Bypassing the Popularity Bias: Repurposing Models for Better Long-Tail Recommendation

Václav Blahut
vaclav.blahut@firma.seznam.cz
Seznam.cz, a.s.
Prague, Czech Republic

Karel Koupil
karel.koupil@firma.seznam.cz
Seznam.cz, a.s.
Prague, Czech Republic

Abstract

Recommender systems play a crucial role in shaping information we encounter online, whether on social media or when using content platforms, thereby influencing our beliefs, choices, and behaviours. Many recent works address the issue of fairness in recommender systems, typically focusing on topics like ensuring equal access to information and opportunities for all individual users or user groups, promoting diverse content to avoid filter bubbles and echo chambers, enhancing transparency and explainability, and adhering to ethical and sustainable practices.

In this work, we aim to achieve a more equitable distribution of exposure among publishers on an online content platform, with a particular focus on those who produce high quality, long-tail content that may be unfairly disadvantaged. We propose a novel approach of repurposing existing components of an industrial recommender system to deliver valuable exposure to underrepresented publishers while maintaining high recommendation quality. To demonstrate the efficiency of our proposal, we conduct large-scale online AB experiments, report results indicating desired outcomes and share several insights from long-term application of the approach in the production setting.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

Recommender systems, Fairness, Long-tail recommendation, Popularity bias, Inverse Retrieval

ACM Reference Format:
Václav Blahut and Karel Koupil. 2024. Bypassing the Popularity Bias: Repurposing Models for Better Long-Tail Recommendation. In *Proceedings of 7th FAccTRec Workshop on Responsible Recommendation at RecSys 2024 (FAccTRec@RecSys '24).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Our online content platform serves millions of daily active users and recommends content produced by many different publishers. Users are offered a personalized selection of online media content, consisting of news and entertainment articles, videos and podcasts, as illustrated in a Figure 1. The platform is privately owned and the majority of its revenue comes from serving online advertisements.

As the most visited content platform in our country, we make great demands on all publishers in terms of quality of the content

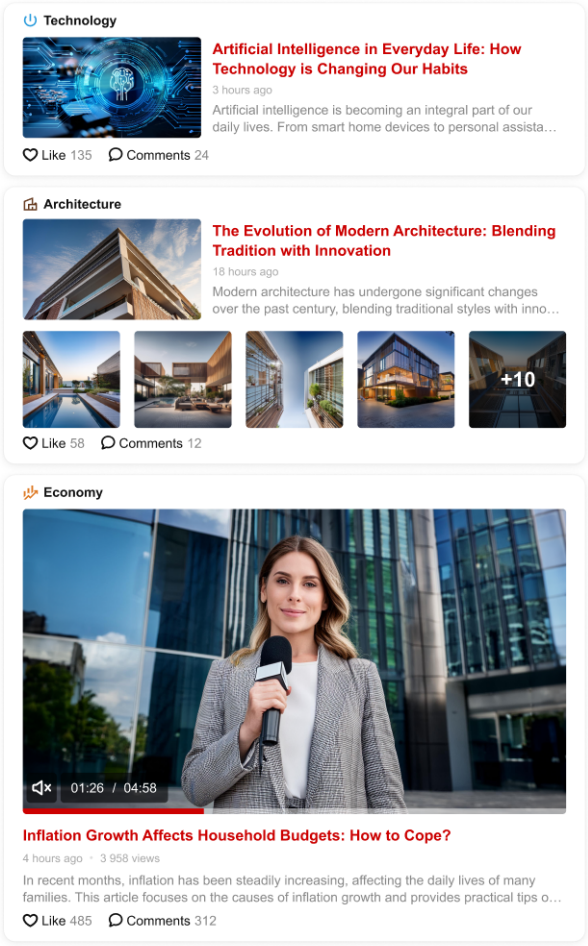

Figure 1: Illustration of our endless feed of content.

itself and impose rules for displaying an advertisement. Long-tail content [1] with high specificity and quality, designed for a smaller audience, is often expensive to produce and in our recommender system suffers from a popularity bias that leads to less attention due to fewer clicks and a lower click-through rate (CTR). As a result, more specific and less popular publishers may not thrive, and being involved in our platform may not be economically beneficial for

[1]The term *long-tail content* [16] refers to the items that are less popular compared to the most popular items. It depends on the shape of exposure distribution curve and often constitutes the majority of items, in contrast to the small number of very popular items.

