spread parameter controls the amount of agreement between the central ranking and the sample ranking(s). In this way, the Mallows model allows for generating voting scenarios, where the level of agreement amongst voters can be dialed up or down.

In a similar vein, the Plackett-Luce model allows for generating voter rankings, by sampling a distribution over rankings parameterized by the set of scores associated with each item. When sampling the Plackett-Luce model (i.e.,

generating a ranking) the likelihood of an item being top-ranked is proportional to its score. By adjusting the scores of items, the distribution of to-be-sampled rankings changes.

In contrast, RankTune does not sample from a distribution of rankings. Instead, it uses a repeat insertion model of iteratively deciding when to place items in order to control the resulting fairness of the generated ranking. Unlike in the Mallows and Plackett-Luce models, using RankTune to create voter rankings allows the end-user to control the level of fairness associated with each voter. For instance, RankTune facilitates creating a preference profile where the majority of voters are unfair, and the minority are fair.

Figure 1 illustrates RankTune applied to creating a voter preference profile from the *Law Students* dataset. For all metrics, we plot the "original" ranking produced in the dataset. This is the ranking from the *lsat score* feature, the male and female categories. Then for each ϕ value, we generate ten rankings and plot the average metrics value with 95% confidence intervals. We see RankTune effectively and robustly creates rankings that range from fair (i.e., $\phi = 0$) to completely unfair (i.e., $\phi = 1$). The "original" ranking is not entirely unfair, but closer to a fair ranking than to the rankings RankTune creates.

Table 3 shows the fairness metric values for the ten constructed rankings for $\phi = 0.9$. Regarding MWV, these ten rankings constitute a "biased" preference profile, where bias is of similar strength. RankTune easily generates custom profiles, where some voters are unfair (high ϕ values) and some voters are fairer (low ϕ values). Experimentation on such profiles can yield interesting insights regarding the behavior of voting rules when voters provide biased or unbiased rankings. Thus, RankTune is not only useful in controlled synthetic environments, but also in augmenting datasets for contexts in which they previously were not applicable.

3 EMPIRICAL COMPARISON WITH AVAILABLE DATA GENERATION STRATEGIES

In this section, we highlight the differences and the innovative aspects of FairRankTune's RankTune method compared to alternate data generation strategies.

3.1 Compared Methods and Inputs for Constructing Ranked Data

We now study the capabilities of FairRankTune's RankTune using a variety of group memberships. Table 4 depicts the nine diverse group distributions we work with in this comparative analysis and in our case study in Section 4.

Name	**Distribution** - D_X
Dist A	$(0.2, 0.3, 0.5)$
Dist B	$(0.1, 0.3, 0.6)$
Dist C	$(0.2, 0.3, 0.1, 0.05, 0.03)$
Dist D	$(0.2, 0.2, 0.2, 0.2, 0.2)$
Dist E	$(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.05, 0.05)$
Dist F	$(0.6, 0.08, 0.02, 0.15, 0.1, 0.05)$
Dist G	$(0.3, 0.7)$
Dist H	$(0.5, 0.5)$
Dist I	$(0.1, 0.9)$

Table 4. Group distributions employed in our empirical analysis and case study. Unless otherwise specified, we use $X = 1,000$ items. For example, *Dist A* has three groups: the first has 200 members, the second has 300 members, and the third has 500 members.

To the best of our knowledge, no prior methodology is designed for multi-group fairness-tunable ranked data generation. Therefore, we contrast FairRankTune's RankTune with the following:

- URG (Unfair Ranking Generator) [67] is a ranking generator specifically designed for two group settings. To use URG, the user provides which of the two groups is designated as "protected" and a parameter f, called fairness probability. At every iteration, URG samples a random number $v \in [0, 1]$ if $v < f$ it places a protected group member; otherwise, it places a member of the non-protected group. As visually depicted in Figure 2b, URG generates fair rankings when f equals the protected group's proportion of item set X.
- DetConstSort [25] is a fair ranking algorithm that swaps candidates in a given ranking to create a new fair ranking such that the proportion of each group in every k-sized prefix of the new ranking matches a provided distribution p. In Geyik et al. [25], each value of p is set to *the proportion of the group in the total item set* to assure the ranking satisfies statistical parity [25]. To study the capability of DetConstSort to create unfair rankings, we adjust p so that the smallest group has value 0.9 (thus over-advantaged compared to its size), and the remainder is split amongst groups. For instance, to create an unfair ranking for *Dist A* with $D_X = (0.2, 0.3, 0.5)$ we set $p = (0.9, 0.05, 0.05)$. Using DetConstSort for creating unfairer rankings compared to the given one was not part of its described capabilities in [25]; thus, knowing what values to use for p is in itself a challenge.
- RS-Norm samples a normal distribution to assign a score to each item and then sorts by decreasing scores. This falls under the random scoring and sorting data generation approach adopted in experiments in [2, 10, 26, 45].
- RS-Uni is also a random scoring and sorting data generation approach used in [1, 27, 45, 68]. In contrast to the above version, it samples a uniform distribution to assign item scores and then again sorts.

