Distributionally-robust Recommendations for Improving Worst-case User Experience

Hongyi Wen[1*], Xinyang Yi[2], Tiansheng Yao[2], Jiaxi Tang[2], Lichan Hong[2], Ed H. Chi[2]
[1]Information Science, Cornell Tech, USA [2]Google Inc., USA
[1]vennhy@gmail.com [2]{xinyang, tyao, jiaxit, lichan, edchi}@google.com

ABSTRACT

Modern recommender systems have evolved rapidly along with deep learning models that are well-optimized for overall performance, especially those trained under Empirical Risk Minimization (ERM). However, a recommendation algorithm that focuses solely on the average performance may reinforce the exposure bias and exacerbate the “rich-get-richer” effect, leading to unfair user experience. In a simulation study, we demonstrate that such performance gap among various user groups is enlarged by an ERM-trained recommender in the long-term. To mitigate such amplification effects, we propose to optimize for the worst-case performance under the Distributionally Robust Optimization (DRO) framework, with the goal of improving long-term fairness for disadvantaged subgroups. In addition, we propose a simple-yet-effective streaming optimization improvement called Streaming-DRO (S-DRO), which effectively reduces loss variances for recommendation problems with sparse and long-tailed data distributions. Our results on two large-scale datasets suggest that (1) DRO is a flexible and effective technique for improving worst-case performance, and (2) Streaming-DRO outperforms vanilla DRO and other strong baselines by improving the worst-case and overall performance at the same time.

CCS CONCEPTS

• **Computing methodologies → Machine learning**; • **Information systems → Recommender systems**.

KEYWORDS

Distributional robustness, Robust learning, Recommendation.

ACM Reference Format:
Hongyi Wen[1*], Xinyang Yi[2], Tiansheng Yao[2], Jiaxi Tang[2], Lichan Hong[2], Ed H. Chi[2]. 2022. Distributionally-robust Recommendations for Improving Worst-case User Experience. In *Proceedings of the ACM Web Conference 2022 (WWW ’22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3485447.3512255

1 INTRODUCTION

Recommender systems have been increasingly important for driving engagements in multiple user-facing products, including content personalizations [12, 33], social networks [20] and e-commerce [7, 19]. A critical task for personalized recommendation is *retrieval*: given a query (user), identify the most few relevant items from a large item corpus. In many large-scale recommenders, queries and items are embedded in the same latent space with Deep Neural Networks (DNNs) as encoders, and trained under the *Empirical Risk Minimization* (**ERM**) framework by minimizing a loss function uniformly for all training samples. The retrieval task is then formulated as the nearest neighbor search problem in the embedding space. The item embeddings are indexed offline, making real-time serving scalable for a corpus at the scale of millions to billions.

Although prior work has demonstrated success in developing accurate recommendation models for the overall performance, it is brought to the community’s attention that certain sub-populations may suffer from worse performance [3, 4, 6, 18, 30, 32]. A classic problem is popularity bias, where more prevalent demographic user groups receive better performance [1]. Several work has been proposed to tackle the sub-population imbalance problem. For example, up-sampling minor subgroups or down-sampling majority subgroups [11]. However these methods are error-prone to outliers and susceptible to over-fitting when oversampling. It is still an open research question regarding how to lift the worst-case performance while keeping the overall performance unharmed.

Inspired by a optimization framework called *Distributionally-Robust Optimization* (**DRO**) that aims to optimize the worst-case performance [10, 21], in this paper, we study the effectiveness of DRO to tackle the subgroup performance gap problem in recommendations. Specifically:

- We empirically validate that a recommender trained with ERM further enlarges the gap between different groups of users through a simulation study. This motivates us to focus on improving the worst-case performance.
- We show that the DRO framework proposed in [27] is suboptimal to achieve distributional robustness in recommendations. We develop a simple-yet-effective optimization improvement (Streaming-DRO) to reduce large loss variances during training.
- Extensive experiments on recommendation datasets demonstrate the effectiveness of Streaming-DRO on improving the worst-case performance while maintaining a better overall performance compared to a set of strong baselines.

2 RELATED WORK

Long-term evaluations of recommender systems. Most evaluations for recommender systems focus on the single-step setting [2, 26]. In a recent work, Yao et al. [30] propose a simulation framework to analyze the sensitivity of recommenders to popularity bias. They found non-trivial temporal dynamics of popularity bias and raised concerns on what types of trajectories a recommender should create

[*] Work done while the first author was an intern at Google Inc.