FAccTRec@RecSys '24, October 14–18, 2024, Bari, Italy
2024. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

them in the long-term. This can be regarded as a systemic bias in our setting.

Recommending long-tail items is generally considered to be valuable to the users [4], as these are items that users are less likely to know about. Additionally, exposing users to these items can increase diversity and serendipity of a recommendation list, which surprises and satisfies the users because properly selected long-tail items can still be very relevant to them [1, 11].

A market that suffers from a popularity bias will lack opportunities to discover more obscure products and will be, by definition, dominated by a few large brands or well-known artists [6]. In our work, we try to mitigate this bias as well as strive to improve the recommendation of the long-tail content to users.

A typical recommender system searches for the best items for each user in terms of a defined objective. We decided to reverse this approach and formalize the problem as follows: *for each item of the selected publisher, we search for its best users*. We believe and further prove that this approach can mitigate the aforementioned systemic bias and provides sufficient revenue for selected long-tail publishers.

Since our recommender system recommends an item to a user and we want to promote a long-tail group of publishers, we need to create a mechanism that aligns our approach to the publisher level. This involves setting a minimum exposure, which is the amount of exposure each of the items produced by selected publishers should receive, that is considered sufficient and is constant across all items. In our experiments, exposure of an item is represented by the number of visible impressions, that is, the number of times a given item has been recommended to a user and visibly displayed on user's screen for defined period of time. To establish the value of minimum exposure, we take into account typical revenue of an advertisement that is displayed on a selected publisher's website, as well as average CTR of an item.

In summary, we make the following contributions:

- we propose a novel approach to recommending long-tail items,
- we employ several system-level fairness metrics,
- we conduct multiple online experiments, including an ablation study
- we investigate the effectiveness of the proposed approach in terms of fairness and performance metrics
- we incorporate the proposed method into our standard recommender system and report long-term experience

2 Related work

In order to measure the effect of the proposed approach on equitable resource distribution at our platform, system-level fairness metrics need to be defined. Lazovich et al. [10] present the distributional inequality metrics, a set of metrics originating from economics, and define desirable criteria to evaluate their ability to measure disparities in content exposure. Diaz et al. [7] define a set of metrics based on the principle of equal expected exposure, which takes into account the relevance of an item.

A common way of improving system-level fairness in recommender systems is via randomization. Diaz et al. [7] use Plackett-Luce sampling, which samples permutations based on the scores from the retrieval or the ranking model. Another method, called Rank Transposition, ignores the scores and randomly shuffles the original ranked list. Bower et al. [5] show that, in industrial settings, randomization only at one specific stage of recommendation process may not lead to improvements in fairness, and that it is necessary to treat fairness holistically, at all stages of possibly very complex recommender system. To do so, authors present Plackett-Luce sampling with Inverse Candidate Frequency Weights, which helps mitigate possible biases arising from retrieval stage of recommendation.

Maximum Inner Product Search (k-MIPS) is a well-studied problem from the area of collaborative filtering recommendation [12]. The idea of reversing the k-MIPS to efficiently find suitable users for an item was first proposed by Amagata and Hara [2]. The authors define the reverse k-MIPS as the problem of finding a set of k users with maximum inner product for a query item, explore existing exact and approximate search algorithms and propose their own search algorithm called Simpfer, which they further improve in [3].

3 Inverse Retrieval Model

Given an item, our goal is to retrieve an ordered set of users who are likely to find the item relevant and to consume it. To achieve that, we suggest repurposing the two-tower retrieval model [15] that is already trained and used in our main recommendation pipeline. In a standard setting of recommending items for a single user, the model is used to generate user and item embeddings. Single user embedding is then used as a query in an Approximate Nearest Neighbors (ANN) index of item embeddings to retrieve the top N most similar items. To tackle our problem of retrieving users for an item, we propose to invert this setting. Given a set of users, we use the model to generate an embedding for each of the users and store those embeddings in the ANN index. Then, we use the embedding of the given item to query this index and retrieve the top N most similar users. Note that any other model optimized for user-item embedding (dis-)similarity can be used as a replacement.