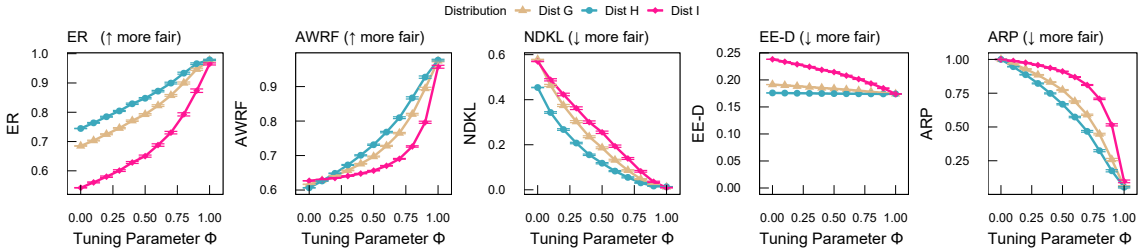
3.2 How does FairRankTune Compare to the State-of-art Binary Group Unfair Ranking Generator?

By supporting multiple groups, RankTune provides increased functionality compared to URG. In addition, as we illustrate below, its fairness-tuning mechanism is easy to use and its resulting effect on the data is simple to intuit. Given URG only supports binary groups, Figure 2 compares RankTune and URG on the binary group distributions (*Dist G - I* indicated in Table 4). Figure 2a illustrates the results of running RankTune with its fairness parameter ϕ adjusted from 0 (unfair) to 1 (more fair), while Figure 2b shows URG when its tuning parameter f is adjusted from 0 to 1.

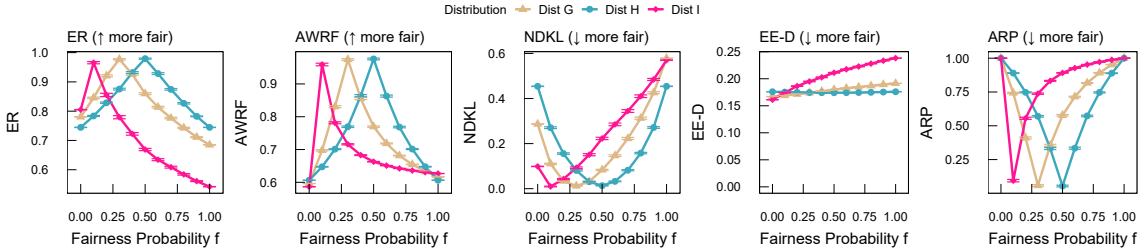
The critical difference between RankTune and URG is that the fairness tuning parameter in RankTune (ϕ) tunes between 0 (unfair) to 1 (more fair), meaning that for any distribution it produces the fairest rankings when $\phi = 1$. While the fairness tuning parameter f (called fairness probability in [67]) in URG tunes between 0 ("protected group" is totally at the top of ranking) and 1 ("protected group" is totally at bottom of the ranking). Hence, URG produces its fairest ranking when f equals the protected group's proportion of item set X. This explains why in Figure 2b, unlike in Figure 2a, each distribution has the fairest result at a different (fairness probability value f. For instance, URG produces the fairest ranking for *Dist I* when $f = 0.1$ and for *Dist H* when $f = 0.5$. In contrast, RankTune behaves in the same consistent way for every group distribution. This makes comparing different item sets easy-to-interpret, since the x-axis goes from fair to unfair. Moreover, the effect of ϕ is predictable; in that, an increase in ϕ increases unfairness.

3.3 How does FairRankTune Compare to Alternate Tools in Multi-Group Settings?

Next, we assess the data generation capabilities of FairRankTune's RankTune and the alternate approaches detailed in Section 3.1) for multi-group settings. Here, we focus on *unfair* ranking generation, as unfair rankings are the commonly accepted test-bed for bias mitigation technologies [47, 68]. Further, the lack of a mechanism to control fairness smoothly from fair to unfair (or vice-versa) in DetConstSort, RS-Norm, and RS-Uni makes this a level comparison. Table 5 displays average fairness metric values, along with a count of how many unique rankings were generated.



(a) RankTune rankings as ϕ is increased from 0 (unfair) to 1 (more fair).



(b) URG [67] rankings as the fairness probability parameter is increased from $0 - 1$.

Fig. 2. Average metric values (with 95% confidence intervals) from 200 RankTune and URG [67] constructed rankings with the binary group distributions in Table 4 for 1,000 items. RankTune, across distributions, is always most fair at $\phi = 1$ and shows smooth increases in unfairness, i.e. plotted lines trend up or down depending on the metric. In contrast, for URG [67], the fairest result is when the "protected group" proportion equals the fairness probability thus we get V-shaped lines where each distribution is fairest at a different value, i.e. the tuning parameter does not tune between fair and unfair, but rather unfair- to fair - to unfair.

