Controlling Popularity Bias in Learning to Rank Recommendation

Himan Abdollahpouri, Robin Burke, Bamshad Mobasher
DePaul University
School of Computing
College of Computing and Digital Media
habdolla@depaul.edu, rburke@cs.depaul.edu, mobasher@cs.depaul.edu

Abstract

Many recommendation algorithms suffer from popularity bias in their output: popular items are recommended frequently and less popular ones rarely, if at all. However, less popular, long-tail items are precisely those that are desirable for increased user satisfaction. In this paper, we introduce a flexible regularization-based framework to enhance the long-tail coverage of recommendation lists in a learning-to-rank algorithm. We show that regularization provides a tunable mechanism for controlling the trade-off between accuracy and coverage. Moreover, the experimental results using two data sets show that it is possible to achieve higher coverage of long tail items without substantial sacrifice of ranking performance.

1 Introduction

Recommender systems have an important role in e-commerce and information sites, helping users find new items. One obstacle to effectiveness of recommenders is in the problem of popularity bias: collaborative filtering recommenders typically emphasize popular items (those with more ratings) much more than other "long-tail" items [11]. Although popular items are often good recommendations, it is important that recommender systems achieve a balance between popular and less-popular items.

Figure 1 illustrates the long-tail phenomenon in the well-known MovieLens 1M dataset [6]. The y axis represents the number of ratings per item and the x axis shows the product rank. The first vertical line separates the top 20% of items by popularity – these items cumulatively have many more ratings than the 80% long tail items to the right. These "short head" items are the very popular blockbuster movies that garner much viewer attention. Similar distributions can be found in books, music, and other consumer taste domains.

The second vertical line divides the long tail of the distribution into two parts. The first part we call the *medium tail*: these items are accessible to collaborative recommendation, even though recommendation algorithms often do not produce them. Beyond this point, items receive so few ratings that meaningful cross-user comparison of their ratings becomes noisy and unreliable. For these items, the *distant long tail* or cold-start items, content-based and hybrid recommendation techniques must be employed. Our work in this paper is concerned with collaborative recommendation and therefore focuses on the medium tail.

In this paper, we present a flexible framework for fairness-aware optimization built on the top of a learning to rank algorithm. When applied to long-tail items, this approach enables the system designer to tune the application to achieve a particular tradeoff between ranking accuracy and the inclusion of long-tail items. However, the framework is sufficiently general that it can be used to control recommendation bias for or against any group of items.

1.1 Related Work

Recommending serendipitous items from the long tail are generally considered to be valuable to users [14, 1], as these are items that users are less likely to know about. Brynjolfsson and his colleagues showed that 30-40% of Amazon book sales are represented by titles that would not normally be found in brick-and-mortar stores [2]. Access to long-tail items is a strong driver for e-commerce growth: the consumer surplus created by providing access to these less-known book titles is estimated at more than seven to ten times the value consumers receive from access to lower prices online.

Long-tail items are also important for generating a fuller understanding of users' preferences. Systems that use active learning to explore each user's profile will typically need to present more long tail items because these are the ones that the user is less likely to have rated, and where user's preferences are more likely to be diverse [10, 13].

Finally, long-tail recommendation can also be understood as a social good. A market that suffers from 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 [4]. Such a market will be more homogeneous and offer fewer opportunities for innovation and creativity.

The idea of the long-tail of item popularity and its impact on recommendation quality has been explored by some researchers [2, 11]. In those works, authors tried to improve the performance of the recommender system in terms of accuracy and precision, given the long-tail in the ratings. Our work, instead, focuses on reducing popularity bias and balancing the presentation of items across the popularity distribution.

2 Learning to Rank

We focus on matrix factorization approaches to recommendation in which the training phase involves learning a low rank $n \times k$ latent user matrix P and a low-rank $m \times k$ latent item matrix Q, such that the estimated rating $\hat{r}_{ui}$ can be expressed as $\hat{r}_{ui} = p_u^T q_i$ where p_u^T is the u^{th} row of P, q_i^T is the i^{th} row of Q, and k is the chosen dimension of the latent space. P and Q are learned through the minimisation of an accuracy-based objective [8].