This Inverse Retrieval (InvR) process can be repeated for all selected long-tail items that did not receive enough exposure at given time. The detailed diagram of the pipeline is shown in Figure 2. By its nature, this approach fits better in offline recommendation setting, so all item-to-users candidates are periodically recomputed, transposed to user-to-items (single user can appear as a candidate for multiple items) and presented to the user once they use the service. Note that the number of users per item is a crucial hyperparameter that has to be carefully selected according to the specific context. The variables that affect the selection include but are not limited to the required minimum item exposure, the number of items and users, the frequency of candidate recomputation and the lifetime of an item, if any kind of item expiration is present.

3.1 Ordering items for single user

In cases when there are multiple items recommended for a single user, we need to decide the ordering of items presented to the user. The most apparent solution is to use the score provided by the ANN index, in our case dot product of the user and the item embedding. This approach, however, can be strongly biased by item popularity as more popular items often gain higher scores [13] regardless

of the actual relevance to the user, defeating the very purpose of recommending long-tail items. A more fairness-oriented solution is to randomly shuffle items. We argue that the most appropriate method is to order the items by the rank of a given user with respect to a given item. In other words, if user ranks high for the given item, compared to all other users according to the score, then this item should be presented to this user at a high position, ignoring the absolute value of the score. This way, the priority is to present the given item to the most promising users first.

3.2 Considerations and limitations

One may argue that an equivalent of item popularity bias may exist in the set of users, heavy users being ranked high in similar manner as popular items are in standard recommender system scenarios. Such problem may occur if the users were represented by their unique ID. In our system, it is naturally avoided by representing the users by their recent interaction history, truncated to a fixed length, so the rank of a user should not depend on their level of activity. Also, to limit the number of item candidates per user during the user-to-item transposition, we retrieve more user candidates for each item, sort all user-item candidate pairs by score or rank (depending on the variant) and in a single pass, we assign items to users until predefined item per user limit is reached, after which the user is skipped for further candidates. This way, the items are distributed among more users instead of accumulating on fewer heavy users and every item is guaranteed to be assigned exactly to the required number of users.

Another challenge is the quality of cold start item embeddings. Since the ID-based item embeddings in the two-tower retrieval model are trained on user-item interactions, the representations of new items with no interactions are virtually random and useless for finding relevant users. We employ standard randomization solution to the item cold start by randomly inserting new items into recommendation slates for a pre-defined period of time or until sufficient number of interactions have been gathered.

4 Setup

4.1 Recommender system description

Our main recommendation pipeline follows the 4-stage framework as described by Higley et al. [9]. The retrieval and ranking stages utilize models from TensorFlow Recommenders library[2], specifically the two-tower model for retrieval and Deep and Cross Network V2[14] for the ranking stage. Both retrieval and ranking models are trained incrementally every 5 minutes. An endless feed of recommended items consists of 20-item slates, incrementally generated online as user scrolls through the feed. The system serves millions of daily active users, handling thousands of requests per second with the latency limits in order of lower hundreds of milliseconds.

The retrieval model, which is also repurposed for InvR, uses hashed ID-based item embeddings. An embedding of user is determined as an average pooling of item embeddings present in given user's history. Clicked items are used as positive examples and visible but not clicked items as negative. The dimensionality of embeddings is 128, the batch size is 128. We train the model for 10 epochs with learning rate 0.1 and AdaGrad optimizer. All retrieval model hyperparameters were optimized previously during series of experiments and have not been further optimized for the InvR usage. Both item index for retrieval and user index for InvR use ScANN[8] library.

[2]https://www.tensorflow.org/recommenders/

4.2 Incorporating InvR items in recommendation

The offline InvR pipeline is executed every 60 minutes. During the online recommendation, the pre-generated InvR candidate items are inserted into the slate after the ranking stage and most of the business logic, so the recommended items are isolated from all possible biases arising from the main recommender, as described in Section 1. We argue that the main bias our approach bypasses is the popularity bias. The only remaining business logic applied to InvR-recommended items is simple deduplication. We dedicate up to 3 positions in a specified range of each 20-item slate for InvR recommendation. If an item has been visibly displayed to a user twice, we exclude it for all future visits of the same user. Once an item reaches the minimum exposure, it is no longer supported by the InvR mechanism, though it may still be organically recommended by the main recommender pipeline.

