A unified optimization toolbox for solving popularity bias, fairness, and diversity in recommender systems

SINAN SEYMEN, Northwestern University, USA
HIMAN ABDOLLAHPOURI, Northwestern University, USA
EDWARD C. MALTHOUSE, Northwestern University, USA

Historically, the main criterion for a successful recommender system was how accurate the recommendations were according to the user's taste. This emphasis on accuracy was later challenged by researchers asking for other types of metrics such as novelty, diversity, fairness of the recommendations. Researchers have proposed different algorithms to improve these metrics of recommendation, but the problem is that each proposed algorithm improves a certain metric (diversity, novelty, etc.) and, usually, it is difficult to improve two or more aspects simultaneously. In this paper, we unify different considerations into a constrained optimization framework where different sets of metrics can be improved by simply using different sets of constraints. Therefore, our framework improves the non-accuracy metrics of the recommendations by combining different constraints designed for separate metrics. Our biggest contribution is offering models that are simple, easy to combine, and data independent. We create models considering popularity, fairness, and diversity metrics since they are the metrics widely investigated in the literature; however, our framework can include other metrics following the ideas proposed in this paper. Experimental results confirm that our general framework has comparable performance with the state-of-the-art methods designed for improving each individual metric, and offers the benefit of being able to accommodate a wide range of considerations.

CCS Concepts: • **Information systems** → **Personalization**; **Recommender systems**.

Additional Key Words and Phrases: Recommender systems, optimization, popularity bias, diversity, fairness

ACM Reference Format:
Sinan Seymen, Himan Abdollahpouri, and Edward C. Malthouse. 2021. A unified optimization toolbox for solving popularity bias, fairness, and diversity in recommender systems. 1, 1 (September 2021), 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

The initial focus of recommender systems (RS) was on estimating users' preferences accurately, where measures including Root Mean Squared Error (RMSE), precision and recall were the primary objectives. Researchers later recognized the importance of other metrics such as diversity and novelty [16], fairness between multiple stakeholders [1] and so on. Various types of criteria have been recognized as important considerations for the success of a RS and for each of them numerous algorithms have been proposed. For example, for improving the fairness of the RS from the providers' perspective, algorithms such as FairRec [27] (based on fair resource allocation) and FairMatch [24] (based on a graph-based maximum flow approach), and PFAR [22] (based on the weighted sum of relevance and fair exposure using the Maximum Marginal Relevance approach) are proposed, each offering a different approach for solving the same problem. Similarly, for mitigating the popularity bias problem Kamishima [20] uses the concept of neutrality for controlling this bias, Abdollahpouri [2] mitigates popularity bias via ensuring a balanced exposure of two groups

Presented at the MORS workshop held in conjunction with the 15th ACM Conference on Recommender Systems (RecSys), 2021, in Amsterdam, Netherlands.

Authors' addresses: Sinan Seymen, Northwestern University, Evanston, Illinois, USA; Himan Abdollahpouri, Northwestern University, Evanston, Illinois, USA, himan.abdollahpouri@northwestern.edu; Edward C. Malthouse, Northwestern University, 1845 Sheridan Road, Evanston, Illinois, USA, 60208, ecm@northwestern.edu.

of popular and less popular items in each recommendation list, Vargas [32] swaps the role of items and users and change the recommendation process as if the goal is to recommend users to each item, and many others. For improving the diversity of the recommendations within each list, Zhou [33] proposes a "heat-spreading" algorithm that can be coupled in a highly efficient hybrid with a diffusion-based recommendation method [34], Eskandanian [11] performs collaborative filtering independently on different segments of users based on the degree of diversity in their profiles, and Di Noia [9] uses a post-processing re-ranking technique to enhance the diversity of an initial recommendation list, and numerous other techniques.