This work is licensed under a Creative Commons Attribution International 4.0 License.

WWW ’22, April 25–29, 2022, Virtual Event, Lyon, France
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9096-5/22/04.
https://doi.org/10.1145/3485447.3512255

in the longer term. As a step further, we focus on user subgroups with different preferences on item popularity, and demonstrate how the performance gap across these user subgroups change over time after multiple rounds of recommendations.

Re-weighting losses for sub-distributions. A line of work attempt to deal with imbalanced datasets with importance sampling [16], where training sample weights are the inverse of the propensities of class frequency. However, directly applying such methods yield poor performance on real-world datasets with high class imbalance due to the large variance of sparse classes [5]. Inspired by the diminishing benefits of additional data points, Cui et al. [8] propose a framework by replacing class frequency with effective number of samples. A class-balanced term is used to rebalance the loss and show improved performance on long-tailed datasets. These two methods are closest to ours in terms of reweighting costs for different sub-classes from the data.

Group fairness. It's worth mentioning that many recent works explore addressing the group fairness problems in the spirit of "Equal of Opportunity" [13] for a variety of machine learning tasks, e.g. image classification [28], language modeling [23], and ranking [3]. A key measure of success is to achieve equality for intergroup performance. We are inspired by a philosophical framework called "Rawlsian Fairness" [24], where we focus on the worst-case optimization problem and do not explicitly tackle the fairness problem, though empirically they could be achieved at the same time.

3 WORST-CASE OPTIMIZATION FOR RECOMMENDATION

3.1 Simulation study

In this simulation study, to empirically check the performance gap across user subgroups when training recommenders with ERM, we (1) use MovieLens-1M dataset [14] to train the recommender and generate top-k recommendations (k=100); (2) simulate user feedback on recommendations; and (3) re-train our recommender with new data. Such a process is repeated 10 rounds to examine the "long-term" effect of a recommender system.

3.1.1 User subgroups. On the MovieLens-1M dataset, users with extensive interaction history tend to consume more long-tailed items (pearson $r = -.824, p < .0001$). Therefore, inspired by Abdollahpouri et al.[2], we identify user subgroups according to their preferences on popular items. That is, grouping users according to the ratio of *popular items* (i.e., top 20%) in their profiles: (1) **Niche users**, who seek to watch long-tail movies (20% of total users), (2) **Diverse users**, who have a broad taste on both popular and long-tail movies (27% of total users), and (3) **Blockbuster users**, who watch popular movies most of time (54% of all users).

3.1.2 Model setups. Following previous works [31, 33], we used a two-tower model trained with softmax cross-entropy loss as the backbone model architecture. In detail, let $\mathbf{q}_i, \mathbf{c}_i$ be the embeddings of queries and items after being encoded by two MLPs. For a batch of positive training pairs $\{(\mathbf{q}_j, \mathbf{c}_j)\}_{j=1}^{N}$, assume a batch of M negative items $\{(\mathbf{c}_i)\}_{i=1}^{M}$ is sampled uniformly from the candidate corpus. Let τ denote the temperature, the softmax cross-entropy loss is defined as $\mathcal{L}(\theta) = -\frac{1}{N}\sum_{j\in[N]} \log \frac{\exp(\mathbf{q}_j^T \mathbf{c}_j/\tau)}{\exp(\mathbf{q}_j^T \mathbf{c}_j/\tau)+\sum_{i\in[M]}\exp(\mathbf{q}_j^T \mathbf{c}_i/\tau)}$.

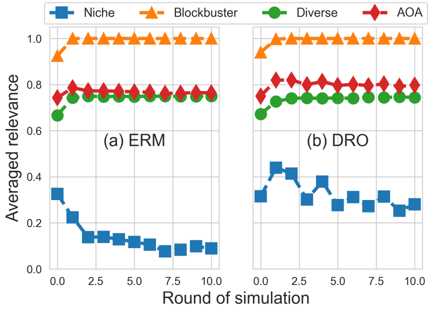

Figure 1: (a) Worst-case performance consistently drops in ERM. However, such a trend would not be observed by monitoring Average-over-all performance alone. (b) DRO shows the promise to mitigate such amplification effects.

A recommender trained under the ERM framework seeks to find the minimizer of the above loss, namely $\theta_{ERM} = \arg\min_{\theta\in\Theta} \mathcal{L}(\theta)$. The underlying assumption is that each training sample takes equal weight of $\frac{1}{|D|}$ in evaluating the hardness of the learning task. Intuitively, ERM assigns more weights to the majority subgroups and less weights on the minority subgroups, which potentially leads to then uneven recommendation performance across sub-populations.