Metric	**Method**	Dist A	Dist B	Dist C	Dist D	Dist E	Dist F	Dist G	Dist H	Dist I
	RankTune	0.62 ± 0.01	0.53 ± 0.01	0.42 ± 0.01	0.60 ± 0.01	0.40 ± 0.01	0.41 ± 0.01	0.70 ± 0.01	0.76 ± 0.01	0.56 ± 0.01
ER	DetConstSort	0.64 ± 0.00	0.54 ± 0.00	0.44 ± 0.00	0.64 ± 0.00	0.44 ± 0.00	0.68 ± 0.00	0.71 ± 0.00	0.78 ± 0.00	0.56 ± 0.00
(fair: 1)	RS-Norm	0.96 ± 0.02	0.96 ± 0.02	0.93 ± 0.03	0.94 ± 0.02	0.88 ± 0.03	0.91 ± 0.03	0.98 ± 0.02	0.98 ± 0.01	0.97 ± 0.02
	RS-Uni	0.96 ± 0.02	0.96 ± 0.02	0.93 ± 0.03	0.94 ± 0.02	0.88 ± 0.03	0.91 ± 0.03	0.98 ± 0.01	0.98 ± 0.01	0.97 ± 0.02
	RankTune	0.54 ± 0.00	0.54 ± 0.00	0.45 ± 0.00	0.46 ± 0.00	0.40 ± 0.00	0.44 ± 0.00	0.63 ± 0.00	0.63 ± 0.00	0.63 ± 0.00
AWRF	DetConstSort	0.60 ± 0.00	0.60 ± 0.00	0.59 ± 0.00	0.62 ± 0.00	0.60 ± 0.00	0.57 ± 0.00	0.63 ± 0.00	0.65 ± 0.00	0.63 ± 0.00
(fair: 1)	RS-Norm	0.97 ± 0.01	0.97 ± 0.02	0.94 ± 0.02	0.95 ± 0.02	0.90 ± 0.02	0.93 ± 0.03	0.98 ± 0.01	0.98 ± 0.01	0.98 ± 0.02
	RS-Uni	0.97 ± 0.01	0.97 ± 0.02	0.94 ± 0.02	0.95 ± 0.02	0.90 ± 0.02	0.93 ± 0.03	0.98 ± 0.01	0.98 ± 0.01	0.98 ± 0.02
	RankTune	0.23 ± 0.00	0.26 ± 0.00	0.35 ± 0.00	0.28 ± 0.00	0.45 ± 0.00	0.38 ± 0.00	0.19 ± 0.00	0.18 ± 0.00	0.23 ± 0.00
EED	DetConstSort	0.23 ± 0.00	0.26 ± 0.00	0.35 ± 0.00	0.28 ± 0.00	0.45 ± 0.00	0.29 ± 0.00	0.19 ± 0.00	0.18 ± 0.00	0.23 ± 0.00
(fair: 0)	RS-Norm	0.21 ± 0.00	0.21 ± 0.00	0.28 ± 0.00	0.28 ± 0.00	0.41 ± 0.00	0.30 ± 0.00	0.17 ± 0.00	0.17 ± 0.00	0.17 ± 0.00
	RS-Uni	0.21 ± 0.00	0.21 ± 0.00	0.28 ± 0.00	0.28 ± 0.00	0.41 ± 0.00	0.30 ± 0.00	0.17 ± 0.00	0.17 ± 0.00	0.17 ± 0.00
	RankTune	0.63 ± 0.04	0.71 ± 0.04	0.90 ± 0.04	0.70 ± 0.04	1.01 ± 0.04	0.93 ± 0.04	0.47 ± 0.03	0.35 ± 0.02	0.49 ± 0.03
NDKL	DetConstSort	0.51 ± 0.00	0.53 ± 0.00	0.51 ± 0.00	0.52 ± 0.00	0.50 ± 0.00	0.50 ± 0.00	0.43 ± 0.00	0.28 ± 0.00	0.49 ± 0.00
(fair: 0)	RS-Norm	0.02 ± 0.01	0.02 ± 0.01	0.04 ± 0.01	0.04 ± 0.01	0.08 ± 0.01	0.04 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	0.01 ± 0.01
	RS-Uni	0.02 ± 0.01	0.02 ± 0.01	0.04 ± 0.01	0.04 ± 0.01	0.08 ± 0.01	0.04 ± 0.01	0.01 ± 0.01	0.01 ± 0.01	0.01 ± 0.01
	RankTune	0.97 ± 0.01	0.99 ± 0.01	0.99 ± 0.00	0.98 ± 0.01	1.00 ± 0.00	0.99 ± 0.00	0.97 ± 0.01	0.95 ± 0.01	0.99 ± 0.00
ARP	DetConstSort	0.77 ± 0.00	0.77 ± 0.00	0.62 ± 0.00	0.63 ± 0.00	0.57 ± 0.00	0.82 ± 0.00	0.95 ± 0.00	0.89 ± 0.00	0.99 ± 0.00
(fair: 0)	RS-Norm	0.04 ± 0.02	0.05 ± 0.02	0.07 ± 0.03	0.06 ± 0.02	0.11 ± 0.03	0.08 ± 0.03	0.03 ± 0.02	0.03 ± 0.02	0.05 ± 0.03
	RS-Uni	0.04 ± 0.02	0.05 ± 0.02	0.07 ± 0.03	0.06 ± 0.02	0.11 ± 0.03	0.08 ± 0.03	0.03 ± 0.02	0.03 ± 0.02	0.05 ± 0.03
	RankTune	200	200	199	200	200	200	200	200	200
Unique	DetConstSort	1	1	1	1	1	1	1	1	1
rankings	RS-Norm	200	200	200	200	200	200	200	200	200
	RS-Uni	200	200	200	200	200	200	200	200	200

