FairDiverse: A Comprehensive Toolkit for Fairness- and Diversity-aware Information Retrieval

Chen Xu*
Renmin University of China
Beijing, China
xc_chen@ruc.edu.cn

Zhirui Deng*
Renmin University of China
Beijing, China
zrdeng@ruc.edu.cn

Clara Rus*
University of Amsterdam
Amsterdam, The Netherlands
c.a.rus@uva.nl

Xiaopeng Ye
Renmin University of China
Beijing, China
xpye@ruc.edu.cn

Yuanna Liu
University of Amsterdam
Amsterdam, The Netherlands
y.liu8@uva.nl

Jun Xu†
Renmin University of China
Beijing, China
junxu@ruc.edu.cn

Zhicheng Dou†
Renmin University of China
Beijing, China
dou@ruc.edu.cn

Ji-Rong Wen
Renmin University of China
Beijing, China
jrwen@ruc.edu.cn

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

Abstract

In modern information retrieval (IR), going beyond accuracy is crucial for maintaining a healthy ecosystem, particularly in meeting fairness and diversity requirements. To address these needs, various datasets, algorithms, and evaluation methods have been developed. These algorithms are often tested with different metrics, datasets, and experimental settings, making comparisons inconsistent and challenging. Consequently, there is an urgent need for a comprehensive IR toolkit, enabling standardized assessments of fairness- and diversity-aware algorithms across IR tasks. To address these issues, we introduce an open-source standardized toolkit called **FairDiverse**. First, FairDiverse provides a comprehensive framework for incorporating fairness- and diversity-aware approaches, including pre-processing, in-processing, and post-processing methods, into different pipeline stages of IR. Second, FairDiverse enables the evaluation of **29** fairness, and diversity algorithms across **16** base models for two fundamental IR tasks—search and recommendation—facilitating the establishment of a comprehensive benchmark. Finally, FairDiverse is highly extensible, offering multiple APIs to enable IR researchers to quickly develop their own fairness- and diversity-aware IR models, and allows for fair comparisons with existing baselines. The project is open-sourced on GitHub: https://github.com/XuChen0427/FairDiverse.

CCS Concepts

• **Information systems** → **Information retrieval**.

Keywords

Information retrieval, Fairness, Diversity, Toolkit

ACM Reference Format:
Chen Xu, Zhirui Deng, Clara Rus, Xiaopeng Ye, Yuanna Liu, Jun Xu, Zhicheng Dou, Ji-Rong Wen, and Maarten de Rijke. 2025. FairDiverse: A Comprehensive Toolkit for Fairness- and Diversity-aware Information Retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3726302.3730280

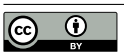
1 Introduction

Information retrieval (IR) tasks, such as search and recommendation, typically aim to select the information that meets user needs [16, 68]. In modern IR, factors beyond the accuracy of information access, such as novelty, diversity, and fairness, are crucial for building a healthy ecosystem [42]. Among these factors, fairness and diversity have gained increasing attention in recent years [22, 43, 54]. Both aim to expose users to a broader range of information sources [23, 54] while also supporting diverse types of providers [48, 61].

To ensure fairness and diversity in IR systems, many fairness-aware [15, 31, 46, 48, 53, 58, 61, 65] and diversity-aware algorithms [43, 50, 54, 71] have been designed as plugins or modules that can be integrated into various stages of the IR pipeline. However, fairness and diversity often suffer from a lack of unified definitions [20, 43]. As a result, the evaluation of these algorithms in IR systems is based on different metrics, datasets, and evaluation settings (details are shown in Section 7). Hence, the performance of these algorithms cannot be compared consistently. Developing a unified, fair, and extensible toolkit for fairness and diversity is critically important and urgently needed to evaluate these algorithms consistently across IR tasks. Such a toolkit framework holds significant value for fostering a trustworthy IR community.

To create a unified and equitable evaluation, we introduce FairDiverse, an open-source standardized toolkit designed to assess

*Equal contributions.
†Jun Xu and Zhicheng Dou are the corresponding authors.



SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730280

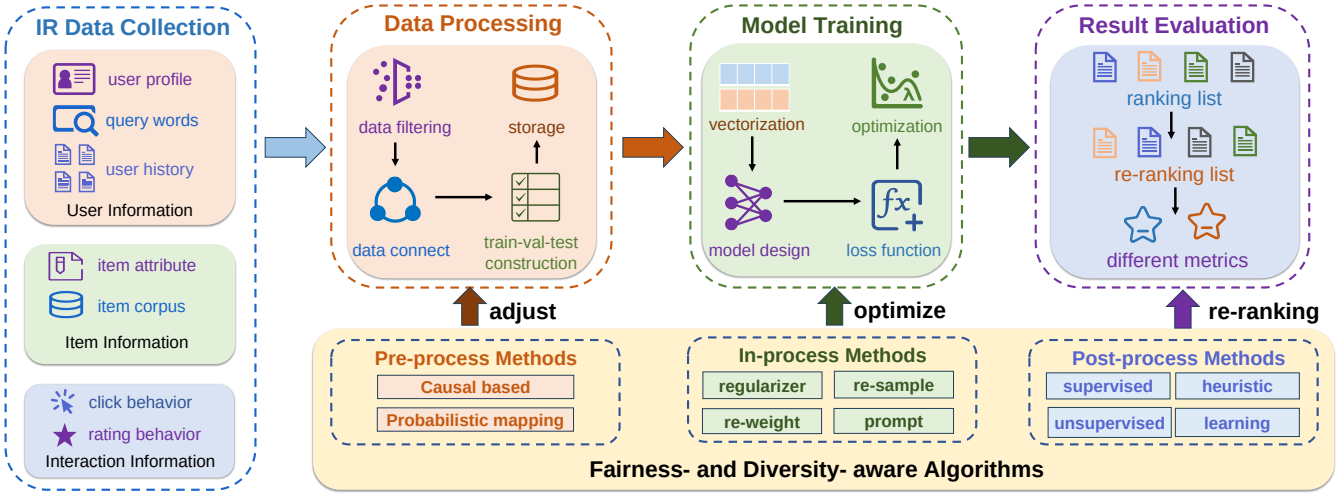


Figure 1: Overall architecture of FairDiverse. We categorize fairness- and diversity-aware algorithms into pre-processing, in-processing, and post-processing stages, corresponding to data processing, model training, and result evaluation phases of IR.

Table 1: Comparison between existing fairness- and diversity-aware toolkits. ✗ denotes that the feature is not supported, while ✓ indicates that the feature is supported.

Features	Recbole [81]	FFB [30]	Fairlearn [8]	AIF360 [5]	Aequitas [33]	**FairDiverse**
Recommendation	✓	✗	✗	✗	✗	✓
Search	✗	✗	✗	✗	✗	✓
Pre-processing	✗	✗	✓	✓	✓	✓
In-processing	✓	✓	✓	✓	✓	✓
Post-processing	✗	✗	✓	✓	✓	✓
Number of models	4	6	6	15	10	**29**

fairness and diversity in IR systems. First, FairDiverse offers detailed guidance on incorporating fairness- and diversity-aware algorithms throughout various stages of the IR process. These algorithms are categorized into pre-processing, in-processing, and post-processing methods, corresponding to data processing, model training, and result evaluation stages in different IR pipeline steps, respectively. Then, FairDiverse implements a wide range of fairness- and diversity-aware models (**29** models) tailored to **16** base models under two fundamental IR tasks: search and recommendation. It offers corresponding implementation code and systematically evaluates these algorithms using over ten accuracy, fairness, and diversity metrics, enabling the construction of a benchmark within FairDiverse.