4.3 Publisher selection

We select those publishers that suffer from low attention/revenues due to the popularity bias as described in Section 1. For this purpose, we design several business and performance criteria that have to

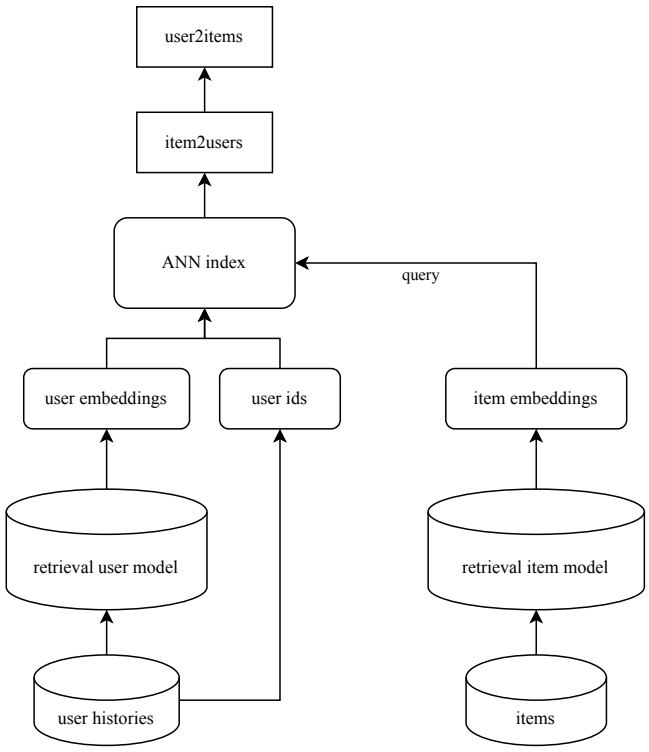

Figure 2: InvR pipeline diagram in production conditions

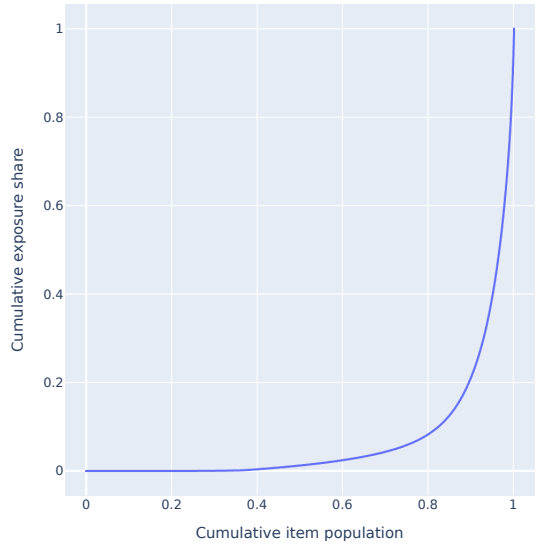

Figure 3: Real-world example of Lorenz curve.

be met in a 6-months window: low total revenue per month, low average number of visible impressions across publisher's items, a low number of clicks in total per month and finally low revenue per each item on average. Besides previous conditions, publishers have to belong to a group of "original niche" which is characterized by a high level of content production quality and is specialized utterly in one content topic.

4.4 User selection

We want to show long-tail content recommendations to the users with stable identity, i.e. those who have sufficiently long history and who consent with personalization. Since the success of our proposal heavily depends on the user actually visiting the platform in the near future so that the InvR-generated item candidates can be presented to them, we select only the users who visited the platform frequently enough in recent past.

4.5 A/B testing

In order to assess the performance of our recommender system, we perform randomized A/B tests on live traffic. For each of the tested variants, we include A/A variant. The size of each variant as well as test duration are chosen empirically w.r.t. statistical significance. In the context of this work, it would be interesting to perform AB testing in terms of items. However, due to the production environment, the unit of randomization in our AB testing system is the user.