Since we are interested in ranking rather than rating prediction, we focus on the learning

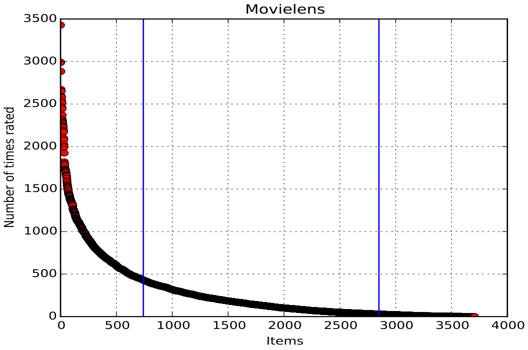

Figure 1: The long-tail of item popularity.

to rank objective function proposed originally in [7] and further developed in [15]:[1]

$$\sum_{u\in U}\sum_{i\in I} c_{ui} \sum_{j\in I} s_j[(\hat{r}_{ui} - \hat{r}_{uj}) - (r_{ui} - r_{uj})]^2 \tag{1}$$

The role of c_{ui} is to select user-item pairs corresponding to positive feedback from all possible pairs. We consider the implicit feedback case in which $c_{ui} = 0$ if $r_{ui} = 0$, and 1 otherwise. s_j is an importance weighting for item j. In this work, because we are trying to reduce the influence of popular items, we use uniform importance weighting $s_j = 1$, for all j.

3 Fairness-aware Regularization

This work was inspired by recent research by Wasilewski and Hurley [16] in which regularization is used to enhance recommendation diversity in a learning to rank setting, specifically using the RankALS algorithm [15]. The authors calculate a pair-wise dissimilarity matrix D for all pairs of items, demonstrate that intra-list diversity (ILD) can be represented in terms of this matrix, and then experiment with different ways that increased ILD can be made into a optimization objective.

Following a similar approach, we explore the use of regularization to control the popularity bias of a recommender system. We start with an optimisation objective of the form:

$$\min_{P,Q} acc(P,Q) + \lambda reg(P,Q) \tag{2}$$

where $acc(.)$ is the accuracy objective, $reg(.)$ is a regularisation term, and λ is a coefficient for controlling the effect of regularizer.

Our goal therefore is to identify a regularization component of the objective that will be minimized when the distribution of recommendations is fair. As an initial assumption, we define a *fair recommendation list* as one that achieves a 50/50 balance between long-tail and short-head items. We intend to generalize this objective in future work.

Wasileski and Hurley start from a dissimilarity matrix D, which contains all pair-wise dissimilarities between items. The regularizer is computed from D in such a way that it pushes the optimization solution towards recommendation lists that balance ranking accuracy with intra-list diversity (ILD).

In our case, we define two sets of items Φ and $\sim\Phi$ corresponding to the short-head and medium-tail items, and define a co-membership matrix D, over these sets. For any pair of items i and j, $d(i,j) = 1$ if i and j are in the same set and 0 otherwise.[2]

Our equivalent for intra-list distance is a measure of fairness for the two sets of items in a given recommendation list L_u. We define intra-list binary unfairness (ILBU) as the average value of $d(i,j)$ across all pairs of items i,j.

$$ILBU(L_u) = \frac{1}{N(N-1)} \sum_{i,j\in L_u} d(i,j) \tag{3}$$

[1] A number of such objectives have been proposed in the literature and the regularisation method we propose here could be incorporated with any such pair-wise objective.

[2] We exclude distant long-tail items here. Extending this approach to content-collaborative hybrids is a topic for future work.

where N is the number of items in the recommendation list. The fairest list is one that contains equal numbers of items from each set, which can be easily seen: if we take a balanced distribution and move one item to the other set, there will be more non-zero pairs and therefore a greater sum over all pairs. Therefore, in this case unlike in [16], the regularizer should be minimized using a positive λ as lower values correspond to an evenly-balanced list.

A more readily interpretable but closely related metric of success we will use for evaluation is the Average Percentage of Tail items (APT) in the recommendation lists, which we define as follows:

$$APT = \frac{1}{|U_t|} \sum_{u \in U_t} \frac{|\{i, i \in (L(u) \cap \sim\Phi)\}|}{|L(u)|} \tag{4}$$

where $|U_t|$ is the number of users in the test set and $L(u)$ is the recommendation list for user u. This measure tells us what percentage of items in users' recommendation lists belongs to the medium tail set.

Note that we do not in our definition of fairness require that each item have an equal chance of being recommended. There are many fewer short-head items and we are, essentially, allocating half of the recommendation list to them.[3] The 50/50 heuristic is a starting point for exploring the effectiveness of our regularization-based technique.

As our ILBU value has the same form as the ILD measure used in [16], we can take advantage of their experience in deriving various regularization terms from it. We chose the LapDQ regularizer, which performed well in Wasilewski and Hurley's experiments: we plan to examine additional regularizers in future work. LapDQ is based on the Laplacian L_D of the D matrix and has the form $tr(Q^T L_D Q)$, where tr is the trace of the matrix. This version of the regularizer only influences the item factors and therefore the p step of the RankALS algorithm, which computes the user factors, is unchanged.