One issue is that all the mentioned approaches for tackling different non-accuracy problems are implemented in isolation and cannot be easily combined to improve two or more aspects at the same time. We address this limitation by developing a constrained optimization toolkit that addresses popularity, fairness, and diversity metrics. Our framework is easy to implement and incorporate other metrics. We believe unifying all different non-accuracy related problems in RS under one umbrella can greatly benefit the research community and therefore we study how to create a list of k items for each user, assuming the preferences of each user for each item have already been estimated using some existing RS. We show how to write various considerations (e.g., diversity, popularity, fairness) as an optimization problem, and show how it can be solved as a post-processing step. We show that our toolkit achieves a comparable performance to the best-in-class algorithms for each specific task, but our toolkit is also able to improve more than one non-accuracy aspect of the recommendations by combining different constraints designed for separate aspects.

2 THE CONSTRAINED OPTIMIZATION TOOLBOX

Optimization models are applied to RS in different forms. Rodriguez [29] formulates recommendations as a constrained optimization problem and proposes the TalentMatch algorithm that matches job candidates to job posts. Another early paper using optimization with RS is Ribiero [28], which searches a Pareto frontier balancing accuracy, diversity and novelty. Jugovac [19] proposes a multi-objective, post-processing model, reviews the literature on multi-objective RS, and tests different heuristic solutions. Sürer [31] proposes integer programming models to solve RS by recommending items from stakeholders (providers) in the system in a sufficient amount for fairness. Antikacioglu and Ravi [7] use a graph optimization approach to increase diversity of the recommendation lists. Similarly, in [4], aggregate diversity is increased by graph-theoretic approach. Gogna and Majumdar [13] use regularization terms in the objective function to increase the diversity and the novelty of the solution. Other multi-objective optimization models [5, 6] are implemented to solve content recommendation problems. Works [5, 6] consider an objective function that maximizes the probability of recommendations using continuous decision variables. In another line of work, Jambor and Wang [18] propose a constrained linear optimization model for increasing the long-tail item recommendations. Most of these approaches have been tailored to solve particular RS problems, while we aim at unifying different non-accuracy aspects of the recommendations into a simple and flexible optimization approach.

This section describes our approach to solve different problems such as popularity bias, provider fairness, and diversity in RS using constrained optimization. For all problems, our technique maximizes the same objective function: the average ratings across all user and item pairs in the recommendations (i.e., the relevance of the recommendations). Different problems are addressed by adding different types of constraints. Thus, all problems have a similar structure, making it very easy to use and understand.

In the literature, some works [10, 12] have recently investigated problems including more than one non-accuracy metric. However, it is not easy to modify these models to remove some metrics and include others. Most of the time, these algorithms need significant changes to be able to incorporate different metrics other than ones that are already

proposed. We suggest a framework that alleviates this problem, where different metrics can be easily mixed and matched. In other words, our approach is inspired by how one can create a different oatmeal each morning by simply using different toppings to the base oats: the relevance objective is the base and different types of constraints are the toppings. In the following subsections, we discuss the our optimization model in more details.

2.1 Base Top-k Model

We now formalize the toolkit, beginning with notation. Let U denote the set of users and I be the set of items in the system. Suppose k items are to be recommended to each user. We assume that the ratings have been predicted with some existing algorithm, with u_{ij} representing the predicted rating for user j and item i. Decision variables x_{ij} indicate which items are recommended, with $x_{ij} = 1$ if item i is recommended to user j, and 0 otherwise. Our base model has an objective to maximize the average predicted ratings of all recommended items, subject to the constraint that each user j receives k recommendations. We can write this as an optimization problem as follows:

$$\max_{x} \quad \frac{1}{k|U|} \sum_{i \in I} \sum_{j \in U} u_{ij} x_{ij} \tag{1}$$

$$\text{subject to:} \quad \sum_{i \in I} x_{ij} = k \quad (\forall j \in U) \tag{2}$$

$$x_{ij} \in \{0, 1\} \quad (\forall i \in I, j \in U) \tag{3}$$

Note that $u_{ij}x_{ij}$ equals u_{ij} for recommended items and 0 otherwise, and therefore their sum divided by the number of recommendations made by the system ($k|U|$) gives the average rating. Constraint (2) forces the model to recommend k items to every user j. Constraint (3) forces decision variables x_{ij} to be binary (an item is either recommended or not). This problem can be solved efficiently by sorting items for each user in descending order of predicted ratings, and then selecting the top k items for every user. Both the objective function and constraints are used in the upcoming models. Therefore, we can consider this Top-k model as the base, and add constraints according to the needs of the system.