In the literature, only a few open-source toolkits and libraries have been developed for fairness- and diversity-aware IR algorithms. Table 1 provides a comparison between these existing resources and the proposed FairDiverse, highlighting features such as supported IR tasks (recommendation and search), algorithm types (pre-processing, in-processing, and post-processing), and the number of implemented models. Other toolkit details are provided in Section 7. As shown in Table 1, FairDiverse provides the largest number of models, offering extensive coverage of all types of fairness- and diversity-aware algorithms. Additionally, it supports major information retrieval (IR) tasks, including search and recommendation, making it a versatile and comprehensive toolkit.

FairDiverse is designed to be highly extensible, providing a range of flexible APIs that allow IR researchers to efficiently develop and integrate their own fairness- and diversity-aware IR models. This extensibility ensures that researchers can tailor the toolkit to their specific needs while maintaining consistency with established evaluation protocols. This makes it an invaluable resource for advancing fairness and diversity in IR systems.

2 Toolkit Overview

In this section, we illustrate the overview pipelines of FairDiverse, as shown in Figure 1. Generally, search and recommendation tasks in IR can be considered ranking tasks with similar pipelines [60, 75]. Next, we will detail the IR pipeline steps, incorporating fairness- and diversity-aware algorithms.

IR data collection. First, we collect the user and item information. Let $\mathcal{U}$ denote the set of users, and $\mathcal{I}$ the set of items. Each user $u \in \mathcal{U}$ may have a different user profile $\mathcal{P}_u$ such as age, gender, occupation, etc. We will record the user's browsing historical item list $H_u = [i_1, i_2, \cdots, i_n]$. Each item $i \in \mathcal{I}$ is associated with specific attributes, such as categories, descriptions, and other metadata.

When a user u interacts with an IR system, they may actively input a query q_u to explicitly express their information need, a scenario commonly referred to as a search task. Alternatively, the user may not provide a query; instead, they rely on the IR system to infer their information needs and deliver relevant content, which characterizes a recommendation task.

Meanwhile, the collected data should also capture user behaviors, such as clicks, ratings, and other interactions. For example, click behavior $c_{u,i} = 1$ indicates that the user has clicked on the item, while $c_{u,i} = 0$ signifies that the user did not interact with the item on the browser or recommender platform. Rating behaviors $r \in [0, 5]$ denotes the preference degree of the item. These interaction behaviors are typically regarded as labels to train the IR models.

Table 2: The models implemented in FairDiverse.

Model types	Models
	Recommendation
Base model	DMF [70], BPR [52], GRU4Rec [56], SASRec [39], Llama3 [26], Qwen2 [4], Mistral [34]
In-processing	APR [31], DPR [84], FairDual [62], FairNeg [15], FOCF [76], IPS [35], Reg [38], Minmax-SGD [2], SDRO [58], FairPrompts [63]
Post-processing	P-MMF [61], CP-Fair [46], FairRec [48], FairRec+ [9], FairSync [64], min-regularizer [61], Tax-Rank [65], Welf [25], RAIF [44], ElasticRank [67]
	Search
Base model	MART [29], RankNet [11], RankBoost [28], AdaRank [69], Coordinate Ascent [45], LambdaMART [59], ListNet [13], Random Forests [10]
Pre-processing	CIF-Rank [74], LFR [77], gFair, iFair [41]
Post-processing	PM2 [22], xQuAD [55], DESA [49], DALETOR [72], DiversePrompts (based on GPT-4o [47] and Claude 3.5 [3])

Data processing. After collecting the IR data, it is essential to preprocess the dataset by filtering out noisy data samples, such as removing users with very few interaction histories, to ensure the quality of the data [82]. Then, we integrate user and item information with interaction behaviors and partition the data into training, validation, and test sets for model training and evaluation.

The pre-processing algorithms are primarily applied at this stage, aiming to mitigate biases present in the model input before training [53]. Specifically, certain features may enhance model performance but are influenced by sensitive attributes such as user race, and pre-processing methods aim to mitigate such effects by adjusting certain item or user features to ensure fairness and diversity.

Pre-processing methods are typically simple, easy to integrate with existing IR systems and offer good generalizability. However, these methods are independent of the model and may remove certain features that are useful for the model.

Model training. After preparing the training data, we first transform the raw data into vectorized representations (*i.e.,* embeddings) suitable for model input. Then, we design the IR models, assign appropriate loss functions, and optimize the models based on the defined loss functions.

The in-processing methods are mainly applied to the model training phase. Typically, they incorporate a fairness- and diversity-aware constraint or regularizer into the IR loss function, optimizing it to enhance ranking accuracy while ensuring fairness or diversity in the results [15, 31].

Result evaluation. Finally, after training the IR models, we apply them to evaluate their performance. We use the trained IR model to infer relevance scores for all user-item pairs in the test set. Based on these scores, we generate a ranked list by selecting items with the highest relevance scores for each user.

Post-processing methods are often based on a given set of relevance scores and re-rank the items to form a new ranked list. They formulate the problem as a constrained linear programming optimization [48, 61, 65]. The objective is to maximize the sum of relevance scores, while fairness- and diversity-aware constraints will be incorporated to ensure a fair and diverse ranked list.

3 Toolkit Details

For package details, we introduce the datasets used, the implemented models, and the evaluation metrics adopted.

3.1 Datasets

We provide details of each used dataset for recommendation and search tasks in the following parts.

3.1.1 Recommendation. Any recommendation dataset can be used for the recommendation task. Specifically, we use the RecBole [82] dataset,[1] which includes 43 commonly used datasets, all fully supported by our toolkit FairDiverse. The datasets span more than ten diverse domains, including games, products, and music. FairDiverse offers a comprehensive and fair comparison across all datasets.

To use them, researchers can simply download the datasets, place them in the `~/recommendation/dataset` directory, and configure the settings in `/properties/dataset/{dataset_name}.yaml`. The configuration files should specify which column names correspond to user ID, item ID, group ID, and other relevant fields. Once set up, running the command will enable algorithm evaluation on different datasets.

3.1.2 Search. Any dataset can be used with the pre-processing fairness models, however, in this framework we provide a working example on the COMPAS dataset [6]. This dataset evaluates the bias in the COMPAS risk assessment tool by Northpointe (now Equivant), which predicts recidivism. It includes criminal history, recidivism probability, and sensitive attributes (gender and race). Following [74], we provide a subset of 4,162 individuals distributed as 25% White males, 59% Black males, 6% White females, and 10% Black females.

For post-processing settings, we use ClueWeb09 Category B data collection [12] for our experiments.[2] The ClueWeb09 dataset consists of 200 queries and 40,537 unique documents from the Web Track 2009-2012 dataset. Notably, queries #95 and #100 are not included in our experiments due to the lack of diversity judgments. The remaining 198 queries are associated with 3 to 8 manually annotated user intents, each accompanied by binary relevance ratings assigned at the intent level.