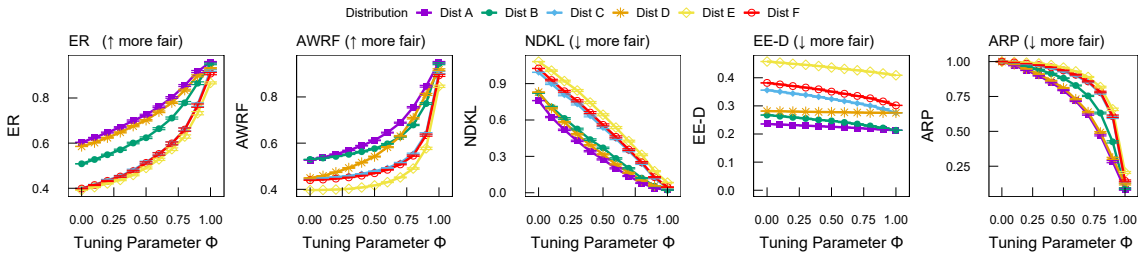
Table 5. Average metric values (with standard deviation) and unique ranking count from 200 trials for each distribution from Table 4.

We observe that `DetConstSort` produces only a single unfair ranking for each distribution. That is, even over 200 distinct trials, it *always produces the exact same ranking*. This is not desirable for data generation, since for a given

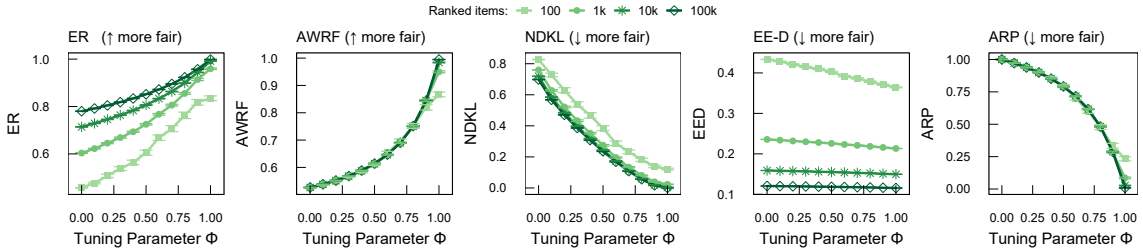
fairness level only one ranking is created. We set `DetConstSort` parameter p as described in Section 3.1; however, there is little intuition on how we should adjust the values in p, per distribution, to increase or decrease the fairness of the produced ranking. In contrast, we see that `RankTune`, `RS-Norm`, and `RS-Uni` all produce a variety of rankings as seen by the unique ranking count. We also observe that `RS-Norm`, and `RS-Uni` *do not generate unfair rankings*, i.e., they consistently result in relatively fair rankings. This can be seen in *Dist A* for ARP (ARP is fair at 0) where `RS-Norm`, and `RS-Uni` have average ARP scores of 0.04 compared to `DetConstSort`'s more unfair score of 0.77 and `RankTune` score of 0.97. Moreover, the low standard deviation of the metric values illustrate that the relative fairness of `RS-Norm`and `RS-Uni` is a frequent occurrence. This finding is in line with the use of randomization as a fair ranking strategy [19, 63]. Thus, to using these strategies to create unfair rankings may require trial and error since there is no way to control the fairness of the resulting ranking.

In summary, across diverse group distributions, `RankTune` is easier to tune and more effective than `DetConstSort`, `RS-Norm`, and `RS-Uni`. It provides an easy-to-use control mechanism for tuning fairness, can reliably generate a vast amount of rankings at each point along the fairness spectrum, and its behavior is consistent across different items sets.

4 CASE STUDY: HOW DO FAIRNESS METRICS COMPARE IN MULTI-GROUP SETTINGS?



(a) RankTune constructed rankings with the multiple group (> 2) distributions in Table 4 for $1,000$ items.



(b) RankTune constructed rankings with *Dist A* from Table 4 for varying numbers of ranked items.

Fig. 3. Case Study: Average metric values (with 95% confidence intervals) from RankTune constructed rankings.ER and AWRF are more fair at 1 (thus upward slopes), and NDKL, EE-D, and ARP are more fair at 0 (thus downward slopes).