2.2 Popularity Model

Many RS have a well-known bias to recommend popular items frequently and not give enough exposure to the majority of other, less popular, items [2]. This bias can be avoided with our *popularity optimization model* (Pop-Opt), which extends the base by adding a constraint to limit the aggregate popularity of all recommended items to a given user. We implement this idea by putting an upper bound (α) on the total popularity of the recommended items. We have the same objective function in (1) subject to constraints (2), (3), and

$$\sum_{i \in I} \sum_{j \in U} x_{ij} \omega_i \leq \alpha, \tag{4}$$

where ω_i measures the popularity of item i as the ratio of the number of ratings item i received to the total number of ratings of all items in the system. Constraint (4) sums the popularity values of the recommended items and forces the sum to be at most α, a tuning parameter that can be adjusted based on the needs of the system. At one extreme, if α is very large then the selected items can be popular without exceeding threshold α and the system can focus on maximizing the average ratings. As we decrease α, the system is forced to make trade-offs and recommend some items with equal or lower ratings that are also less popular (more novelty). Choosing values for α is very intuitive. If we somewhat care about popularity, the average of ω_i times $k|U|$ can be used as a starting α value. Select a smaller value of α to offer less popular items. Constraint (4) is called a *knapsack constraint* in the literature [25], and one big advantage

of using this simple structure is that off-the-shelf optimization programs such as Gurobi are very efficient in solving this common structure. We evaluate the model with the *average recommended popularity* over all lists (*ARP*), and aggregate diversity (Agg. Div.), which is the number of unique recommended items [3]:

$$ARP = \frac{1}{|U|} \sum_{j \in U} \sum_{i \in L_j} \frac{\omega_i}{|L_j|}, \tag{5}$$

$$\text{Agg. Div.} = \frac{1}{|I|} \left| \bigcup_{j \in U} L_j \right|, \tag{6}$$

where L_j is the set of all items recommended to user j. Smaller values of *ARP* are desirable because they indicate lower popularity (more novelty). Higher values of Agg. Div is desirable because it shows the algorithm has covered a larger number of unique items in its recommendations.

2.3 Provider Fairness Model

In multi-stakeholder contexts such as a retail platform, provider fairness ensures that different *providers* (e.g., vendors) receive some minimum threshold number of recommendations. We assume that items are partitioned into *groups*. For example, items on a retail platform could be grouped by vendor or news articles could be grouped by publisher (e.g., Fox News, MSNBC, CNN, etc.). Let G_s be the set of item indices in group $s \in S$, where S is the set of all groups. Similar to Pop-Opt, provider fairness can be expressed as a constraint. Our *provider fairness optimization* (Fair-Opt) model uses the same objective function (1) as the base, subject to constraints (2), (3), and a new one that imposes a lower and upper bound (both can be tuned by the system designer) on the number of times items recommended from each group:

$$\psi_s C_s \geq \sum_{i \in G_s} \sum_{j \in U} x_{ij} \geq \gamma_s C_s \quad (\forall s \in S) \tag{7}$$

where $C_s = \frac{|G_s|}{|I|} \cdot k|U|$ is the fraction of items in group s times the total number of items recommended. Tuning parameters γ_s and ψ_s control the lower and upper bounds for the number of times items recommended from group s.

The literature on fairness usually only considers the lower bound [27, 31]. Without upper bounds, however, some items can be offered significantly more frequently than the rest, which creates an unfair distribution of recommendations across items. We choose upper bound parameter $\psi_s C_s$ as $\lfloor 1 + (2 - \gamma_s) C_s \rfloor$, which depends on γ_s. Different upper bounds can be selected according to the needs of the system.

We now discuss the selection of the number of groups. On one extreme, there could be one group with $|G_1| = |I|$ and $C_1 = k|U|$, which is the total items recommended, and the constraint would have no effect (for reasonable values of ϕ_1 and γ_1). The other extreme is where each item is a separate group, i.e., every provider has one item in the system. This case is called the *item fairness problem*, where all $\gamma_s = \gamma$, $\forall s$.