5 Evaluation and discussion

5.1 Metrics

To evaluate the impact of the proposed method on the exposure of long-tail items, we employ these system-level item-wise fairness metrics:

a) **bottom 50 percent share (B50PS)** is a metric derived from Lorenz curve, inspired by top X percent share metric presented by Lazovich et al.[10], modified to capture changes in the long-tail items region of the Lorenz curve. It shows the portion of exposure received by the least popular half of the whole item set. While the choice of 50th percentile is arbitrary and depends on the skewness of the Lorenz curve, it is a strict lower-bound of what can be considered a long-tail of a distribution, since even in the edge case of uniform distribution, it is equal to the percentage of equal share. Therefore, in many real-world scenarios such as can be seen in Figure 3, where the Lorenz curve is heavily skewed, the bottom half of the item set can safely be considered a long-tail.

b) **percentage of sufficiently exposed items (PSEI)** is the percentage of the treated items that have received more than the pre-defined minimum exposure.

The two selected metrics above are meant to be complementary – B50PS focuses on all items while PSEI captures changes only for InvR-treated items.

The overall amount of users and their attention is beyond our control, making the overall exposure relatively fixed. Consequently, promoting a subset of items will inevitably come at the expense of the rest of the items, especially the most popular ones. To illustrate the impact on the most popular items exposition, we report the aforementioned **top 1 percent share (T1PS)**[10].

To compare the InvR variants described in Section 5.2 with each other and with the ablation study baseline in terms of relevance, we report average **click-through rate (CTR)** and average number of **clicks**, both per user, filtered to InvR-generated recommendations only. All reported metric changes are in relative scale.

5.2 Variants

The **Baseline** is our recommender system as described in Section 4.1, without any usage of the InvR. We conduct an ablation study to investigate the performance gain of using the model in InvR pipeline by including **Random** variant that acts as a baseline for InvR. In this variant, the same conditions apply (set of items and users, periodical offline batch computation and incorporation into the slate), but instead of using the model for user selection, we pick users for an item randomly. Finally, we use three InvR variants described in Section 3.1 to show differences in the item orderings. **InvR Random** orders the item candidates for a user randomly, **InvR Score** orders the candidates by the absolute value of the score and **InvR User rank** by the user rank w.r.t. the item.

5.3 Overall results

See Table 1 for the experimental result. The metric bottom 50 percent share, B50PS shows that while all variants have boosted the exposure of long-tail items, the InvR User rank variant surpassed others by a large margin. This metric is computed on all items, not only those produced by selected publishers. As for the percentage

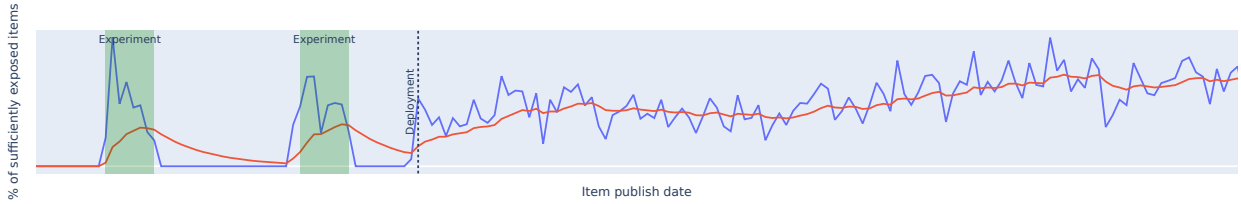

Figure 4: Percentage of sufficiently exposed items (PSEI) over time.