For the details of the derivation of the regularizer, readers are referred to [16]. One way to understand the function of $LapDQ$ is to note that our D matrix is block-structured, and has the same structure as the adjacency matrix for a graph consisting of two separate components of fully interconnected items. For such a graph, the eigenvector corresponding to the largest eigenvalue represents the partition of the graph. In our case, we already know what this partition is – it is the division of the nodes into short-head and medium-tail items. The trace of $Q^T L_D Q$ is the product of each latent factor with the Laplacian with the original Q vectors. If the item factors most resemble the partitioning eigenvector – that is, they contain elements from both partitions in opposite signs, then this vector product, and therefore the matrix trace, is minimized. The regularizer therefore pushes each item factor to span both partitions.

4 Experimental results

We used two datasets in our experiments. The first one is the well-known Movielens 1M dataset that contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000 [6]. The second dataset is the Epinions

[3] Note that it would be possible to adjust the balance between long-tail and short-head items by creating a smaller penalty for larger numbers of short-head items. These possibilities we leave for future work.

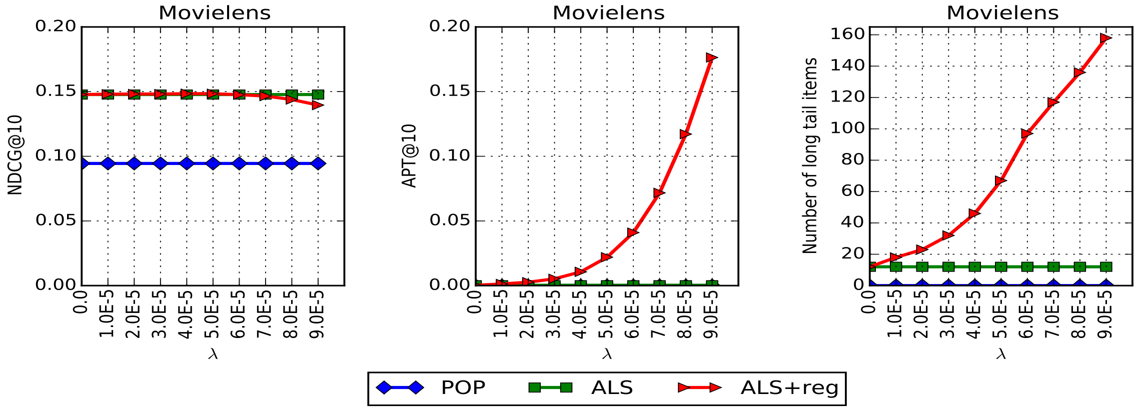

Figure 2: The nDCG, APT, and total long-tail coverage tradeoff (Movielens)

dataset which is gathered from a consumers opinion site where users can review items (such as cars, books, movies, and software) and assign them numeric ratings in the range 1 (min) to 5 (max) [9]. This dataset has total number of 664,824 ratings given by 40,163 users to 139,736 items. In Movielens, each user has a minimum of 20 ratings but in Epinions, there are many users with only a single rated item.

Our examination of the data showed that users with longer profiles were much more likely to have long-tail items in their profiles. From each dataset, we removed users who had less than 30 ratings. The retained users were those likely to have rated enough long-tail items so that our objective could be evaluated in a train / test scenario. We also removed distant long-tail items from each dataset, using a limit of 30 ratings. This has a very significant impact on the Epinions dataset, as there is a huge distant part of the tail containing many cold-start products with only one rating.

After the removal of short-profile users and distant long tail items, the Movielens dataset has 5,289 users who rated 2,836 movies with a total number of 972,471 ratings, a reduction of about 3%. Applying the same criteria to the Epinions dataset decreases the data to 157,887 ratings given by 5,383 users to 3,423 items, a reduction of more than 75%.

For the purposes of this experiment, we define the short-head of the distribution (the set Φ) to be the top 20% of the items. For Movielens, the short-head items were those with more than 400 ratings. In Epinions, a short-head item need only to have more than 46 ratings – which demonstrates the more skewed nature of ratings in the Epinions data.

We used Librec [5], an open source Java framework for recommender system implementation, modified by the inclusion of our regularized version of RankALS. We used a random split of 80% of the data for training and the remaining 20% for testing. For each user in the test set, we recommended a list of 10 items using three different algorithms: most popular(Pop), standard ALS (ALS) and our fairness-aware regularized ALS ($ALS + Reg$).