We measure fairness with Z (Inequality in Producer Exposures) [27], which is defined as follows:

$$Z = -\sum_{s \in S} \left(\frac{R_s}{|U|k} \right) \log_{|S|} \left(\frac{R_s}{|U|k} \right), \tag{8}$$

where $R_s = \sum_{j \in U} 1(s \in L_j)$ is the total number of recommendations from group s. This metric is 1 when every provider gets the same number of recommendations, and decreases as the disparity between providers increases.

2.4 Diversity Model

Another consideration in creating top-k lists is to have *diversity* in that the recommended items are not highly similar to each other [35]. Our approach is to put items in distinct groups (based on their topic, genre, etc.), and constrain the number of distinct groups represented in each top-k list to be at least w, which is another tuning parameter under the control of the system designer. Our *diversity optimization model* (Div-Opt) is the same as the base model, i.e., objective (1) subject to (2) and (3), with additional constraints:

$$\sum_{i \in G_s} x_{ij} \geq y_{sj} \quad (\forall s \in S, j \in U) \qquad (9) \qquad \sum_{s \in S} y_{sj} \geq w \quad (\forall j \in U) \qquad (10) \qquad y_{sj} \in \{0,1\} \quad (\forall s \in S, j \in U) \qquad (11)$$

We introduce a new decision variable y_{sj} that indicates whether any items from group s are recommended to user j. Constraint (11) guarantees that the new decision variable y_{sj} only takes values 0 or 1. Constraint (9) ensures that at least one recommendation is made from category s for user j, y_{sj} can get value of 1. Otherwise, it is always fixed to 0. Constraint (10) ensures that the top-k list for each user j includes at least w distinct categories.

Note that in the Div-Opt formulation every item belongs to exactly one group, which we call the binarized version. This problem can be solved in an efficient manner through sorting [8, 23]. One solution is to sort through every category separately according to the estimated ratings of the users, and recommend at least w items from separate categories. Next, we recommend items according to highest ratings until every user has exactly k items in their lists. We need Div-Opt when we mix and match different considerations to optimize combinations of metrics at the same time.

Our metric of choice for the diversity is ILS (Intra-list similarity) [35], defined as follows:

$$\text{ILS} = \frac{1}{|U|} \sum_{j \in U} \sum_{i \in L_j} \sum_{\substack{i' \in L_j \\ i' \neq i}} \frac{d(i,i')}{|L_j|(|L_j|-1)}, \qquad (12)$$

where L_j is the recommendation list of user j, and $d(i,i')$ is the distance between two items in the same list. We slightly modify the metric in [35], to put our ILS values between 0 and 1, where low values are desirable.

2.5 Combining different objectives

We combine all the different parts of the optimization models from the previous subsections. Unlike existing approaches, the beauty of our toolbox for solving different non-accuracy aspects of RS is that all constraints introduced so far can be included together. Our *combined model* (Comb-Opt) has the same objective function in (1) subject to constraints (2), (3), (4), (7), (9), (10), and (11).

Our models use different ideas from the constrained optimization literature, including upper and lower bounding in Fair-Opt, weighted sums in Pop-Opt, and auxiliary variables in Div-Opt. Using these ideas, readers can include their own metrics with small modifications. Our framework can therefore accommodate different metrics if they can be modeled as constraints. Similarly, the metrics we have investigated can be modeled differently. Our main consideration is to use the same objective function for all the models and keep the constraints as simple as possible.

We now discuss some shortcomings of the combined model and provide potential ways of approaching them. First, some values of parameters w, α, γ, ψ may be *infeasible*, which means that there does not exist any solution satisfying all the constraints at the same time. One solution is to grid-search different parameters. Another solution is to penalize these constraints on the objective function whenever they are not satisfied. Second, creating decision variables for large

models can require a lot of memory. In that case, some sort of approximation is required to make the models smaller. One way to do this is to fix some of the decision variables to 0 or 1 according to their estimated utilities before starting the optimization. Some item-user pairs with very low predicted ratings can be ignored at the start of the process, so they basically require no memory. To keep the models and the discussion compact, we leave these for future work.