Table 1: Experimental results

Variant	B50PS	PSEI	T1PS	CTR (InvR)	Clicks (InvR)
Baseline	0 %	0 %	0 %	-	-
Random	+9.2 %	+181 %	-1.0 %	0 %	0 %
InvR Random	+9 %	+43 %	-0.5 %	+271 %	+106 %
InvR Score	+9.9 %	+41 %	-0.9 %	+261 %	+103 %
InvR User rank	+33.3 %	+45 %	-1.6 %	+300 %	+120 %

of sufficiently exposed items metric, PSEI, where only items from selected publishers are considered, we can see that the Random variant was outstandingly successful. This comes as no surprise, because - in the Random variant - items are uniformly distributed between all users, leading to maximal exposure, while all other InvR methods focus on a much smaller set of relevant users. The cost of the high exposure is a severely low CTR and Clicks when compared to other InvR variants, as a long-tail item delivered to a random user will most likely be irrelevant to the user. The discrepancies between B50PS and PSEI for Random and InvR variants are consequent upon the fact that each metric is measured over a different set of items (see the mention of the complementarity of these two metrics in Section 5.1).

In terms of the impact on the most popular items exposition captured by top 1 percent share, T1PS, where a reduction is (from a fairness point of view) considered positive change, the InvR User rank variant significantly outperformed others.

Focusing on the CTR and Clicks, we found it surprising that the InvR Score variant did not surpass the others thanks to the popularity bias in scores, which remains unexplained. We conclude that the InvR User rank item ordering method is consistently the right approach.

Since InvR effectively pushes down other, possibly more popular and clickable items, it is natural that relevance-focused KPIs may suffer. To illustrate the cost of deploying the winning InvR User rank variant to production, we report 1.04 % drop in overall user average CTR and 1.67 % decrease in overall average number of clicks per user.

5.4 Further experiments

After the main experiment where described variants were tested, a few more experiments followed. For example, we examined the impact of position bias on item exposure by modifying the range of positions where InvR items are inserted into slate, confirming that by placing item on higher position, the exposure increases significantly, with significantly negative impact on KPIs. We also experimented with different user counts per item, in which case we discovered that the increase in exposure is sublinear to the increase of users per item. Other experiments tested various user-item allocation algorithms that assign candidates more evenly in edge cases (leading to increase in exposure of the least popular items), inclusion of unregistered users (no significant impact) and loosening the criteria for publisher selection (lower overall exposure per item).

5.5 Mentionable side-effects and long-term experience

Apart from the results above, other interesting details surfaced when we analysed the experiment. The main recommender pipeline secondarily benefited in terms of diversity (measured by total number of unique recommended items) thanks to the fact that the InvR-generated data was used for training the main model. The number of unique items recommended by the main pipeline in InvR variants increased by 18 % when compared to Baseline.

Finally, the InvR User rank variant was deployed as our new production baseline. Over the course of the following months, we observed interesting behaviors in the system. While there was an immediate increase in PSEI metric after the deployment of InvR, the PSEI continued to grow until it reached similar values as the Random variant did in our main experiment, while maintaining good performance in terms of KPIs. The increase of global PSEI after deployment, as well as during the experiments, can be seen in Figure 4. The line in blue is the actual daily value and the red line is exponentially weighted moving average with $\alpha = 0.125$.

We also noticed that on long-term average, only about 40 % of minimum exposure is generated by InvR itself until it gets shut off for given item due to reaching the minimum exposure, suggesting that InvR helps "kick-start" the item and the main recommender takes care of the rest.

6 Conclusion and future work

In this paper, we proposed a novel way of repurposing part of industrial recommender system to tackle the problem of long-tail content recommendation, which we argue is mainly caused by popularity bias. After describing the method and all its variants, we reported the results of online experiments, showing significant increase in exposure of selected long-tail publishers, accompanied

by impacts on the KPIs. We discussed some further observations and side-effects as well as long-terms experience after successful deployment to production.

In conclusion, proposed approach meets the needs of two primary stakeholders: it supplies publishers with sufficient and relevant traffic and delivers well-personalized recommendations to users, resulting in a positive long-term impact on the platform stability from business perspective.

There are multiple directions of possible future work. The InvR may greatly benefit from a more sophisticated solution of item cold start, applicable to the two-tower retrieval model, which would be to replace item ID-based embedding with pre-trained content-based one, further trained on user-item interactions. We may also perform the model hyperparameter optimization for the InvR use case or experiment with completely different user and item embedding models. To rule out the influence of selection of long-tail publishers, we may conduct an experiment where all items receive the InvR treatment. We would also like to focus more on the impact of our solution on the user experience and user-centred fairness.