3.2 Models

We provide all the implemented models in Table 2. Then we will delve into the details of each model. Note that to integrate different models into our toolkit architecture, some models may be

[1] https://github.com/RUCAIBox/RecSysDatasets
[2] https://lemurproject.org/clueweb09.php/

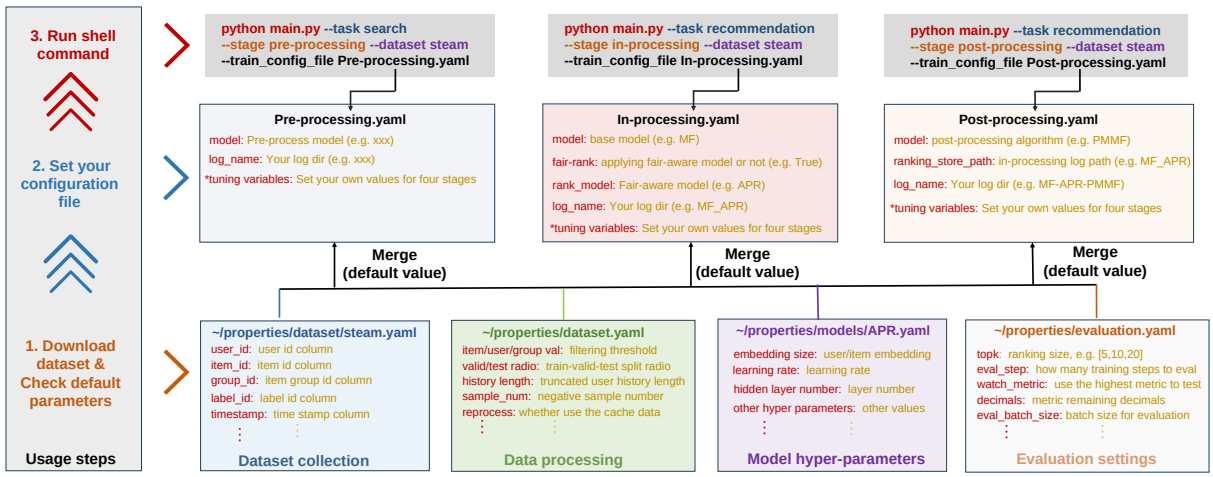


Figure 2: The usage of FairDiverse with three steps: (1) Download the datasets and check the default parameters of the four stages of pipelines; (2) Set custom configuration file to execute the pipeline. The ∗tuning_variables allow you to define variable values for the default settings across the four pipeline stages, with the In-processing configuration file overriding these default values when specified; (3) Run the shell command, with the task, stage, dataset, and your custom configuration file.

re-implemented. As a result, their performance may vary due to differences in implementation and experimental settings.

3.2.1 Recommendation. In the recommendation task, many models focus on in-processing and post-processing methods. First, we categorize the base recommendation models into non-LLM (large language model) and LLM-based models. non-LLM-based models mainly utilize user-item interaction behaviors to learn a good representation of users and items. LLM-based models rely on the prompts to rank the items according to their textual information such as item titles [19, 21]. The models are:

- Non-LLM-based models:
  - **DMF** [70]: which optimizes the matrix factorization with the deep neural networks.
  - **BPR** [52]: optimizes pairwise ranking via implicit feedback.
  - **GRU4Rec** [56]: employs gated recurrent units (GRUs) for session-based recommendations.
  - **SASRec** [39]: leverages self-attention mechanisms to model sequential user behavior.
- LLM-based models: **Llama3** [26], **Qwen2** [4], **Mistral** [34]: utilizing rank-specific prompts to conduct ranking tasks under LLMs [19].

We categorize in-processing models into re-weight, re-sample, regularizer, and prompt-based methods. The re-weighting and re-sample-based method adjusts sample weights/ratios during loss calculation, assigning higher weights/radios to underperforming item groups to enhance their support. Regularizer-based methods incorporate fairness- and diversity-aware regularization terms into the original loss function. In contrast, prompt-based methods, designed for LLM-based models, introduce fairness-aware prompts to enhance support for underperforming item groups. They are:

- Re-weight-based models:
  - **APR** [31]: an adaptive reweighing method that dynamically prioritizes samples near the decision boundary to mitigate distribution shifts.
  - **FairDual** [62]: applies dual-mirror gradient descent to dynamically compute the weight for each sample to support the worst-off groups.
  - **IPS** [35]: employs the reciprocal of the sum popularity of items within the group as the weight assigned to that group.
  - **Minmax-SGD** [2]: applies optimizing techniques to dynamically sample groups.
  - **SDRO** [58]: Improves DRO with the distributional shift to optimize group MMF.
- Re-sample-based models: **FairNeg** [15]: adjusts the group-level negative sampling distribution in the training process.
- Regularizer-based models:
  - **DPR** [84]: applies a fair-aware adversarial loss based on statistical parity and equal opportunity.
  - **FOCF** [76]: applies a fair-aware regularization loss of different groups.
  - **Reg** [38]: applies a penalty on the squared difference between the scores of two groups across all positive user-item pairs.
- Prompt-based models: **FairPrompts** [63].[3]

We categorize the post-processing models into heuristic and learning-based models. Heuristic models primarily use algorithms like greedy search to re-rank items, while learning-based models dynamically generate fairness- and diversity-aware scores, which are incorporated into the original relevance score for re-ranking. They are:

- Heuristic models:
  - **CP-Fair** [46]: applies a greedy solution to optimize the knapsack problem of fair ranking.
  - **min-regularizer** [61]: adds an additional fairness score to the ranking scores, capturing the gap between the current utility and the worst-off utility.
  - **RAIF** [44]: a model-agnostic repeat-bias-aware item fairness optimization algorithm based on mixed-integer linear programming.[4]

[3]We do not fine-tune the LLMs but only use manually designed prompts.
[4]Note that we remove repeat bias term, change the item fairness objective to make the exposure of each group closer, and extend RAIF into multi-group cases.

- Learning-based methods:
  - **P-MMF** [61]: applies a dual-mirror gradient descent method to optimize the accuracy-fairness trade-off problem.
  - **FairRec** [48], **FairRec+** [9]: proposes leveraging Nash equilibrium to guarantee Max-Min Share of item exposure.
  - **FairSync** [64]: proposes to guarantee the minimum group utility under distributed retrieval stages.
  - **Tax-Rank** [65]: applies the optimal transportation (OT) algorithm to trade-off fairness-accuracy.
  - **Welf** [25]: use the Frank-Wolfe algorithm to maximize the Welfare functions of worst-off items.
  - **ElasticRank** [67]: use elastic theory in economics to optimize the fair re-ranking process.

3.2.2 Search. Our framework uses the Ranklib library to offer a variety of ranking models, including 8 popular algorithms: MART [29], RankNet [11], RankBoost [28], AdaRank [69], Coordinate Ascent [45], LambdaMART [59], ListNet [13], Random Forests [10].

We divide pre-processing models into two categories: causal based models and probabilistic mapping clustering models. All model implementations are adapted to optimize for multiple sensitive attributes and non-binary groups.

- Causal based models:
  - **CIF-Rank** [74] estimates the causal effect of the sensitive attributes on the data and makes use of them to correct for the bias encoded.
- Probabilistic mapping clustering models: Create representations that are independent of the available sensitive attributes.
  - **LFR** [77] optimizes for group fairness by making sure that the probability of a group to be mapped to a cluster is equal to the probability of the other group.
  - **iFair** [41] optimizes for individual fairness by making sure that the distance between similar individuals is maintained in the new space.
  - **gFair** based on iFair [41], optimizes for group fairness by ensuring that the distance between similar individuals from a group is close to similar individuals from the other group.

For post-processing search models, we often utilize the diversity-aware re-ranking models. These models can be roughly categorized into unsupervised methods and supervised methods.

- Unsupervised methods:
  - **PM2** [22]: optimizes proportionality by iteratively determining the topic that best maintained the overall proportionality.
  - **xQuAD** [55]: utilizes sub-queries representing pseudo user intents and diversifies document rankings by directly estimating the relevance of the retrieved documents to each sub-queries.
  - **DiversePrompts**: a diversity ranking model based on large language models. We design specific prompts tailored for search result diversification based on the two latest closed-source LLMs: GPT-4o [47] and Claude 3.5 [3].
- Supervised methods:
  - **DESA** [49]: employs the attention mechanism to model the novelty of documents and the explicit subtopics.
  - **DALETOR** [72]: proposes diversification-aware losses to approach the optimal ranking.

3.3 Evaluation Metrics

We will delve into the details of each of used evaluation metrics in the following parts.

Recommendation. In recommendation, evaluation metrics are generally categorized into two types:

- Ranking accuracy-based metric: Mean Reciprocal Rank (MRR), Hit Radio (HR), and Normalized Discounted Cumulative Gain (NDCG) [68], utility loss (*i.e.,* Regret) [61].[5]
- Fairness- and diversity-based metric: MMF [61], GINI index [25], Entropy [36], and MinMaxRatio [51].