We evaluated the results for ranking performance in terms of normalized Discounted Cumulative Gain, nDCG@10. For the fairness-aware aspect of the algorithm, we evaluated the system based on the APT measure, the proportion of medium-tail items in the average recommendation list, and we also measured medium-tail coverage, the total number of items from the medium-tail set that are recommended to any user in the test data. This is a

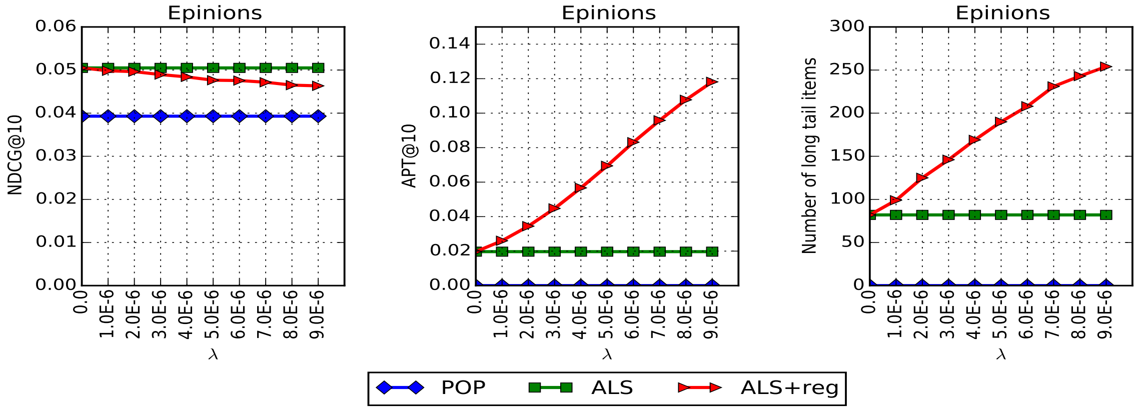

Figure 3: The nDCG, APT, and total long-tail coverage tradeoff (Epinions)

holistic view of how well the system is avoiding popularity bias.

Figure 2 shows the trade-off between nDCG@10, the medium-tail proportion, and the total number of medium-tail items covered in the recommendations, evaluated for different values of λ for the Movielens dataset. As expected, increased weight for the regularizer is associated with more tail items in each list, with increased total medium-tail coverage and with lower ranking performance. The APT measure approaches 20% for the highest lambda values – corresponding to 2 medium-tail items in each list on average. Without the fairness-aware approach, these lists have only short-head items. Notably, there is a fairly substantial region in which coverage increases with minimal measurable nDCG loss.

Full cross-validation results were not available as of this writing, but partial results show very little variation from the single fold results reported here. On the Movielens dataset at λ =7.0E-5, which seems to be an optimal point in terms of nDCG and medium-tail coverage tradeoff, we see a statistically-significant increase in average medium-tail coverage: 5.6 to 114 ($p <$ 1.0E-5) with statistically-insignificant loss of nDCG: 0.1449 to 0.1441 ($p > 0.25$).

Figure 3 shows a similar pattern for the Epinions dataset. The nDCG performance drops a little more noticeably in this dataset, but there is still a region that has increased medium-tail coverage with minor ranking loss. At $lambda$ =2.0E-6, the increase in average medium-tail coverage is smaller, but still significant ($p <$ 1E-4) and the difference in nDCG is not significant at $p < 0.05$, actual p-value = 0.066.

What we see in these results is similar to the findings of [16], namely that a regularization term incorporating list-wise properties of interest can be incorporated successfully into a learning to rank framework. In Wasilewski and Hurley's work, this was a similarity-based term that enabled them to enhance list-wise diversity; in our case, it is a group-membership-based term that enables us to enhance list-wise coverage of medium tail items.

Note in particular that because this loss function is pair-wise and local to individual recommendation lists, it does not aim to increase catalog coverage in the medium tail directly. It is possible, in a pathological case, to increase list-wise coverage of medium tail items (i.e. low ILBU / high APT) by providing the same items to all users. This minimizes the objective without enhancing medium-tail coverage. Our results show that, for these datasets the optimization of our local objective also increases overall coverage of medium-tail items across the test data.

5 Conclusion and Future Work

Long-tail items are an important key to practical success for recommender systems. Since short-head items are likely to be well known among many users, the ability to recommend items outside of this band of popularity will determine if a recommender can introduce users to new products and experiences. Yet, it is well-known that recommendation algorithms have strong biases towards popular items.