3.1.3 Evaluation. After training the two-tower recommender with ERM, we select the top-100 recommendations from full corpus. To simulate users' preferences, we used the β-preference feedback model proposed in [30]. This model assumes binary feedback from users, and we use *top 20% interacted item* as the threshold to differentiate between popular and non-popular items. Specifically, blockbuster users favors popular items while niche users favors non-popular items. For diverse user, they have a probability of 0.75 finding popular items as relevant. To evaluate recommendation performance, we use the averaged relevance of top-100 recommendations over all users (**AOA**) and group-wise performance that is calculated on each subgroup.

3.1.4 Results. As shown in Figure 1(a), we found a recommender trained with ERM leads to performance gap among the target subgroups, where niche users receive the worst performance. After 10 rounds, the relevance of recommendations drops from 0.3 to 0.09, while the overall performance keeps on the same level. Such observations raise concerns on the robustness of ERM-trained recommenders: certain user groups may receive decreasing performance consistently in long-term and get marginalized if we do not focus on the worst-case performance. It's worth mentioning that the current subgroups leads to rather balanced data splits (34%/34%/32% of total ratings). In other words, *predicting the preferences of niche users is more difficult than blockbluster users, not simply due to less training data in the niche user group.* We also experimented with unbalanced data splits and found even larger performance gaps. In contrast, in Figure 1(b), we demonstrate that DRO is a promising optimization framework to mitigate this issue, as it keeps the worst-case performance at a consistent level in the long run. In the next section, we elaborate how we leverage DRO for recommendation, with highlights on specific challenges and our proposed techniques that realize its full potentials.

3.2 Distributionally-robust Recommendations

In this section, we start by introducing the "Min-Max" objective of DRO as an alternative to the conventional objective of ERM. We highlight a specific challenge when attempting DRO for recommendations, namely, the loss variance issue due to extremely sparse data. To address the problem, we propose an extension to DRO and show the ease of implementation and flexibility of our method.

3.2.1 Distributional robust optimization. Consider the retrieval problem with a labeled dataset $(x, y) \sim D$ and a surrogate loss function $\mathcal{L}(\theta; (x, y))$, the ERM framework minimizes the expected loss over the empirical data distribution:

$$\min_{\theta \in \Theta} \{\mathbb{E}_{(x,y)\sim D}[\mathcal{L}(\theta; (x, y))]\} \tag{1}$$

Robustness to skewed sub-distributions is critical for recommenders, as in real-world systems the distributions of users and items are consistently shifting [31]. Distributional Robustness Optimization [15] offers an appealing angle of tackling the robust learning problem by optimizing the worst-case performance over any subset of the dataset from a probability distribution $\mathcal{P}$:

$$\min_{\theta \in \Theta} \{\sup_{\hat{D} \in \mathcal{P}} \mathbb{E}_{(x,y)\sim \hat{D}}[\mathcal{L}(\theta; (x, y))]\} \tag{2}$$

In theory, the worst-case optimization objective leads to a model that has robustness guarantees around the uncertain set $\mathcal{P}$, which is referred to as *distributional robustness*. The choice of $\mathcal{P}$ has a trade-off between the richness of the uncertainty set for robustness guarantees and the tractability of optimization problem (e.g. for deep models). Empirical study has shown that on a real-world applications, overly focusing on any specific subset is too conservative as it subjects to outliers/noisy data and leads to pessimistic performance [27]. Instead, a more applicable approach is to optimize for the uncertain set defined on mixtures of m subgroups with distributions $D_1, D_2, ..., D_m$, i.e., $\mathcal{P} := \{\sum_{g=1}^{m} w_g D_g, w \in \Delta_m\}$, where Δ_m is the $m-1$-dimensional probability simplex. DRO with group-aware training data (Group-DRO) minimize the worst-group loss:

$$\theta_{DRO} = \arg\min_{\theta \in \Theta} \{\max_{w_g \in w} \sum_{g=1}^{m} w_g \mathbb{E}_{(x,y)\sim D_g}[\mathcal{L}(\theta; (x, y))]\} \tag{3}$$

For recommendations, the group setting is desirable, as it has the potential to pose certain robustness guarantees on subgroups with important product or societal implications, such as user demographics, item genre, etc. To solve the minimax problem in Eq. 3, an online algorithm is proposed by Sagawa et al. [27]. On a high-level, a weight distribution w is updated through exponentiated gradient ascent with step size η, where higher masses are assigned to subgroups with higher losses. Model parameters θ are then updated with stochastic gradient descent with learning rate γ for each batch.