Search. For the search task, we adopt the official diversity evaluation metrics of the Web Track, including ERR-IA [14], α-nDCG [17], and the diversity measure Subtopic Recall (denoted as S-rec) [78]. These metrics assess the diversity of document rankings by explicitly rewarding novelty while penalizing redundancy. We follow the Web Track and utilize the provided shell command to evaluate model performance.

Furthermore, we provide support for fairness metrics, including group fairness measures such as demographic parity, ensuring proportional representation of groups, as well as proportional exposure of groups. Additionally, one can compute the in-group fairness metric proposed by Yang et al. [73], which computes the ratio between the lowest accepted score and the highest rejected score within a group. On top of group fairness, one can compute individual fairness by doing a pairwise comparison between candidates' distance in the feature space and their achieved exposure [27].

4 Toolkit Usage

Figure 2 provides an overview of the three main steps for utilizing our toolkit, FairDiverse. We will describe each step of usage in detail. The detailed configuration file parameters can be found in https://xuchen0427.github.io/FairDiverse/.

4.1 Usage Steps

Step 1. First, download the dataset you wish to test, as described in Section 3.1, and store it in the `/dataset` directory. Then, specify the parameters in `/properties/dataset/{data_name}.yaml`. Next, you need to review all default parameters for data processing, model hyperparameters, and evaluation settings.

Step 2. Then, you need to create your own configuration file, specifying the selected models and the log directory path. If you want to modify the default pipeline parameters, you can specify them directly in your own configuration file, which will override the values in the default configuration files.

Step 3. Enter `/fairdiverse` dictionary and execute the command

```
python main.py --task "recommendation"
--stage "in-processing" --dataset "steam"
--train_config_file "In-processing.yaml"
```

The args should specify the task (recommendation/search), stage (pre-processing, in-processing, post-processing), dataset, and your custom configuration file as defined in Step 2. Finally, the evaluation results and item/user utility allocations will then be recorded in your specified log file.

[5] Note that the evaluation metric NDCG in post-processing is slightly different from the common definition: NDCG in post-processing means the re-ranking quality compared to original ranking quality [61].

4.2 Usage Example

Besides our provided `main.py` file and shell command, you can also utilize the following test codes to run the toolkit.

Recommendation. Our repository includes an example dataset, Steam.[6] We provide the simple code snippets for running the in-processing and post-processing models, which are listed below. The user needs to specify the chooses "model," "dataset" and "log_name" for training and testing.

```
from recommendation.trainer import RecTrainer

config = {'model': 'BPR', 'data_type': 'pair', 'fair-rank':
    True, 'rank_model': 'APR', 'use_llm': False,
    'log_name': "test", 'dataset': 'steam'}

trainer = RecTrainer(train_config=config)
trainer.train()
```

```
from recommendation.reranker import RecReRanker

config = {'ranking_store_path': 'steam-base-mf', 'model':
    'CPFair', 'fair-rank': True, 'log_name': 'test',
    'fairness_metrics': ["GINI"], 'dataset': 'steam'}

reranker = RecReRanker(train_config=config)
reranker.rerank()
```

Search. Our repository includes a running example of the pre-processing models on the COMPAS dataset. We provide the simple code snippet for running the pre-processing models, listed as follows. One can set the "preprocessing_model" field to any of the supported models: CIFRank, LFR, gFair and iFair. Each pre-processing model has its own config file under `search/properties/models` which is automatically loaded based on your choice.

```
from search.trainer_preprocessing_ranker import
    RankerTrainer

config={"train_ranker_config": {"preprocessing_model":
    "iFair", "name": "Ranklib", "ranker": "RankNet", "lr":
    0.0001, "epochs": 10}}

reranker=RankerTrainer(train_config=config)
reranker.train()
```

For the post-processing models, our repository also provides a running example on the ClueWeb09 dataset. The simple code snippet for running these models is shown as follows.

```
from search.trainer import SRDTrainer

config = {'model': 'xquad', 'dataset': 'clueweb09', 'task':
    'search', 'mode': 'train', "log_name": "test",
    "model_save_dir": "model/", "tmp_dir": "tmp/"}

trainer = SRDTrainer(train_config=config)
trainer.train()
```

[6] http://cseweb.ucsd.edu/~wckang/Steam_games.json.gz

5 Benchmark Results Analysis

In this section, we present an analysis of partial benchmark results derived from our toolkit FairDiverse. Note that our goal is not to compare different models but to highlight the analytical direction of these results, helping researchers interpret and understand the findings more effectively and efficiently.

5.1 Recommendation

In recommendation, we primarily evaluate the performance of commonly used in-processing models, which integrate fairness constraints during training, and post-processing models, which adjust rankings after predictions to enhance fairness.

In-processing models. Table 3 presents the performance of the implemented in-process fairness and diversity-aware models in terms of accuracy (NDCG, HR, MRR) and fairness/diversity (MMF, GINI, Entropy) under the default parameters of our toolkit. The benchmark results are obtained using the Steam dataset and BPR [52] ranking models. The fairness/diversity metric is calculated at the Steam game category level.

First, from Table 3, we observe that LLM-based models generally exhibit higher fairness and diversity but lower ranking performance. In contrast, non-LLM-based models achieve better ranking performance but struggle with the long-tail problem. Secondly, different methods often exhibit significant variance in accuracy and fairness/diversity performance, excelling in some metrics while underperforming in others. Moreover, the trends across different fairness metrics are not always consistent.

Our toolkit provides researchers with a unified and convenient tool to compare various methods, explore trade-offs between different metrics, and analyze the reasons behind performance gaps. Researchers can use our toolkit to analyze results and validate their ideas across different base models and datasets.

Post-processing models. Table 4 shows the results of implemented post-process fairness and diversity-aware models in terms of re-ranking accuracy (NDCG, u-loss) and fairness/diversity (MMF, GINI, Entropy, and MMR). The benchmark results are obtained using the Steam dataset and the ranking lists provided from DMF [70] models. The fairness/diversity metric is also calculated at the Steam game category level.

First, from Table 3, we observe that post-processing models outperform in-processing methods in fairness and diversity. However, they often face an accuracy-fairness trade-off, sacrificing accuracy to enhance the fairness and diversity of item categories. With our toolkit, FairDiverse, researchers can explore this trade-off across different parameters, base models, and datasets.

5.2 Search

As for the search task, we first evaluate the output of a ranking model trained on data that was debiased by a pre-processing model and then observe the diversity of the final ranking results achieved by post-processing models.

Pre-processing models. The results of implemented pre-processing models applied on the input of a ranking model are denoted in Table 5. In this setting the ranking model is RankNet using the implementation provided by the Ranklib Library. We evaluate the

Table 3: Partial benchmark results for recommendation tasks on Steam datasets for different ranking sizes K. They are evaluated on BPR ranking models with in-processing fairness-aware and diversity-aware approaches. ↓ and ↑ indicate that a smaller or larger metric value, respectively, corresponds to better model performance. It is important to note that the reported results are based on default parameters.