A primary objective of the proposed `FairRankTune` toolkit is to aid the empirical evaluation of bias-mitigating interventions and fairness metrics. While existing comparisons of fair ranking metrics for statistical parity have been limited to binary groups [32, 50], we now use our toolkit to study fairness metrics beyond the two-group assumption comparatively. In Figure 3a, we plot all fairness metrics for rankings generated for all multi-group distributions (colored lines) from Table 4. Then, we fix a particular group distribution (*Dist A*) and observe the differences in fairness metrics

on ranking generated by RankTune when the number of ranked items increases. Figure 3b displays *Dist A*. From this empirical comparison, we derive two takeaways:

(1.) *Directly comparing values in a single metric across datasets can be misleading.* We see that metrics vary in their ranges and sensitivity to different group distributions, as seen by how different metrics relatively order *Dist A - Dist F* when $\phi = 1$ or other ϕ values in Figure 3a. All metrics report their most unfair values for *Dist E*, which contains the most groups (11), followed by *Dist F*, which includes the second largest number of groups. For a metric such as EE-D, a given value, for instance, EE-D = x' will represent a more considerable disparity (unfairness) when there are fewer groups, and the same value EE-D= x' represents a smaller disparity when there are more groups. Put another way, depending on the data, EE-D = x' could be fair or unfair. This means comparing two rankings of different data by metric value alone can be flawed as it does not reveal which is more biased. Fairness is relative to the data; thus, metrics values should be interpreted contextually.

(2.) *For a fixed group distribution, some metrics are more sensitive to the data size.* As seen by how different metrics relatively order *Dist A - F* when $\phi = 1$ or other ϕ values in Figure 3b, not all metrics vary in their sensitivity to the number of items being ranked. While EE-D and ER show sensitivity to the number of ranked items, as seen by how each colored line is primarily distinct from one another, AWRF, NDKL, and ARP are less affected by the number of items, as seen by how the colored lines are almost indistinguishable. When comparing datasets with identical group distributions, metrics such as NDKL or ARP provide level comparisons.

5 USE CASES: FROM METRIC DESIGN AND SYSTEM DEVELOPMENT TO FACILITATING LEARNING

Next, we discuss several use cases pertaining to the Fairness and Ethical AI community for our proposed toolkit.

Study and Design of Metrics and Algorithms: Researchers can leverage FairRankTune to test novel fair ranking strategies on data generated with RankTune. Additionally, the fairness metrics included in FairRankTune provide a simple and efficient way to analyze the bias mitigation of newly proposed ranking strategies. In the case of designing metrics, it simplifies the comparison of a new metric to the five existing metrics included in FairRankTune. The FairRankTune metric library streamlines utilizing multiple metrics since all metrics take the same input, making it easy to quickly run a variety of metrics for any number of rankings. Moreover, by supporting seven different formulations of fairness metrics, e.g., combining per-group exposure via different formulas, FairRankTune facilitates analyzing the formulation of popular fairness metrics in terms of statistical bias and conducting user studies examining which formulations are more interpretable to practitioners.

Designing Fairness-focused Visual Analytics Systems: While algorithm and metric design for ranking tasks have been at the forefront of the fairness community, an open area of research is the development of visual interfaces for diagnosing and explaining fairness issues [54, 65]. Not only can FairRankTune be directly leveraged for metric calculation in a visual interface, but it also provides opportunities to create diverse data for testing and aiding the design of such systems. Researchers and engineers can easily use the RankTune method to develop specific data for their design processes. In particular, the ability of RankTune to create biased rankings of any number of groups and items facilitates answering critical open questions such as how to visualize and communicate fairness issues when there are many groups.

Ready-to-use Learning module for Educators and Students: As AI ethics is increasingly integrated into computer science and STEM curriculums, educators from high school to university levels require tools to facilitate hands-on learning. Our toolkit aids instructors by providing a freestanding learning module. Students can generate rankings with RankTune, utilize the provided fair re-ranking algorithm(s) to mitigate bias and explore quantifying unfairness

with the measures provided in the metric library. By promoting an active learning experience, FairRankTune facilitates firsthand experiential learning.

6 LIMITATIONS AND FUTURE WORK

This work presents an open-sourced toolkit for fair ranking tasks, including metric and re-ranking algorithm implementations and a new fairness-tunable ranked data generation method. While we have empirically verified that our RankTune data generation method is able to tune the degree of statistical parity fairness in the rankings it generates, it is ultimately a heuristic approach. As any resource requires refinement, we are open and eager to receive requests and ideas for adding to the toolkit. Avenues for enriching FairRankTune include: consider how we might approach the task of generating ranked data embodying score-based fairness issues. Further, FairRankTune does not inherently address multi-sided fairness concerns [12, 57, 58]. RankTune models unfairness for ranked items, i.e., producers, and does not simulate unfairness issues for ranking viewers, i.e., consumers. Future work might consider integrating fairness concerns for consumers [8, 35], or multi-sided (consumers and producers) issues [60, 64] in the FairRankTune toolkit.