3 RESULTS

In this section, we compare our optimization model for solving the non-accuracy aspects of RS (popularity bias, diversity, and provider fairness) with various other models proposed specifically to solve each individual problem. We use the MovieLens 1M data set [15] since it contains the genre of each movie, which is needed for the diversity problem. If a movie has more than one genre then we randomly assign one out of listed genres and keep the assignments consistent throughout. All the problems are solved using a laptop with Intel(R) Core (TM) i7-8750H CPU @ 2.20GHz 2.21 GHz, processor information, and with 16.0 GB of installed RAM. We use the Gurobi software [14] with optimization gap 10^{-4} throughout. Gurobi uses the branch & bound (B&B) algorithm, and it can have exponential time complexity [26] in worst-case scenario. However, Gurobi includes heuristics on top of B&B, and in practice the solution time performance is significantly better than exponential.

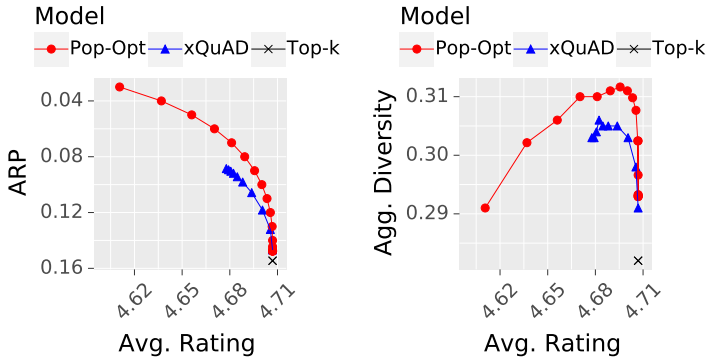
We apply the Singular Value Decomposition (SVD) [21] method implemented in the Python Surprise package [17] to estimate ratings for every user-item pair, although any RS algorithm could be used. We solve the optimization problems as a post-processing step using the predicted rating matrix. All our optimization models and benchmarks are solved using the same predicted ratings matrix. We set the size of the recommendation list given to each user to $k = 10$. Our code is available at GitHub: Ghttps://github.com/sseymen-tech/unified_toolbox.

3.1 Popularity Bias

We compare our Pop-Opt technique with the approach proposed in [2] (we label it *xQuAD*), which is one of the most efficient approaches proposed in recent years for controlling popularity bias. Figure 1 shows the average popularity of the recommendations (*ARP*) and the aggregate diversity for Pop-Opt and xQuAD for different values of α. From left to right, α changes from 0.03 to 0.16 with increments of 0.01. Larger α values give the same result because Constraint (4) is satisfied trivially. We can see that, for larger values of α, the Pop-Opt achieves a comparable average popularity to the xQuAD model with even slightly better average rating. Regarding aggregate diversity, we also see that our model outperforms xQuAD for larger values of α which, as we mentioned before, is a tunable parameter. Thus, Pop-Opt can achieve a comparable performance with the state-of-the-art technique to mitigate popularity bias.

3.2 Provider Fairness

For provider fairness, we compare our Fair-Opt with FairRec [27], which is specifically designed for this task. Figures 2 and 3 show the fairness metric Z and average rating for both techniques. From left to right the γ parameters used in both algorithms are $(0.99, 0.95, 0.9, 0.8, 0.5, 0.3, 0)$. We observe that for both relaxed and strict upper bound choices, Fair-Opt beats FairRec in both average rating and fairness metrics. When we compare Fair-Opt results with and without upper bounds in Figures 2 and 3, we notice that upper bounds improve fairness value Z without reducing the average rating of the recommendations.