Models/Metric	$K = 10$						$K = 20$					
	NDCG↑	MRR↑	HR↑	MMF↑	GINI↓	Entropy↑	NDCG↑	MRR↑	HR↑	MMF↑	GINI↓	Entropy↑
Llama3-FairPrompts	0.0304	0.0565	0.0265	0.0364	0.7332	3.8800	0.0444	0.1083	0.0308	0.0980	0.6201	4.3738
Qwen2-FairPrompts	0.0312	0.0546	0.0284	0.0324	0.7503	3.7629	0.0455	0.1070	0.0329	0.0871	0.6453	4.2590
Mistral-FairPrompts	0.0323	0.0559	0.0303	0.0315	0.7494	3.7751	0.0481	0.1132	0.0355	0.0861	0.6455	4.2602
APR	0.2925	0.2934	0.4085	0.0324	0.7257	3.9513	0.3193	0.2999	0.5058	0.0590	0.6485	4.2997
FairDual	0.3204	0.3073	0.4727	0.0330	0.7123	4.0301	0.3479	0.3136	0.5702	0.0563	0.6577	4.2745
IPS	0.3073	0.3090	0.4213	0.0249	0.7314	3.9183	0.3347	0.3155	0.5196	0.0570	0.6517	4.2824
Minmax-SGD	0.2672	0.2604	0.3961	0.0218	0.7501	3.7252	0.2958	0.2679	0.4991	0.0470	0.6936	3.9982
SDRO	0.3009	0.3056	0.4081	0.0350	0.7212	3.9754	0.3298	0.3124	0.5137	0.0619	0.6451	4.3156
FairNeg	0.2964	0.2975	0.4125	0.0673	0.6671	4.2222	0.3208	0.3036	0.5020	0.0778	0.6158	4.4067
FOCF	0.2879	0.2879	0.4041	0.0294	0.7272	3.9460	0.3141	0.2942	0.4992	0.0579	0.6472	4.3086
Reg	0.2979	0.2981	0.4162	0.0306	0.7270	3.9465	0.3245	0.3043	0.5127	0.0584	0.6497	4.2917

Table 4: Partial benchmark results for recommendation tasks on Steam datasets for different ranking sizes K. They can be evaluated using the shell command provided in our GitHub repository. It is important to note that the reported results are based on default parameters.

Models/Metric	$K = 10$						$K = 20$					
	R-NDCG↑	u-loss↓	MMF↑	GINI↓	Entropy↑	MMR↑	R-NDCG↑	u-loss↓	MMF↑	GINI↓	Entropy↑	MMR↑
CP-Fair	0.9981	0.0035	0.2135	0.4424	4.8441	0.0196	0.9969	0.0055	0.2118	0.4349	4.9064	0.0301
min-regularizer	0.9272	0.1234	0.3984	0.1373	5.3796	0.2740	0.9359	0.0896	0.4004	0.1326	5.3818	0.2761
RAIF	0.9881	0.0233	0.2937	0.3293	5.0080	0.0248	0.9829	0.0302	0.3333	0.2585	5.1558	0.0358
P-MMF	0.9691	0.0536	0.3140	0.2792	5.1911	0.0685	0.9675	0.0482	0.3289	0.2429	5.2597	0.0987
FairRec	0.9529	0.1088	0.1866	0.5098	4.5372	0.0157	0.9497	0.0990	0.1779	0.5179	4.5234	0.0175
FairRec+	0.9773	0.0540	0.1758	0.5254	4.5108	0.0119	0.9750	0.0510	0.1609	0.5463	4.4594	0.0134
FairSync	0.9816	0.0353	0.2477	0.3951	4.9152	0.0237	0.9785	0.0383	0.2553	0.3705	5.0165	0.0312
TaxRank	0.9438	0.1015	0.2421	0.3791	4.9846	0.0149	0.9303	0.1080	0.2907	0.3078	5.1287	0.0209
Welf	0.9668	0.0575	0.3216	0.2638	5.2254	0.0820	0.9682	0.0467	0.3322	0.2383	5.2674	0.1112

Table 5: Benchmark results for search task on the COMPAS dataset for different ranking sizes K, obtained on RankNet ranking models with pre-processing fairness-aware approaches. Evaluated using the shell command provided in our GitHub repository. %D: diversity of the Female-Black group (the disadvantaged intersectional group); IGF: in-group-fairness measure as an average over the groups; yNN: individual fairness; NDCG-loss: NDCG loss. The reported results are based on default parameters.

Models/Metric	$K = 100$				$K = 300$			
	%D↑	IGF↑	yNN↑	NDCG-loss↑	%D↑	IGF↑	yNN↑	NDCG-loss↑
RankNet	0.10	-	0.86	0.92	0.11	-	0.86	0.95
CIFRank	0.13	1.00	0.86	0.78	0.10	1.00	0.86	0.84
LFR	0.09	1.00	0.86	0.93	0.09	0.93	0.86	0.72
gFair	0.14	1.00	0.86	0.39	0.10	1.00	0.86	0.52
iFair	0.42	0.55	0.86	0.85	0.19	0.30	0.86	0.89

performance on the COMPAS dataset. NDCG-loss represents the loss in utility w.r.t. the original ranking and the original scores.

All models, except LFR, manage to improve or maintain the diversity in top-k of the Female-Black group, which is the most disadvantaged intersectional group. Out of all models gFair has the biggest loss in utility, while individual fairness (yNN) is not affected. In-group-fairness (IGF) is measured on the transformed representations, not on the output ranking, to check whether the transformed data respects the order within a group. It can be observed that CIFRank and gFair obtain perfect IGF. Using FairDiverse researchers can compare the impact of pre-processing models on the output ranking given the trade-offs between group fairness, individual fairness and utility loss.

Post-processing models. The results of implemented post-process search result diversification models are denoted in Table 6. We evaluate the performance based on the ClueWeb09 dataset. The initial ranking list is provided by Lemur.[7] We utilize the top 50 documents from the initial ranking list for testing these diversified ranking models' performance.

From the results, we can observe that, first, supervised diversified search models demonstrate superior performance compared to

[7] https://lemurproject.org/clueweb09.php/

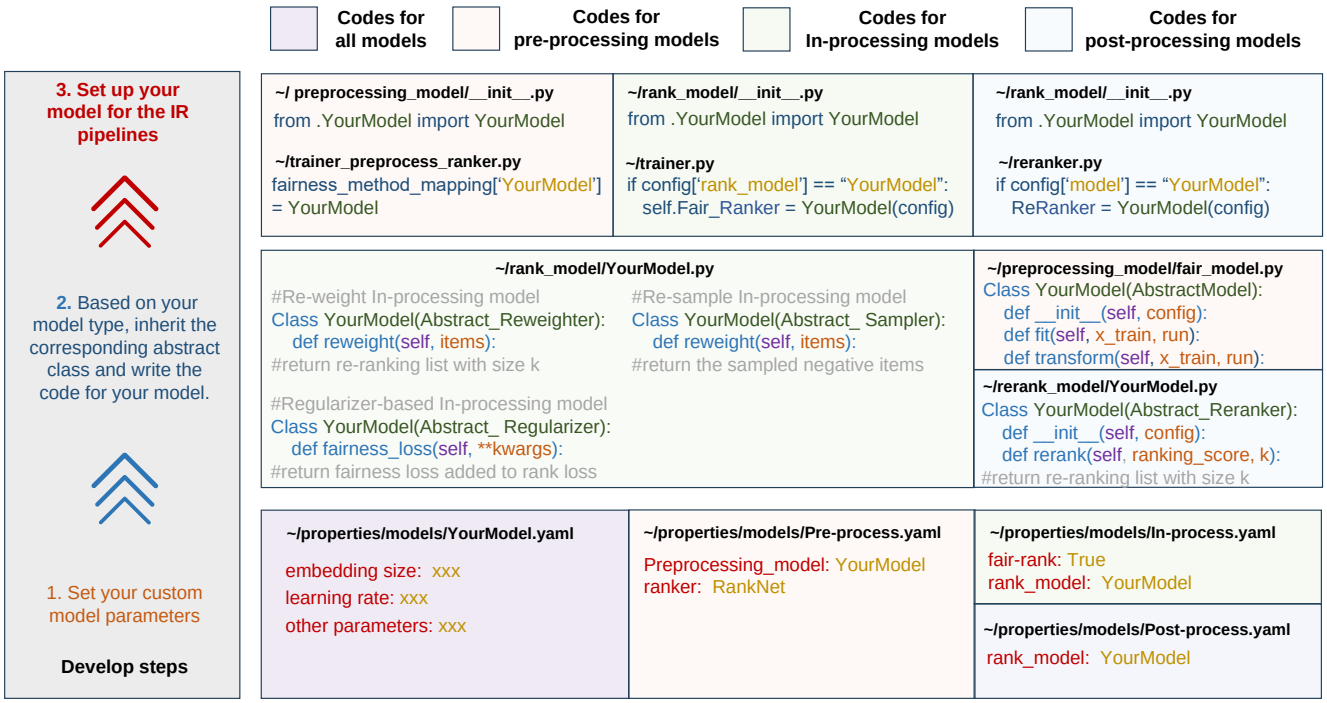


Figure 3: The custom steps for fairness and diversity-aware search and recommender models named *YourModel*. The differently colored areas indicate the code you need to add when developing different types of model. Generally, you can follow three steps: (1) define custom model parameters, (2) develop your model based on its type, and (3) integrate it into the pipeline.