7 CONCLUSION

The primary motivation of this work is to ensure a positive broader impact by providing practitioners with an easy-to-use tool FairRankTune. FairRankTune is a dedicated Python library supporting fair ranking analysis and experimentation. It provides a user-friendly interface to many bias-measurement metrics, bias-mitigation techniques, and the first of its kind fairness-tunable ranked data generator RankTune. We compare RankTune with current tools for data generation, highlighting how RankTune provides both greater functionality and ease of use. We conduct a case study using RankTune and the metric library included in FairRankTune to empirically compare statistical parity metrics in multi-group settings, providing takeaways for using and selecting metrics. Moreover, we highlight how our primary audience of students, researchers, data scientists, and educators can use FairRankTune for critical applications and explorations. Currently, disadvantaged groups can benefit as this population deploys fairness interventions.

REFERENCES

[1] Amanda A. Aird, Paresha Farastu, Joshua Sun, Amy Voida, Nicholas Mattei, and Robin D. Burke. 2023. Dynamic fairness-aware recommendation through multi-agent social choice. *ArXiv* abs/2303.00968 (2023).

[2] Nil-Jana Akpinar, Cyrus DiCiccio, Preetam Nandy, and Kinjal Basu. 2022. Long-term Dynamics of Fairness Intervention in Connection Recommender Systems. *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society* (2022).

[3] Nil-Jana Akpinar, Manish Nagireddy, Logan Stapleton, Hao Fei Cheng, Haiyi Zhu, Steven Wu, and Hoda Heidari. 2022. A Sandbox Tool to Bias(Stress)-Test Fairness Algorithms. *ArXiv* abs/2204.10233 (2022).

[4] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilović, et al. 2019. AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development* 63, 4/5 (2019), 4–1.

[5] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H Chi, et al. 2019. Fairness in recommendation ranking through pairwise comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2212–2220.

[6] 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*. 405–414.

[7] Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. 2020. Fairlearn: A toolkit for assessing and improving fairness in AI. *Microsoft, Tech. Rep. MSR-TR-2020-32* (2020).

[8] Ludovico Boratto, Gianni Fenu, Mirko Marras, and Giacomo Medda. 2022. Consumer Fairness in Recommender Systems: Contextualizing Definitions and Mitigations. In *European Conference on Information Retrieval*.

[9] Dimitrios Bountouridis, Jaron Harambam, Mykola Makhortykh, Mónica Marrero, Nava Tintarev, and Claudia Hauff. 2019. SIREN: A Simulation Framework for Understanding the Effects of Recommender Systems in Online News Environments. *Proceedings of the Conference on Fairness, Accountability, and Transparency* (2019).

[10] Amanda Bower, Kristian Lum, Tomo Lazovich, Kyra Yee, and Luca Belli. 2022. Random Isn't Always Fair: Candidate Set Imbalance and Exposure Inequality in Recommender Systems. *ArXiv* abs/2209.05000 (2022).

[11] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. 2016. *Handbook of Computational Social Choice*. Cambridge University Press.

[12] R. Burke, Nasim Sonboli, and Aldo Ordonez-Gauger. 2018. Balanced Neighborhoods for Multi-sided Fairness in Recommendation. In *FAT*.

[13] Kathleen Cachel, Elke Rundensteiner, and Lane Harrison. 2022. MANI-Rank: Multiple Attribute and Intersectional Group Fairness for Consensus Ranking. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE.

[14] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. 2017. Optimized pre-processing for discrimination prevention. *Advances in neural information processing systems* 30 (2017).

[15] L Elisa Celis, Damian Straszak, and Nisheeth K Vishnoi. 2017. Ranking with fairness constraints. *arXiv preprint arXiv:1704.06840* (2017).

[16] Mattia Cerrato, Marius Köppel, Alexander Segner, Roberto Esposito, and Stefan Kramer. 2020. Fair pairwise learning to rank. *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)* (2020), 729–738.

[17] European Commission. 2018. 2018 reform of EU data protection rules. https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes_en.pd

[18] Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. 2017. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining*. 797–806.

[19] Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. 2020. Evaluating Stochastic Rankings with Expected Exposure. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (2020).

[20] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. 2001. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*. 613–622.

[21] Michael D. Ekstrand, Anubrata Das, R. Burke, and Fernando Diaz. 2021. Fairness and Discrimination in Information Access Systems. *ArXiv* abs/2105.05779 (2021).

[22] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 259–268.

[23] Yunhe Feng and C. Shah. 2022. Has CEO Gender Bias Really Been Fixed? Adversarial Attacking and Improving Gender Fairness in Image Search. In *AAAI*.

[24] Yongqing Gao, Guangda Huzhang, Weijie Shen, Yawen Liu, Wen-Ji Zhou, Qing Da, Dan Shen, and Yang Yu. 2021. Imitate TheWorld: A Search Engine Simulation Platform. *ArXiv* abs/2107.07693 (2021).

[25] Sahin Cem Geyik, Stuart Ambler, and Krishnaram Kenthapadi. 2019. Fairness-aware ranking in search & recommendation systems with application to linkedin talent search. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining*. 2221–2231.