ARP metric comparison

Agg. Diversity metric comparison

Fig. 1. Comparison of xQuAD with Pop-Opt

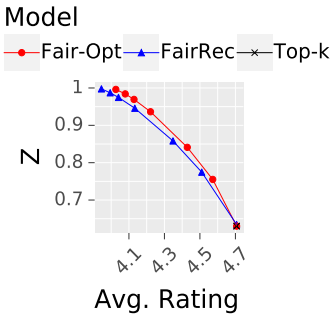


Fig. 2. Fairness Results without upper bounds

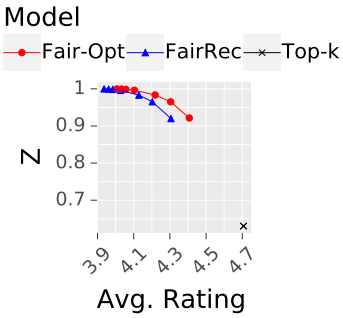


Fig. 3. Fairness Results with upper bounds

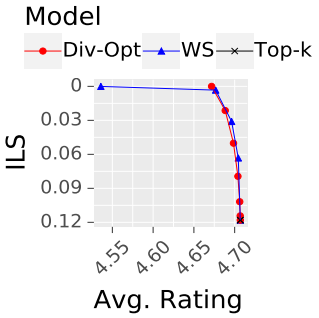


Fig. 4. Diversity Results

3.3 Diversity

We compare our diversity model (Div-Opt) with one of the most common ways of improving diversity in recommendation lists that uses a simple weighted sum of relevance and Intra List Similarity (ILS) [8] (It is denoted by *WS* in the plot). In Figure 4, from left to right Div-Opt parameter w takes values $(10, 9, 8, 7, 6, 5, 0)$. From 5 to 0 solutions are the same. for $w = 10$, we achieve ILS=0, because every user is recommended one item from every distinct category. Thus, Div-Opt can increase the number of distinct recommended genres to every user without a significant decrease in average rating. Div-Opt has achieved a comparable performance to the *WS* algorithm when $w = (9, 8, 7, 6, 5, 0)$ yet outperforms it for $w = 10$ as its ILS is close to zero but has a higher average rating.

3.4 Combined Model Results

Table 1 exhibits the results of our Comb-Opt model. Algorithms that optimize for a specific metric should perform well on that metric, but Comb-Opt can achieve great performance on ILS, *Z* and *ARP* without significantly lowering the average rating value. The results for xQuAD with lowest and highest popularity metric *ARP* (highest and lowest average rating respectively) are reported. Similarly, the results for *WS* for lowest and highest ILS are reported. For the

item fairness metric, we remove upper bounds from (no ψ parameter) both FairRec and Comb-Opt, except with the solutions with superscript $*$, which represent the solutions with upper bounds.

Model	w	γ	α	ILS	Z	ARP	Avg. Rating	Agg. Div.
Comb-Opt	1	0.9	0.09	0.142	0.969	0.067	4.128	1.0
Comb-Opt*	6	0.9	0.055	0.129	0.999	0.053	4.055	1.0
Comb-Opt	7	0.1	0.085	0.080	0.668	0.085	4.672	1.0
Comb-Opt	7	0.5	0.08	0.089	0.843	0.079	4.421	1.0
Comb-Opt	7	0.1	0.08	0.080	0.668	0.080	4.669	1.0
Comb-Opt	7	0.5	0.05	0.092	0.838	0.050	4.393	1.0
Comb-Opt	7	0.99	0.05	0.104	0.994	0.050	4.017	1.0
Comb-Opt	8	0.8	0.1	0.058	0.934	0.076	4.206	1.0
Comb-Opt	9	0.1	0.08	0.022	0.658	0.080	4.653	1.0
Top-k	-	-	-	0.118	0.630	0.155	4.707	0.282
FairRec	-	0.8	-	0.143	0.946	0.077	4.133	1.0
FairRec	-	0.99	-	0.147	0.998	0.054	3.945	1.0
FairRec*	-	0.99	-	0.147	0.999	0.052	3.936	1.0
xQuAD	-	-	-	0.103	0.629	0.089	4.678	0.303
xQuAD	-	-	-	0.118	0.634	0.147	4.707	0.291
WS	-	-	-	0	0.698	0.118	4.536	0.351
WS	-	-	-	0.118	0.634	0.147	4.707	0.291

Table 1. Results of Comb-Opt with different parameter choices

Note that in Table 1 Comb-Opt$^*(w = 6, \gamma = 0.9, \alpha = 0.055)$ beats all FairRec results in all metrics, except ARP of FairRec*. Comb-Opt$(w = 7, \gamma = 0.1, \alpha = 0.08)$ beats both xQuAD solutions in all metrics except average rating. Comb-Opt loses a bit of average rating but improves all the other metrics.