Table 6: Benchmark results for the post-processing search result diversification models on the ClueWeb09 datasets with different ranking sizes K. We evaluate the performance using the shell command provided by the official Web Track which is also available in our GitHub repository. A larger metric value indicates superior model performance. The reported results are based on default parameter settings.

Models/Metric	$K = 5$			$K = 10$			$K = 20$		
	ERR-IA	α-nDCG	S-rec	ERR-IA	α-nDCG	S-rec	ERR-IA	α-nDCG	S-rec
PM2	0.2626	0.3292	0.4793	0.2824	0.3684	0.5708	0.2913	0.3989	0.6407
xQuAD	0.2002	0.2511	0.3961	0.2166	0.2838	0.4701	0.2272	0.3230	0.5761
DiversePrompts (GPT-4o)	0.2890	0.3514	0.4972	0.3054	0.3833	0.5791	0.3131	0.4099	0.6396
DiversePrompts (Claude 3.5)	0.3136	0.3800	0.4981	0.3292	0.4079	0.5741	0.3372	0.4348	0.6486
DESA	0.3497	0.4226	0.5195	0.3642	0.4452	0.5914	0.3703	0.4655	0.6438
DALETOR	0.2770	0.3362	0.4609	0.2948	0.3732	0.5644	0.3047	0.4085	0.6581

unsupervised models. Moreover, diversified rankers based on LLMs consistently outperform traditional unsupervised methods. This observation suggests that the knowledge acquired by LLMs during the pre-training stage significantly enhances the diversity of search results. To facilitate the exploration of these models, we present FairDiverse, a comprehensive toolkit that enables researchers to analyze various parameters, base models, and datasets.

6 Customizing Models in FairDiverse

We outline the steps for customizing and evaluating new IR models using the APIs we provide. Detailed API descriptions and source code can be found in https://xuchen0427.github.io/FairDiverse/. The provided APIs can be used by installing them via pip:

```
pip install fairdiverse
```

6.1 Steps

Figure 3 illustrates the three key steps for implementing fairness- and diversity-aware IR models named *YourModel*.

Step 1. Configure your custom model parameters and save them in a newly created `YourModel.yaml` file in the `/properties/models/` directory. Then you can change the model in the running configuration file `Post-processing.yaml`.

Step 2. Select the appropriate Python abstract class from our provided options based on your model type and implement your model in a newly created file, `YourModel.py`, stored in the corresponding directory. You can use the integrated tools and common parameters within the abstract class. Researchers only need to focus on designing the model without worrying about the rest of the pipeline.

Step 3. Configure your model for the training pipeline by following these steps: import your custom model package in the corresponding file (`/model_type/__init__.py`) and define the model in the appropriate script (`/train.py`, `/reranker.py`).

6.2 Examples

Here, we provide two example codes demonstrating how to design a custom search and recommendation model, respectively.

```
#/recommendation/rank_model/YourModel.py
class YourModel(Abstract_Regularizer):
    def __init__(self, config, group_weight):
        super().__init__(config)

    def fairness_loss(self, input_dict):
        return torch.var(input_dict['scores'])

#/recommendation/rank_model/__init__.py
from .YourModel import YourModel

#/recommendation/trainer.py
if config["model"] == "YourModel":
  self.Model = YourModel(config)
```

```
#/search/preprocessing_model/YourModel.py
class YourModel(PreprocessingFairnessIntervention):
    def __init__(self, configs, dataset):
        super().__init__(configs, dataset)

    def fit(self, X_train, run):
    # Train the fairness model using the training set.

    def transform(self, X_train, run file_name=None):
    # Apply the fairness transformation to the dataset.

#/search/preprocessing_model/__init__.py
from .YourModel import YourModel
fairness_method_mapping['YourModel'] = YourModel
```

7 Related Work

Beyond-Accuracy in IR. In modern IR systems, beyond-accuracy objectives play a crucial role in building a more effective and responsible ecosystem [24, 37]. Beyond-accuracy objectives primarily include diversity [22], fairness [48, 61], novelty [32], and serendipity [80]. Among these factors, this toolkit primarily focuses on fairness and diversity.

Fairness and diversity in IR. Fairness and diversity are gaining increasing attention in the IR field, as both seek to support underrepresented user and item groups [20, 22, 43, 57]. Previous studies have often explored fairness and diversity from the perspectives of different stakeholders, such as users and items [1], as well as at varying granularities, including both group-level and individual-level fairness [7, 61]. Based on different stages in the IR pipeline, previous methods are often categorized into three types: pre-processing [53], in-processing [15, 31, 38], and post-processing approaches [22, 23, 61, 65]. As for the evaluation, they are also based on different metrics, including the Gini index [25], MMF [61] in recommendation, and α-nDCG [17], NRBP [18] in search. However, fairness and diversity often lack unified evaluation settings. This paper introduces FairDiverse, a benchmarking toolkit designed to comprehensively assess different models under different IR tasks.

Fairness and diversity toolkits. Most fairness and diversity toolkits are implemented under classification tasks. For example, FFB [30] implements diverse in-processing models for addressing group fairness problems. Fairlearn[8], AIF360[5] and Aequitas[33] implement the unfairness mitigation algorithms using Scikit-learn [40] API design. However, these methods cannot be directly applied to ranking tasks. Although some toolkits [81] have been proposed to incorporate fairness and diversityd in IR, they primarily focus on recommendation tasks and implement only a limited number of in-processing models. Our toolkit, FairDiverse, offers the most extensive collection of models, covering a wide range of fairness- and diversity-aware algorithms across both search and recommendation tasks. Moreover, FairDiverse is highly extensible, offering flexible APIs for easy integration of new fairness- and diversity-aware IR models, unlike other toolkits with complex class inheritance.

8 Conclusion & Future Work

We have presented FairDiverse, a comprehensive IR toolkit designed to facilitate standardized evaluations of fairness-aware and diversity-aware algorithms. FairDiverse offers three key advantages over other toolkits. (1) It provides a comprehensive framework for integrating various models at different IR pipeline stages; (2) It implements 29 fairness- and diversity-aware models for 16 recommendation and search base models; (3) It offers flexible APIs for researchers to develop and integrate custom models.

However, a limitation of FairDiverse is that it only supports single-round fairness- and diversity-aware IR models. It does not yet support dynamic settings, such as long-term fairness or fairness under dynamic feedback loops [66]. In future work, we plan: (i) to include more LLM-based models [83]; (ii) to support dynamic scenarios and exploring the use of LLM agents [79] for simulation and evaluation; and (iii) to incorporate more beyond-accuracy models, including novelty- and serendipity-aware models.