3.2.2 Reduce uncertainty in loss estimations. In practice, DRO is appealing to our problem as it can be easily applied to training deep models [15]. However, there are a few challenges specific to recommendation: (1) the output space in recommendation is much larger, making certain subgroups or sub-distributions even sparser. For example, recommending items from a corpus of millions comparing to selecting the next word from a fixed vocabulary of thousands for language modeling; (2) user feedback data is more skewed and long-tailed than the data used for image classification or language modeling. These unique challenges make it uncertain whether the success of DRO in other domains can be directly transferred to recommendation. For certain sparse subgroups, we suspect the weight update according to surrogate loss will be unstable due to large batch-to-batch loss variance (Algorithm 1, line 5). Empirically, we found when training with DRO, the loss variance for worst-case subgroups are indeed higher. Instead of re-sampling redundant data from the sparse subgroups, we propose a streaming algorithm to reduce uncertainty in loss estimations. The key idea is to keep streaming estimations of the empirical loss at iteration t for each subgroup g, in a way that is similar to SGD with learning rate α. We use $\tilde{\mathcal{L}}$ instead of the raw surrogate losses $\mathcal{L}$ to update w (Algorithm 1, line 6-7). A small α would result in more conservative updates as it is less affected by batches where sparse subgroups do not exist. Empirically, we also found small learning rates (e.g, $\alpha = 0.1$) result in better performance.

Algorithm 1 Optimization algorithm for Streaming-DRO.

```
1:  Input: Training iteration $T$, $m$ subgroups, initial of model parameters $\theta^{(0)}$, subgroup weights $w^{(0)}$, initial subgroup losses $\tilde{\mathcal{L}}_g^{(0)}$, step size $\eta$, learning rate $\gamma$, streaming learning rate $\alpha$.
2:  for $t = 1, ..., T$ do
3:      for $g = 1, ..., m$ do
4:          $x, y \sim D_g$
5:          $\mathcal{L}_g^{(t)} \leftarrow \mathcal{L}(\theta^{(t-1)}; (x, y))$
6:          $\tilde{\mathcal{L}}_g^{(t)} \leftarrow (1-\alpha)\tilde{\mathcal{L}}_g^{(t-1)} + \alpha\mathcal{L}_g^{(t)}$
7:          $w_g^{(t)} \leftarrow w_g^{(t-1)} \exp(\eta\tilde{\mathcal{L}}_g^{(t)})$
8:      end for
9:      for $g = 1, ..., m$ do
10:         $w_g^t \leftarrow w_g^{(t)}/\sum_g w_g^{(t)}$
11:         $\theta^{(t)} \leftarrow \theta^{(t-1)} - \gamma w_g^{(t)} \nabla\tilde{\mathcal{L}}_g^{(t)}$
12:     end for
13: end for
```

4 EXPERIMENTAL RESULTS

We conducted offline experiments on MovieLens-20M and Amazon Book Review [14, 22] to evaluated the effectiveness of optimizing worst-case performance with DRO and the proposed Streaming-DRO algorithm. After basic pre-processings of removing outliers and noisy data, we partition the users into three user groups similarly in Section 3.1.1. The resulting statistics of the datasets are shown in Table 2. For each user, we sort their rating history chronologically and split the rating history by ratios of 80%/10%/10% to form the train, validation and test sets.

We use the two-tower DNN model introduced in Section 3.1.2 as the backbone model. For the query and item DNNs, each DNN is a Multi-Layer Perceptrons (MLP) with ReLu activation except the last hidden layer. The query and item embeddings are obtained from L2-normalizing the last hidden layers' activations. We grid-searched the backbone model's hyper-parameters (the learning rate, softmax temperature (τ) and MLP architecture) on validation dataset based on *Recall@100*. For both datasets, we use MLPs with

	ML20M						*Amazon*					
Model	Blockbuster	Diverse	Niche	AOA	WORST	GAP	Blockbuster	Diverse	Niche	AOA	WORST	GAP
ERM	0.457	0.261	0.193	0.356	0.193	0.264	0.129	0.123	0.153	0.130	0.123	0.030
IPW	0.460	0.260	0.192	0.357	0.192	0.269	0.119	0.124	0.194	0.130	0.118	0.075
CB	0.460	0.266	0.199	0.360	0.199	0.261	0.131	0.130	0.175	0.136	0.128	0.047
DRO	0.443	0.282	0.218	0.358	0.218	**0.225**	0.132	0.151	0.160	0.142	0.132	0.028
S-DRO	0.451	0.288	0.222	**0.365**	**0.222**	0.229	0.139	0.152	0.161	**0.146**	**0.139**	**0.022**

Table 1: WORST, AOA (higher is better) and GAP (lower is better) performance measured by Recall@100.