[26] Azin Ghazimatin, Matthäus Kleindessner, Chris Russell, Ziawasch Abedjan, and Jacek R. Golebiowski. 2022. Measuring Fairness of Rankings under Noisy Sensitive Information. *2022 ACM Conference on Fairness, Accountability, and Transparency* (2022).

[27] Avijit Ghosh, Ritam Dutt, and Christo Wilson. 2021. When Fair Ranking Meets Uncertain Inference. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2021).

[28] Ananya Gupta, Eric Johnson, Justin Payan, Aditya Kumar Roy, Ari Kobren, Swetasudha Panda, Jean-Baptiste Tristan, and Michael Wick. 2021. Online post-processing in rankings for fair utility maximization. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 454–462.

[29] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems* 29 (2016), 3315–3323.

[30] Ömer Kirnap, Fernando Diaz, Asia J. Biega, Michael D. Ekstrand, Ben Carterette, and Emine Yilmaz. 2021. Estimation of Fair Ranking Metrics with Incomplete Judgments. *Proceedings of the Web Conference 2021* (2021). https://api.semanticscholar.org/CorpusID:235324806

[31] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2016. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807* (2016).

[32] Caitlin Kuhlman, Walter Gerych, and Elke Rundensteiner. 2021. Measuring group advantage: A comparative study of fair ranking metrics. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 674–682.

[33] Caitlin Kuhlman, MaryAnn VanValkenburg, and Elke Rundensteiner. 2019. Fare: Diagnostics for fair ranking using pairwise error metrics. In *The World Wide Web Conference*. 2936–2942.

[34] Michelle Seng Ah Lee and Jat Singh. 2021. The landscape and gaps in open source fairness toolkits. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–13.

[35] Yunqi Li, H. Chen, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2021. User-oriented Fairness in Recommendation. *Proceedings of the Web Conference 2021* (2021).

[36] Yunqi Li, H. Chen, Shuyuan Xu, Yingqiang Ge, Juntao Tan, Shuchang Liu, and Yongfeng Zhang. 2022. Fairness in Recommendation: A Survey. *ArXiv* abs/2205.13619 (2022).

[37] R. Duncan Luce. 1979. Individual Choice Behavior: A Theoretical Analysis.

[38] Eli Lucherini, Matthew Sun, Amy A. Winecoff, and Arvind Narayanan. 2021. T-RECS: A Simulation Tool to Study the Societal Impact of Recommender Systems. *ArXiv* abs/2107.08959 (2021).

[39] Kristian Lum, Yunfeng Zhang, and Amanda Bower. 2022. De-biasing "bias" measurement. *2022 ACM Conference on Fairness, Accountability, and Transparency* (2022).

[40] Colin L Mallows. 1957. Non-null ranking models. *Biometrika* 44, 1/2 (1957), 114–130.

[41] James McInerney, Ehtsham Elahi, Justin D. Basilico, Yves Raimond, and Tony Jebara. 2021. Accordion: A Trainable Simulator for Long-Term Interactive Systems. *Proceedings of the 15th ACM Conference on Recommender Systems* (2021).

[42] Omid Memarrast, Ashkan Rezaei, Rizal Fathony, and Brian D. Ziebart. 2021. Fairness for Robust Learning to Rank. *ArXiv* abs/2112.06288 (2021).

[43] Martin Mladenov, Chih-Wei Hsu, Vihan Jain, Eugene Ie, Christopher Colby, Nicolas Mayoraz, Hubert Pham, Dustin Tran, Ivan Vendrov, and Craig Boutilier. 2021. RecSim NG: Toward Principled Uncertainty Modeling for Recommender Ecosystems. *ArXiv* abs/2103.08057 (2021).

[44] Alistair Moffat and Justin Zobel. 2008. Rank-biased precision for measurement of retrieval effectiveness. *ACM Trans. Inf. Syst.* 27 (2008), 2:1–2:27. https://api.semanticscholar.org/CorpusID:18532232

[45] Preetam Nandy, Cyrus DiCiccio, Divya Venugopalan, Heloise Logan, Kinjal Basu, and Noureddine El Karoui. 2020. Achieving Fairness via Post-Processing in Web-Scale Recommender Systems. *2022 ACM Conference on Fairness, Accountability, and Transparency* (2020).

[46] Harikrishna Narasimhan, Andrew Cotter, Maya Gupta, and Serena Wang. 2020. Pairwise fairness for ranking and regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5248–5255.

[47] Gourab K. Patro, Lorenzo Porcaro, Laura Mitchell, Qiuyue Zhang, Meike Zehlike, and Nikhil Garg. 2022. Fair Ranking: A Critical Review, Challenges, and Future Directions. In *2022 ACM Conference on Fairness, Accountability, and Transparency* (Seoul, Republic of Korea) *(FAccT '22)*. Association for Computing Machinery, New York, NY, USA, 1929–1942. https://doi.org/10.1145/3531146.3533238

[48] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini. 2008. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 560–568.