Acknowledgments

This work was funded by the National Key R&D Program of China (2023YFA1008704), the National Science and Technology Major Project No. 2022ZD0120103, the National Natural Science Foundation of China (No. 62472426). The work was partially done at the Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE. This research was (partially) supported by the Dutch Research Council (NWO), under project numbers 024.004.022, NWA.1389.20.183, and KICH3.LTP.20.006, and the European Union's Horizon Europe program under grant agreement No 101070212. 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, Gediminas Adomavicius, Robin Burke, Ido Guy, Dietmar Jannach, Toshihiro Kamishima, Jan Krasnodebski, and Luiz Pizzato. 2020. Multistakeholder Recommendation: Survey and Research Directions. *User Modeling and User-Adapted Interaction* 30, 1 (2020), 127–158.

[2] Jacob D Abernethy, Pranjal Awasthi, Matthäus Kleindessner, Jamie Morgenstern, Chris Russell, and Jie Zhang. 2022. Active Sampling for Min-Max Fairness. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings*

of Machine Learning Research, Vol. 162), Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 53–65.

[3] Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku. https://api.semanticscholar.org/CorpusID:268232499

[4] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609* (2023).

[5] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. 2018. AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias. https://arxiv.org/abs/1810.01943

[6] Machine Bias. 2016. There's Software Used Across the Country to Predict Future Criminals. And It's Biased Against Blacks. ProPublica. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminalsentencing.

[7] 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.

[8] 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.* Technical Report MSR-TR-2020-32. Microsoft.

[9] Arpita Biswas, Gourab K. Patro, Niloy Ganguly, Krishna P Gummadi, and Abhijnan Chakraborty. 2021. Toward Fair Recommendation in Two-sided Platforms. *ACM Transactions on the Web (TWEB)* 16, 2 (2021), 1–34.

[10] Leo Breiman. 2001. Random forests. *Machine learning* 45 (2001), 5–32.

[11] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. 89–96.

[12] Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. 2009. The clueweb09 dataset, 2009. *URL http://boston. lti. cs. cmu. edu/Data/clueweb09* (2009).

[13] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. 129–136.

[14] Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. 2009. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009*, David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy Lin (Eds.). ACM, 621–630. https://doi.org/10.1145/1645953.1646033

[15] Xiao Chen, Wenqi Fan, Jingfan Chen, Haochen Liu, Zitao Liu, Zhaoxiang Zhang, and Qing Li. 2023. Fairly Adaptive Negative Sampling for Recommendations *(WWW '23)*. Association for Computing Machinery, New York, NY, USA, 3723–3733. https://doi.org/10.1145/3543507.3583355

[16] Gobinda G Chowdhury. 2010. *Introduction to modern information retrieval.* Facet publishing.

[17] Charles L. A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, Sung-Hyon Myaeng, Douglas W. Oard, Fabrizio Sebastiani, Tat-Seng Chua, and Mun-Kew Leong (Eds.). ACM, 659–666. https://doi.org/10.1145/1390334.1390446

[18] Charles L. A. Clarke, Maheedhar Kolla, and Olga Vechtomova. 2009. An Effectiveness Measure for Ambiguous and Underspecified Queries. In *Advances in Information Retrieval Theory, Second International Conference on the Theory of Information Retrieval, ICTIR 2009, Cambridge, UK, September 10-12, 2009, Proceedings (Lecture Notes in Computer Science, Vol. 5766)*, Leif Azzopardi, Gabriella Kazai, Stephen E. Robertson, Stefan M. Rüger, Milad Shokouhi, Dawei Song, and Emine Yilmaz (Eds.). Springer, 188–199. https://doi.org/10.1007/978-3-642-04417-5_17

[19] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering ChatGPT's Capabilities in Recommender Systems. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 1126–1132. https://doi.org/10.1145/3604915.3610646

[20] Sunhao Dai, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. Bias and Unfairness in Information Retrieval Systems: New Challenges in the LLM Era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 6437–6447. https://doi.org/10.1145/3637528.3671458

[21] Sunhao Dai, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. Bias and Unfairness in Information Retrieval Systems: New Challenges in the LLM Era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 6437–6447. https://doi.org/10.1145/3637528.3671458

[22] Van Dang and W. Bruce Croft. 2012. Diversity by proportionality: an election-based approach to search result diversification. In *The 35th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012*, William R. Hersh, Jamie Callan, Yoelle Maarek, and Mark Sanderson (Eds.). ACM, 65–74. https://doi.org/10.1145/2348283.2348296

[23] Van Dang and W Bruce Croft. 2012. Diversity by proportionality: an election-based approach to search result diversification. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 65–74.

[24] Maarten de Rijke. 2023. Beyond-Accuracy Goals, Again. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 2–3.

[25] Virginie Do, Sam Corbett-Davies, Jamal Atif, and Nicolas Usunier. 2021. Two-sided Fairness in Rankings via Lorenz Dominance. *Advances in Neural Information Processing Systems* 34 (2021), 8596–8608.

[26] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[27] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. 214–226.

[28] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of machine learning research* 4, Nov (2003), 933–969.

[29] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[30] Xiaotian Han, Jianfeng Chi, Yu Chen, Qifan Wang, Han Zhao, Na Zou, and Xia Hu. 2023. FFB: A Fair Fairness Benchmark for In-Processing Group Fairness Methods. arXiv:2306.09468 [cs.LG]

[31] Zhihao Hu, Yiran Xu, and Xinmei Tian. 2023. Adaptive Priority Reweighing for Generalizing Fairness Improvement. In *2023 International Joint Conference on Neural Networks (IJCNN)*. 01–08. https://doi.org/10.1109/IJCNN54540.2023.10191757

[32] Neil Hurley and Mi Zhang. 2011. Novelty and diversity in top-n recommendation–analysis and evaluation. *ACM Transactions on Internet Technology (TOIT)* 10, 4 (2011), 1–30.

[33] Sérgio Jesus, Pedro Saleiro, Beatriz M Jorge, Rita P Ribeiro, João Gama, Pedro Bizarro, Rayid Ghani, et al. 2024. Aequitas Flow: Streamlining Fair ML Experimentation. *arXiv preprint arXiv:2405.05809* (2024).

[34] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] https://arxiv.org/abs/2310.06825

[35] Meng Jiang, Keqin Bao, Jizhi Zhang, Wenjie Wang, Zhengyi Yang, Fuli Feng, and Xiangnan He. 2024. Item-side Fairness of Large Language Model-based Recommendation System. In *Proceedings of the ACM on Web Conference 2024*. 4717–4726.

[36] Lou Jost. 2006. Entropy and diversity. *Oikos* 113, 2 (2006), 363–375.

[37] Marius Kaminskas and Derek Bridge. 2016. Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 7, 1 (2016), 1–42.

[38] Toshihiro Kamishima and Shotaro Akaho. 2017. Considerations on recommendation independence for a find-good-items task. (2017).

[39] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive Sequential Recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[40] Oliver Kramer and Oliver Kramer. 2016. Scikit-learn. *Machine learning for evolution strategies* (2016), 45–53.

[41] Preethi Lahoti, Krishna P Gummadi, and Gerhard Weikum. 2019. ifair: Learning individually fair data representations for algorithmic decision making. In *2019 ieee 35th international conference on data engineering (icde)*. IEEE, 1334–1345.

[42] Ming Li, Yuanna Liu, Sami Jullien, Mozhdeh Ariannezhad, Andrew Yates, Mohammad Aliannejadi, and Maarten de Rijke. 2024. Are We Really Achieving Better Beyond-Accuracy Performance in Next Basket Recommendation?. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 924–934. https://doi.org/10.1145/3626772.3657835

[43] Yunqi Li, Hanxiong Chen, Shuyuan Xu, Yingqiang Ge, Juntao Tan, Shuchang Liu, and Yongfeng Zhang. 2022. Fairness in Recommendation: A Survey. *arXiv preprint arXiv:2205.13619* (2022).

[44] Yuanna Liu, Ming Li, Mohammad Aliannejadi, and Maarten de Rijke. 2025. Repeat-bias-aware Optimization of Beyond-accuracy Metrics for Next Basket Recommendation. *arXiv e-prints* (2025), arXiv–2501.