Recall that high Z, Agg. Div., and ARP values, and low ARP and ILS values are desired. In our experiments with Comb-Opt, we see that all metrics can be improved simultaneously. If one does not care about a certain metric, the constraints for that metric can be ignored in the combined model. Parameters can overwrite each other, for example in Comb-Opt with $w = 1, \gamma = 0.9$ and $\alpha = 0.09$, we have a relatively low $ARP = 0.067$ when our bound is 0.09. This happens because when γ goes to 1, every item is recommended proportionally close to each other, which gives non-popular items to be recommended as frequently as popular ones. Therefore, the $\alpha = 0.09$ bound is trivially satisfied by the fairness constraints. However, this is normal since we are dealing with different problems at the same time, and it is expected that they have some effect on each other.

Parameter choice plays a significant role, and while optimizing more than one metric, grid-search of parameters are suggested to see the behavior of the data. Comb-Opt with parameters $w = 7, \gamma = 0.5$, and $\alpha = 0.08$, for example, seem to find a good balance of all metrics.If some metrics are not important in the given problem, their corresponding constraints can be discarded. For example, if diversity is not important, Comb-Opt with parameters $w = 1, \gamma = 0.9$, and $\alpha = 0.09$ finds a solution with good fairness Z and popularity ARP metrics at the same time. Overall, parameter choice can be made according to the needs of the system.

Overall, Comb-Opt tends to improve all the metrics at the same time. If the RS goal is to alleviate popularity bias, fairness, and diversity metrics, we suggest adding all the constraints proposed in this paper. However, if one does not care about fairness, then it makes sense to remove the fairness constraints from the Comb-Opt. Then we immediately

have a solution that can alleviate popularity bias and diversity metrics. On the other hand, the constraints proposed in this paper can be removed and new ones can be added easily, according to the needs of the specific RS.

4 CONCLUSIONS AND DISCUSSION

We propose an optimization toolbox for solving different types of non-accuracy problems. This method focuses on finding the optimal solution of the system given different constraints, which aim to solve various non-accuracy problems such as popularity bias, provider fairness, and diversity. We show that, all these different metrics can be considered at the same time while generating recommendations, and they can even beat algorithms specialized for the specific problems.

One downside of our model is its memory requirement. Therefore, for larger problems, scaling of the models is an issue. There are possible solutions to these problems, such as focusing on sub-optimal solutions which are close to optimal, fixing values to some decision variables before starting to optimization and so on. We leave the problem of finding good approximations of Comb-Opt for future work.

Our toolbox can be applied to many other problems. For example, retailers concerned with stock-outs can add upper bounds for how often an item is offered. Likewise, a platform with perishable items (e.g., fresh produce or meat, hotel rooms, airline seats) could add a lower bound so they are offered more frequently. In situations with sponsored recommendations, the manufacturer of the item may have a maximum advertising budget that it is willing to spend, which could be handled by adding upper bounds. Seymen [30] applies a similar approach to the top-k list calibration problem. All these individual problems and various combinations of them and can easily be solved simply by modifying the constraints or adding new ones. Thus, post-processing optimization models offer a flexible toolkit for managing RS involving multiple objectives and/or stakeholders.

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] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2019. Managing popularity bias in recommender systems with personalized re-ranking. In *The Thirty-Second International Flairs Conference*.

[3] Gediminas Adomavicius and YoungOk Kwon. 2011. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering* 24, 5 (2011), 896–911.

[4] Gediminas Adomavicius and YoungOk Kwon. 2011. Maximizing aggregate recommendation diversity: A graph-theoretic approach. In *Proc. of the 1st International Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011)*. Citeseer, 3–10.

[5] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2011. Click shaping to optimize multiple objectives. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 132–140.

[6] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2012. Personalized click shaping through lagrangian duality for online recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 485–494.

[7] Arda Antikacioglu and R Ravi. 2017. Post processing recommender systems for diversity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 707–716.

[8] Keith Bradley and Barry Smyth. 2001. Improving recommendation diversity. In *Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science, Maynooth, Ireland*, Vol. 85. Citeseer, 141–152.