Dataset	Users	Items	Ratings	User Splits	Data Splits
ML20M	138K	15K	20.0M	(54%,29%,17%)	(31%,36%,35%)
Amazon	137K	44K	5.6M	(52%,35%,13%)	(48%,41%,11%)

Table 2: An overview of the datasets. *ML20M* and *Amazon* refers to MovieLens-20M and Amazon Book respectively. User Splits and Data Split correspond to the order of *(Blockbuster, Diverse, Niche).*

hidden layers of [128, 64], temperature $\tau = 0.07$, and a batch size of 1024. We trained the model using Adagrad [9] with learning rate of 0.1. For each positive pair in a mini-batch, we uniformly sample 1024 items from the corpus as negatives in the softmax loss [29]. We compare with following learning frameworks for training the backbone recommender:

- *Empirical Risk Minimization (ERM)*: the objective is to optimize for the overall performance (Eq. 1).
- *Inverse Propensity Weighting (IPW)* [16]: training example are weighted by the inverse of propensities.
- *Class Balanced Loss (CB)* [8]: proposed for long-tailed image classification task, adjusted to recommendation task by adding the class-balanced term to the softmax loss.
- *Group-DRO (DRO)* [27]: Distributionally-robust Optimization framework with the training objective in Eq. 3.
- *Streaming-DRO (S-DRO)*: Group-DRO with streaming loss estimations (Algorithm 1).

To make a fair comparison, we fix the backbone model architecture with the set of hyper-parameters best optimized on the empirical validation distribution, i.e., favoring ERM. On top of the backbone model, we further tune hyper-parameters for DRO, e.g. step size η. As indicated in Algorithm 1, the step size η controls the updates of group weights. A larger η would result in faster accumulations of weights on subgroups with higher losses. However, we notice that if η is set too large (e.g., $\eta = 0.1$), the weight for the worst-case subgroup is over-allocated and leads to insufficient learning for other groups. We tune η tuned to the optimal range for each dataset. For evaluations, we focus on the worst-case performance over any subgroup measured by retrieval metric Recall@k (**WORST**). As auxiliary metrics, we show the averaged-over-all performance (**AOA**) and the gap between the best-case and worst-case performance (**GAP**). We report evaluation metrics based on the *full item corpus* to avoid potential bias using sampled metrics [17] and results are averaged across 5 runs[1].

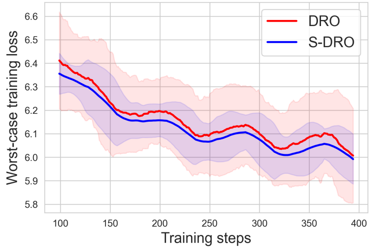

Figure 2: S-DRO significantly reduce loss variances and achieve improved worst-case loss than vanilla DRO.

Main findings. We found that DRO and S-DRO improve the worst-case performance on both datasets (Table 1). For example, comparing S-DRO to ERM, Recall@100 is improved by 14% on ML20M and 13% on the Amazon. Baselines such as IPW and CB do not consistently improve the worst-case metrics, and the performance gap is even larger on Amazon. We suspect that this is because IPW and CB weight training samples proportional to their group density during loss optimization, such a re-weighting strategy might fail when the task difficulty is not correlated with group density (for example, on Amazon Blockbuster users account for most of data splits, but they receive the worst performance). In contrast, DRO and S-DRO mitigate the gap between best-case and worst-case subgroups, while not harming or sometimes even improving the overall performance. To get insights on how S-DRO achieve better performance than vanilla DRO, Figure 2 illustrates worst-case loss of the subgroup under DRO and S-DRO. As expected, the batch-to-batch loss variance for DRO is large. S-DRO effectively reduces the loss variance for such sparse subgroup and outperforms DRO in improving the worst-case performance.