[45] Donald Metzler and W Bruce Croft. 2007. Linear feature-based models for information retrieval. *Information Retrieval* 10 (2007), 257–274.

[46] Mohammadmehdi Naghiaei, Hossein A Rahmani, and Yashar Deldjoo. 2022. CPFair: Personalized Consumer and Producer Fairness Re-ranking for Recommender Systems. *arXiv preprint arXiv:2204.08085* (2022).

[47] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] https://arxiv.org/abs/2303.08774

[48] Gourab K. Patro, Arpita Biswas, Niloy Ganguly, Krishna P. Gummadi, and Abhijnan Chakraborty. 2020. FairRec: Two-sided Fairness for Personalized Recommendations in Two-sided Platforms. In *Proceedings of the Web Conference*. 1194–1204.

[49] Xubo Qin, Zhicheng Dou, and Ji-Rong Wen. 2020. Diversifying Search Results using Self-Attention Network. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 1265–1274. https://doi.org/10.1145/3340531.3411914

[50] Xubo Qin, Zhicheng Dou, and Ji-Rong Wen. 2020. Diversifying search results using self-attention network. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1265–1274.

[51] Abdur Rehman, Kashif Javed, Haroon A Babri, and Muhammad Nabeel Asim. 2018. Selection of the most relevant terms based on a max-min ratio metric for text classification. *Expert Systems with Applications* 114 (2018), 78–96.

[52] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. *arXiv preprint arXiv:1205.2618* (2012).

[53] Clara Rus, Andrew Yates, and Maarten de Rijke. 2024. A Study of Pre-processing Fairness Intervention Methods for Ranking People. In *European Conference on Information Retrieval*. Springer, 336–350.

[54] Rodrygo LT Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th international conference on World wide web*. 881–890.

[55] Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti (Eds.). ACM, 881–890. https://doi.org/10.1145/1772690.1772780

[56] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 17–22.

[57] Lequn Wang and Thorsten Joachims. 2021. User Fairness, Item Fairness, and Diversity for Rankings in Two-Sided Markets. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*. 23–41.

[58] Hongyi Wen, Xinyang Yi, Tiansheng Yao, Jiaxi Tang, Lichan Hong, and Ed H. Chi. 2022. Distributionally-robust Recommendations for Improving Worst-case User Experience. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 3606–3610. https://doi.org/10.1145/3485447.3512255

[59] Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13 (2010), 254–270.

[60] Jiayi Xie, Shang Liu, Gao Cong, and Zhenzhong Chen. 2024. UnifiedSSR: A Unified Framework of Sequential Search and Recommendation. In *Proceedings of the ACM on Web Conference 2024*. 3410–3419.

[61] Chen Xu, Sirui Chen, Jun Xu, Weiran Shen, Xiao Zhang, Gang Wang, and Zhenhua Dong. 2023. P-MMF: Provider Max-min Fairness Re-ranking in Recommender System. In *Proceedings of the ACM Web Conference 2023*. 3701–3711.

[62] Chen Xu, Yuxin Li, Wenjie Wang, Liang Pang, Jun Xu, and Tat-Seng Chua. 2025. Bridging Jensen Gap for Max-Min Group Fairness Optimization in Recommendation. *ICLR* (2025).

[63] Chen Xu, Wenjie Wang, Yuxin Li, Liang Pang, Jun Xu, and Tat-Seng Chua. 2024. A Study of Implicit Ranking Unfairness in Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 7957–7970. https://doi.org/10.18653/v1/2024.findings-emnlp.467

[64] Chen Xu, Jun Xu, Yiming Ding, Xiao Zhang, and Qi Qi. 2024. FairSync: Ensuring Amortized Group Exposure in Distributed Recommendation Retrieval. In *Proceedings of the ACM Web Conference 2024* (Singapore, Singapore) *(WWW '24)*. Association for Computing Machinery, New York, NY, USA, 1092–1102. https://doi.org/10.1145/3589334.3645413

[65] Chen Xu, Xiaopeng Ye, Wenjie Wang, Liang Pang, Jun Xu, and Tat-Seng Chua. 2024. A Taxation Perspective for Fair Re-ranking. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Washington DC, USA) *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 1494–1503. https://doi.org/10.1145/3626772.3657766

[66] Chen Xu, Xiaopeng Ye, Jun Xu, Xiao Zhang, Weiran Shen, and Ji-Rong Wen. 2023. LTP-MMF: Towards Long-term Provider Max-min Fairness Under Recommendation Feedback Loops. *arXiv preprint arXiv:2308.05902* (2023).

[67] Chen Xu, Jujia Zhao, Wenjie Wang, Liang Pang, Jun Xu, Tat-Seng Chua, and Maarten de Rijke. 2025. Understanding Accuracy-Fairness Trade-offs in Re-ranking through Elasticity in Economics. arXiv:2504.14991 [cs.IR]

[68] Jun Xu, Xiangnan He, and Hang Li. 2018. Deep Learning for Matching in Search and Recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 1365–1368.

[69] Jun Xu and Hang Li. 2007. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 391–398.

[70] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep Matrix Factorization Models for Recommender Systems.. In *IJCAI*, Vol. 17. Melbourne, Australia, 3203–3209.

[71] Le Yan, Zhen Qin, Rama Kumar Pasumarthi, Xuanhui Wang, and Michael Bendersky. 2021. Diversification-aware learning to rank using distributed representation. In *Proceedings of the Web Conference 2021*. 127–136.

[72] Le Yan, Zhen Qin, Rama Kumar Pasumarthi, Xuanhui Wang, and Michael Bendersky. 2021. Diversification-Aware Learning to Rank using Distributed Representation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 127–136. https://doi.org/10.1145/3442381.3449831

[73] Ke Yang, Vasilis Gkatzelis, and Julia Stoyanovich. 2019. Balanced Ranking with Diversity Constraints. *arXiv preprint arXiv:1906.01747* (2019).

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

[75] Jing Yao, Zhicheng Dou, Ruobing Xie, Yanxiong Lu, Zhiping Wang, and Ji-Rong Wen. 2021. USER: A unified information search and recommendation model based on integrated behavior sequence. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2373–2382.

[76] Sirui Yao and Bert Huang. 2017. Beyond parity: Fairness objectives for collaborative filtering. *Advances in neural information processing systems* 30 (2017).

[77] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *International conference on machine learning*. PMLR, 325–333.

[78] ChengXiang Zhai, William W Cohen, and John Lafferty. 2015. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Acm sigir forum*, Vol. 49. ACM New York, NY, USA, 2–9.

[79] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024. On generative agents in recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval*. 1807–1817.

[80] Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, and Tamas Jambor. 2012. Auralist: introducing serendipity into music recommendation. In *Proceedings of the fifth ACM international conference on Web search and data mining*. 13–22.

[81] Wayne Xin Zhao, Yupeng Hou, Xingyu Pan, Chen Yang, Zeyu Zhang, Zihan Lin, Jingsen Zhang, Shuqing Bian, Jiakai Tang, Wenqi Sun, Yushuo Chen, Lanling Xu, Gaowei Zhang, Zhen Tian, Changxin Tian, Shanlei Mu, Xinyan Fan, Xu Chen, and Ji-Rong Wen. 2022. RecBole 2.0: Towards a More Up-to-Date Recommendation Library. *arXiv preprint arXiv:2206.07351* (2022).

[82] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2021. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. In *CIKM*. ACM, 4653–4664.

[83] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2025. A Survey of Large Language Models. arXiv:2303.18223 [cs.CL] https://arxiv.org/abs/2303.18223

[84] Ziwei Zhu, Jianling Wang, and James Caverlee. 2020. Measuring and Mitigating Item Under-Recommendation Bias in Personalized Ranking Systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 449–458. https://doi.org/10.1145/3397271.3401177