[49] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. 2017. On fairness and calibration. *Advances in neural information processing systems* 30 (2017).

[50] Amifa Raj and Michael D Ekstrand. 2022. Measuring Fairness in Ranked Results: An Analytical and Empirical Comparison. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 726–736.

[51] Pedro Saleiro, Benedict Kuester, Loren Hinkson, Jesse London, Abby Stevens, Ari Anisfeld, Kit T Rodolfa, and Rayid Ghani. 2018. Aequitas: A bias and fairness audit toolkit. *arXiv preprint arXiv:1811.05577* (2018).

[52] Marlesson R. O. Santana, Luckeciano Carvalho Melo, Fernando H. F. Camargo, Bruno Brandão, Anderson Soares, Renan M. Oliveira, and Sandor Caetano. 2020. MARS-Gym: A Gym framework to model, train, and evaluate Recommender Systems for Marketplaces. *2020 International Conference on Data Mining Workshops (ICDMW)* (2020), 189–197.

[53] 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 *Companion Proceedings of The 2019 World Wide Web Conference*. 553–562.

[54] Hilson Shrestha, Kathleen Cachel, Mallak Alkhathlan, Elke Rundensteiner, and Lane Harrison. 2022. FairFuse: Interactive Visual Support for Fair Consensus Ranking. In *2022 IEEE Visualization and Visual Analytics (VIS)*. IEEE, 65–69.

[55] 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*. 2219–2228.

[56] Ashudeep Singh and Thorsten Joachims. 2019. Policy learning for fairness in ranking. *Advances in Neural Information Processing Systems* 32 (2019).

[57] Nasim Sonboli, Robin D. Burke, Michael D. Ekstrand, and Rishabh Mehrotra. 2022. The multisided complexity of fairness in recommender systems. *AI Magazine* (2022).

[58] Nasim Sonboli, Jessie Smith, Florencia Cabral Berenfus, R. Burke, and Casey Fiesler. 2021. Fairness and Transparency in Recommendation: The Users' Perspective. *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization* (2021).

[59] Sriram Vasudevan and Krishnaram Kenthapadi. 2020. LiFT: A Scalable Framework for Measuring Fairness in ML Applications. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2773–2780.

[60] Lequn Wang and Thorsten Joachims. 2020. User Fairness, Item Fairness, and Diversity for Rankings in Two-Sided Markets. *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval* (2020).

[61] Yuan Wang, Zhiqiang Tao, and Yi Fang. 2022. A Meta-learning Approach to Fair Ranking. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2022).

[62] Linda F. Wightman. 1998. LSAC National Longitudinal Bar Passage Study. LSAC Research Report Series.

[63] Haolun Wu, Chen Ma, Bhaskar Mitra, Fernando Diaz, and Xue Liu. 2021. A Multi-objective Optimization Framework for Multi-stakeholder Fairness-aware Recommendation. *ACM Transactions on Information Systems (TOIS)* (2021).

[64] Haolun Wu, Bhaskar Mitra, Chen Ma, Fernando Diaz, and Xue Liu. 2022. Joint Multisided Exposure Fairness for Recommendation. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2022).

[65] Tiankai Xie, Yuxin Ma, Jian Kang, Hanghang Tong, and Ross Maciejewski. 2021. FairRankVis: A Visual Analytics Framework for Exploring Algorithmic Fairness in Graph Mining Models. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 368–377.

[66] Ke Yang, Joshua R Loftus, and Julia Stoyanovich. 2020. Causal intersectionality for fair ranking. *arXiv preprint arXiv:2006.08688* (2020).

[67] Ke Yang and Julia Stoyanovich. 2017. Measuring fairness in ranked outputs. In *Proceedings of the 29th international conference on scientific and statistical database management*. 1–6.

[68] Meike Zehlike, Francesco Bonchi, Carlos Castillo, Sara Hajian, Mohamed Megahed, and Ricardo Baeza-Yates. 2017. Fa* ir: A fair top-k ranking algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1569–1578.

[69] Meike Zehlike and Carlos Castillo. 2020. Reducing disparate exposure in ranking: A learning to rank approach. In *Proceedings of The Web Conference 2020*. 2849–2855.

[70] Meike Zehlike, Tom Sühr, Ricardo Baeza-Yates, Francesco Bonchi, Carlos Castillo, and Sara Hajian. 2022. Fair Top-k Ranking with multiple protected groups. *Information Processing & Management* 59, 1 (2022), 102707.

[71] Meike Zehlike, Tom Sühr, Carlos Castillo, and Ivan Kitanovski. 2019. FairSearch: A Tool For Fairness in Ranked Search Results. *Companion Proceedings of the Web Conference 2020* (2019).

[72] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in Ranking, Part I: Score-based Ranking. *ACM Computing Surveys (CSUR)* (2022).

[73] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in Ranking, Part II: Learning-to-Rank and Recommender Systems. *ACM Computing Surveys (CSUR)* (2022).