5 CONCLUSION

In this work, we propose to address the problem of group fairness in recommendations from the prospective of maximizing worst-case performance. We propose a streaming optimization for DRO which reduces the variance in loss estimations due to data sparsity in recommendations. Through experiments on large-scale datasets, we demonstrate the effectiveness of our proposed technique in achieving improved worst-case and overall performance at the same time. A future direction is to study alternative proxies for group hardness during loss optimization, especially when surrogate loss and end-to-end retrieval performance is not well-aligned [26]. Another direction is to develop adaptive weight-updating strategy, in the similar spirit of meta-learning [25]. We expect these new

[1] Our code is available at: https://github.com/google-research/google-research/tree/master/robust_retrieval

directions to achieve better sweet spots by improving the worst-case and overall performance for a wide range of applications.

REFERENCES

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 42–46.

[2] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, Bamshad Mobasher, and Edward Malthouse. 2021. User-centered Evaluation of Popularity Bias in Recommender Systems. In *Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization*. 119–129.

[3] 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.

[4] Alex Beutel, Ed H Chi, Zhiyuan Cheng, Hubert Pham, and John Anderson. 2017. Beyond globally optimal: Focused learning for improved recommendations. In *Proceedings of the 26th International Conference on World Wide Web*. 203–212.

[5] Jonathon Byrd and Zachary Lipton. 2019. What is the effect of importance weighting in deep learning?. In *International Conference on Machine Learning*. PMLR, 872–881.

[6] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt. 2018. How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 224–232.

[7] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[8] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9268–9277.

[9] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, 7 (2011).

[10] John Duchi and Hongseok Namkoong. 2018. Learning models with uniform performance via distributionally robust optimization. *arXiv preprint arXiv:1810.08750* (2018).

[11] Yonatan Geifman and Ran El-Yaniv. 2017. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941* (2017).

[12] Casper Hansen, Christian Hansen, Lucas Maystre, Rishabh Mehrotra, Brian Brost, Federico Tomasi, and Mounia Lalmas. 2020. Contextual and sequential user embeddings for large-scale music recommendation. In *Fourteenth ACM Conference on Recommender Systems*. 53–62.

[13] Moritz Hardt, Eric Price, and Nathan Srebro. 2016. Equality of opportunity in supervised learning. *arXiv preprint arXiv:1610.02413* (2016).

[14] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[15] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness Without Demographics in Repeated Loss Minimization. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 1929–1938.

[16] Herman Kahn and Andy W Marshall. 1953. Methods of reducing sample size in Monte Carlo computations. *Journal of the Operations Research Society of America* 1, 5 (1953), 263–278.

[17] Walid Krichene and Steffen Rendle. 2020. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1748–1757.

[18] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1073–1082.

[19] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 7, 1 (2003), 76–80.

[20] Augusto Q Macedo, Leandro B Marinho, and Rodrygo LT Santos. 2015. Context-aware event recommendation in event-based social networks. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 123–130.

[21] Hongseok Namkoong and John C Duchi. 2016. Stochastic Gradient Methods for Distributionally Robust Optimization with f-divergences.. In *NIPS*, Vol. 29. 2208–2216.

[22] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 188–197.

[23] Flavien Prost, Nithum Thain, and Tolga Bolukbasi. 2019. Debiasing embeddings for reduced gender bias in text classification. *arXiv preprint arXiv:1908.02810* (2019).

[24] John Rawls. 1958. Justice as fairness. *The philosophical review* 67, 2 (1958), 164–194.

[25] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*. PMLR, 4334–4343.

[26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[27] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731* (2019).

[28] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. 2020. Towards fairness in visual recognition: Effective strategies for bias mitigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8919–8928.

[29] Ji Yang, Xinyang Yi, Derek Zhiyuan Cheng, Lichan Hong, Yang Li, Simon Wang, Taibai Xu, and Ed H. Chi. 2020. Mixed Negative Sampling for Learning Two-tower Neural Networks in Recommendations.

[30] Sirui Yao, Yoni Halpern, Nithum Thain, Xuezhi Wang, Kang Lee, Flavien Prost, Ed H Chi, Jilin Chen, and Alex Beutel. 2021. Measuring Recommender System Effects with Simulated Users. *arXiv preprint arXiv:2101.04526* (2021).

[31] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed Chi. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 269–277.

[32] Yin Zhang, Derek Zhiyuan Cheng, Tiansheng Yao, Xinyang Yi, Lichan Hong, and Ed H Chi. 2020. A Model of Two Tales: Dual Transfer Learning Framework for Improved Long-tail Item Recommendation. *arXiv preprint arXiv:2010.15982* (2020).

[33] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. 2019. Recommending what video to watch next: a multitask ranking system. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 43–51.