In this paper, we have concentrated on the medium tail – that part of the long tail for which there is sufficient data for collaborative filtering to be effective. Within this set of items, we have shown that it is possible to model the tradeoff between long-tail catalog coverage and ranking accuracy as a multi-objective optimization problem. Our regularization framework, built on related work in diversity optimization, provides a mechanism for tuning the RankALS algorithm to achieve a desired point in this tradeoff space. This fairness-aware regularization approach can, in principle, be applied in any learning algorithm in which the objective is formulated as a pair-wise function of items, such as Bayesian Personalized Ranking [12]. We plan to explore this possibility in future work.

This work suggests a number of other possible extensions. We have defined our objective as a balanced 50/50 presentation of short-head and medium-tail items. However, in some applications, this may be impractical or undesirable. In our future work, we will examine how the co-membership matrix D may be manipulated to target different levels of long-tail item presence (80/20, for example).

Our fairness criterion was formulated relative to two fixed sets. One could imagine generalizing this idea to a situation in which fairness is defined in other ways. For example, in [3], the authors discuss a scenario which fairness across multiple product suppliers is sought. In our future work, we will examine if our regularization approach can be extended to encompass this situation as well.

The presence of the distant long tail – items that cannot be integrated into a collaborative filtering framework – suggests the need for a hybrid strategy that incorporates content-based recommendation as well. Content-based recommendation does not depend on peer evaluations and is immune to popularity bias, but may be less effective when content data is noisy or incomplete. Our fairness-aware approach points the way to novel hybrids sensitive to the shape of the popularity distribution.

6 Acknowledgments

This work is supported in part by the National Science Foundation under grant IIS-1423368.

References

[1] Chris Anderson. *The long tail: Why the future of business is selling more for less.* Hyperion, 2006.

[2] Erik Brynjolfsson, Yu Jeffrey Hu, and Michael D Smith. “From niches to riches: Anatomy of the long tail”. In: *Sloan Management Review* (2006), pp. 67–71.

[3] Robin D. Burke et al. “Towards Multi-Stakeholder Utility Evaluation of Recommender Systems”. In: *Workshop on Surprise, Opposition, and Obstruction in Adaptive and Personalized Systems, UMAP 2016*. 2016.

[4] Òscar Celma and Pedro Cano. “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*. ACM. 2008, p. 5.

[5] Guibing Guo et al. “LibRec: A Java Library for Recommender Systems.” In: *UMAP Workshops*. 2015.

[6] F Maxwell Harper and Joseph A Konstan. “The MovieLens Datasets: History and Context”. In: *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5.4 (2015), p. 19.

[7] Michael Jahrer and Andreas Töscher. “Collaborative Filtering Ensemble.” In: *KDD Cup*. 2012, pp. 61–74.

[8] Yehuda Koren and Robert Bell. “Advances in collaborative filtering”. In: *Recommender systems handbook*. Springer, 2015, pp. 77–118.

[9] Paolo Massa and Paolo Avesani. “Trust-aware recommender systems”. In: *Proceedings of the 2007 ACM conference on Recommender systems*. ACM. 2007, pp. 17–24.

[10] Tien T Nguyen et al. “Exploring the filter bubble: the effect of using recommender systems on content diversity”. In: *Proceedings of the 23rd international conference on World wide web*. ACM. 2014, pp. 677–686.

[11] Yoon-Joo Park and Alexander Tuzhilin. “The long tail of recommender systems and how to leverage it”. In: *Proceedings of the 2008 ACM conference on Recommender systems*. ACM. 2008, pp. 11–18.

[12] Steffen Rendle et al. “BPR: Bayesian personalized ranking from implicit feedback”. In: *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press. 2009, pp. 452–461.

[13] Paul Resnick et al. “Bursting your (filter) bubble: strategies for promoting diverse exposure”. In: *Proceedings of the 2013 conference on Computer supported cooperative work companion*. ACM. 2013, pp. 95–100.

[14] Guy Shani and Asela Gunawardana. “Evaluating recommendation systems”. In: *Recommender systems handbook*. Springer, 2011, pp. 257–297.

[15] Gábor Takács and Domonkos Tikk. “Alternating least squares for personalized ranking”. In: *Proceedings of the sixth ACM conference on Recommender systems*. ACM. 2012, pp. 83–90.

[16] Jacek Wasilewski and Neil Hurley. “Incorporating Diversity in a Learning to Rank Recommender System.” In: *FLAIRS Conference*. 2016, pp. 572–578.