Acknowledgments

We thank Radek Tomšů, Vít Líbal, Milan Vancl, Jaroslav Kuchař and Jan Vršovský for their extensive feedback on this paper. We thank Josef Florian for providing important details of the production system.

References

[1] Gediminas Adomavicius and YoungOk Kwon. 2012. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering* 24, 5 (2012), 896–911. https://doi.org/10.1109/TKDE.2011.15

[2] Daichi Amagata and Takahiro Hara. 2021. Reverse Maximum Inner Product Search: How to efficiently find users who would like to buy my item? arXiv:2110.07131 [cs.DB] https://arxiv.org/abs/2110.07131

[3] Daichi Amagata and Takahiro Hara. 2023. Reverse Maximum Inner Product Search: Formulation, Algorithms, and Analysis. *ACM Trans. Web* 17, 4, Article 26 (jul 2023), 23 pages. https://doi.org/10.1145/3587215

[4] Chris Anderson. 2006. *The long tail : why the future of business is selling less of more.* Hyperion.

[5] Amanda Bower, Kristian Lum, Tomo Lazovich, Kyra Yee, and Luca Belli. 2022. Random Isn't Always Fair: Candidate Set Imbalance and Exposure Inequality in Recommender Systems. arXiv:2209.05000 [cs.IR] https://arxiv.org/abs/2209.05000

[6] Òscar Celma and Pedro Cano. 2008. From hits to niches? or how popular artists can bias music recommendation and discovery. In *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition* (Las Vegas, Nevada) *(NETFLIX '08)*. Association for Computing Machinery, New York, NY, USA, Article 5, 8 pages. https://doi.org/10.1145/1722149.1722154

[7] Fernando Diaz, Bhaskar Mitra, Michael D. Ekstrand, Asia J. Biega, and Ben Carterette. 2020. Evaluating Stochastic Rankings with Expected Exposure. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 275–284. https://doi.org/10.1145/3340531.3411962

[8] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating Large-Scale Inference with Anisotropic Vector Quantization. In *International Conference on Machine Learning*. https://arxiv.org/abs/1908.10396

[9] Karl Higley, Even Oldridge, Ronay Ak, Sara Rabhi, and Gabriel de Souza Pereira Moreira. 2022. Building and Deploying a Multi-Stage Recommender System with Merlin. In *Proceedings of the 16th ACM Conference on Recommender Systems* (Seattle, WA, USA) *(RecSys '22)*. Association for Computing Machinery, New York, NY, USA, 632–635. https://doi.org/10.1145/3523227.3551468

[10] Tomo Lazovich, Luca Belli, Aaron Gonzales, Amanda Bower, Uthaipon Tantipongpipat, Kristian Lum, Ferenc Huszár, and Rumman Chowdhury. 2022. Measuring disparate outcomes of content recommendation algorithms with distributional inequality metrics. *Patterns* 3, 8 (Aug. 2022), 100568. https://doi.org/10.1016/j.patter.2022.100568

[11] Jingjing Li, Ke Lu, Zi Huang, and Heng Tao Shen. 2017. Two Birds One Stone: On both Cold-Start and Long-Tail Recommendation. In *Proceedings of the 25th ACM International Conference on Multimedia* (Mountain View, California, USA) *(MM '17)*. Association for Computing Machinery, New York, NY, USA, 898–906. https://doi.org/10.1145/3123266.3123316

[12] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. arXiv:2005.09683 [cs.IR] https://arxiv.org/abs/2005.09683

[13] Wondo Rhee, Sung Min Cho, and Bongwon Suh. 2022. Countering Popularity Bias by Regularizing Score Differences. In *Proceedings of the 16th ACM Conference on Recommender Systems* (Seattle, WA, USA) *(RecSys '22)*. Association for Computing Machinery, New York, NY, USA, 145–155. https://doi.org/10.1145/3523227.3546757

[14] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 1785–1797. https://doi.org/10.1145/3442381.3450078

[15] 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.

[16] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. 2012. Challenging the Long Tail Recommendation. arXiv:1205.6700 [cs.DB] https://arxiv.org/abs/1205.6700