[9] Tommaso Di Noia, Jessica Rosati, Paolo Tomeo, and Eugenio Di Sciascio. 2017. Adaptive multi-attribute diversity for recommender systems. *Information Sciences* 382 (2017), 234–253.

[10] Jorge Díez, David Martínez-Rego, Amparo Alonso-Betanzos, Oscar Luaces, and Antonio Bahamonde. 2019. Optimizing novelty and diversity in recommendations. *Progress in Artificial Intelligence* 8, 1 (2019), 101–109.

[11] Farzad Eskandanian, Bamshad Mobasher, and Robin Burke. 2017. A clustering approach for personalizing diversity in collaborative recommender systems. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. 280–284.

[12] Ruoyuan Gao and Chirag Shah. 2020. Toward creating a fairer ranking in search engine results. *Information Processing & Management* 57, 1 (2020), 102138.

[13] Anupriya Gogna and Angshul Majumdar. 2017. DiABlO: Optimization based design for improving diversity in recommender system. *Information Sciences* 378 (2017), 59–74.

[14] Gurobi Optimization, LLC. 2021. Gurobi Optimizer Reference Manual. https://www.gurobi.com

[15] 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.

[16] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 5–53.

[17] Nicolas Hug. 2017. Surprise, a Python library for recommender systems. http://surpriselib.com.

[18] Tamas Jambor and Jun Wang. 2010. Optimizing multiple objectives in collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*. 55–62.

[19] Michael Jugovac, Dietmar Jannach, and Lukas Lerche. 2017. Efficient optimization of multiple recommendation quality factors according to individual user tendencies. *Expert Systems with Applications* 81 (2017), 321–331.

[20] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2014. Correcting Popularity Bias by Enhancing Recommendation Neutrality.. In *RecSys Posters*.

[21] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[22] Weiwen Liu and Robin Burke. 2018. Personalizing fairness-aware re-ranking. In *International Workshop on Fairness Accountability and Transpatency in Recommender Systems (FACTREC)*. arXiv preprint arXiv:1809.02921.

[23] Edward C Malthouse, Khadija Ali Vakeel, Yasaman Kamyab Hessary, Robin Burke, and Morana Fuduric. 2019. A Multistakeholder Recommender Systems Algorithm for Allocating Sponsored Recommendations.. In *RMSE@ RecSys*.

[24] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. Fairmatch: A graph-based approach for improving aggregate diversity in recommender systems. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*. 154–162.

[25] Silvano Martello and Paolo Toth. 1987. Algorithms for knapsack problems. In *North-Holland Mathematics Studies*. Vol. 132. Elsevier, 213–257.

[26] David R Morrison, Sheldon H Jacobson, Jason J Sauppe, and Edward C Sewell. 2016. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization* 19 (2016), 79–102.

[27] 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 2020*. 1194–1204.

[28] Marco Tulio Ribeiro, Anisio Lacerda, Adriano Veloso, and Nivio Ziviani. 2012. Pareto-efficient hybridization for multi-objective recommender systems. In *Proceedings of the sixth ACM conference on Recommender systems*. 19–26.

[29] Mario Rodriguez, Christian Posse, and Ethan Zhang. 2012. Multiple objective optimization in recommender systems. In *Proceedings of the sixth ACM conference on Recommender systems*. 11–18.

[30] Sinan Seymen, Himan Abdollahpouri, and Edward C Malthouse. 2021. A Constrained Optimization Approach for Calibrated Recommendations. In *Proc. of the 15th ACM Conf. on Rec. Sys.* ACM.

[31] Özge Sürer, Robin Burke, and Edward C Malthouse. 2018. Multistakeholder recommendation with provider constraints. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 54–62.

[32] Saúl Vargas and Pablo Castells. 2014. Improving sales diversity by recommending users to items. In *Proceedings of the 8th ACM Conference on Recommender systems*. 145–152.

[33] Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences* 107, 10 (2010), 4511–4515.

[34] Tao Zhou, Jie Ren, Matúš Medo, and Yi-Cheng Zhang. 2007. Bipartite network projection and personal recommendation. *Physical review E* 76, 4 (2007), 046115.

[35] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*. 22–32.