Personalized Regularization Learning for Fairer Matrix Factorization

Sirui Yao[1] and Bert Huang[2]

[1] Virginia Tech, Blacksburg, VA 24060, USA ysirui@vt.edu
[2] Tufts University, Medford, MA 02155, USA bert@cs.tufts.edu

Abstract. Matrix factorization is a canonical method for modeling user preferences for items. Regularization of matrix factorization models often uses a single hyperparameter tuned globally based on metrics evaluated on all data. However, due to the differences in the structure of per-user data, a globally optimal value may not be locally optimal for each individual user, leading to an unfair disparity in performance. Therefore, we propose to tune individual regularization parameters for each user. Our approach, *personalized regularization learning* (PRL), solves a secondary learning problem of finding the per-user regularization parameters by back-propagating through alternating least squares. Experiments on a benchmark dataset with different user group splits show that PRL outperforms existing methods in improving model performance for disadvantaged groups. We also analyze the learned parameters, finding insights into the effect of regularization on subpopulations with varying properties.

Keywords: Matrix Factorization · Fairness · Error Disparity · Personalized Regularization

1 Introduction

Matrix factorization is an important and widely adapted collaborative filtering technique for training recommender systems to predict ratings. However, MF has been found to be easily influenced by data biases and becomes unfair [15,10]. For example, demographic groups for whom training data is less frequently available can suffer less accurate predictions of their preferences [15]. This phenomenon is a form of *error-based unfairness* where users may receive lower quality service because of a demographic attribute that ideally should not affect their experience. To make it worse, the group of users who receive less accurate recommendations are more likely to abandon the service, leading to an even more biased environment and more unfair models in the future [9].

Collecting more and better quality data for the disadvantaged groups will help a model better learn these users' preferences. However, this approach is usually expensive or even infeasible. For example, a recommender service provider cannot request users to change how they interact with the service. Therefore, the more important questions is, can we handle these biases more appropriately to make better use of the available data, and build a model with improved accuracy for the ill-served users?

In this work, we first consider different types of data biases, which all refer to a certain form of divergence in the structure of per-user or per-group data. We verify on synthetic datasets that these biases can lead to one subgroup experiencing higher error than the others. We then consider the connection between prediction error and the role of regularization. If we acknowledge the difference in per-user data, then instead of tuning a global hyperparameter, a matrix factorization could benefit from personalized regularization, which better accommodates each individual user. This strategy not only directly addresses the cause of error disparity, but also provides more interpretability compared to directly manipulate the latent features since regularization is a comparatively well-understood concept.

Since personalized regularization drastically increases the number of hyperparameters, commonly used hyperparameter searching procedures—such as grid search and random search—become prohibitively expensive. It is also challenging to derive the parameters from heuristic because, in joint embedding models like matrix factorization, the effect of personalized regularization parameters are not independent of each other. Therefore, we propose a learning problem, *personalized regularization learning* (PRL), to learn the optimal set of hyperparameters that minimizes a secondary objective, in our case, the error of the disadvantage groups. We consider the secondary objective as a function of the personalized regularization parameters. To enable direct back-propagation and facilitate efficient learning, we leverage the closed-form solutions of *alternating least squares* (ALS) to solve MF.

The main contributions of this paper are as follows:

1. We identify the insufficiency of global regularization for matrix factorization in dealing with complexity or sparsity imbalance across users, and conduct validation on synthetic data with explicitly injected biases;
2. We propose *personalized regularization learning* (PRL), an interpretable algorithm for learning personalized regularization by back-propagating through the closed-form computation of ALS;
3. We demonstrate the effectiveness of the proposed approach with experiments on a benchmark dataset with different user group splits, comparing against three baseline models.

2 Related Work

In this section, we review literature related to the problem we aim to solve and our proposed approach.

Error disparity Error disparity is a form of error-based unfairness. Yao and Huang [15] discuss four variants of unfairness metrics computed based on the divergence between prediction and labels. [15] propose to reduce error disparity by adding the optimized metrics to the standard matrix factorization objective such that the model is incentivized to reduce disparity. Both [9] and [11] solve the error disparity problem by minimizing the maximum subgroup error, which is an upper bound on error disparity. In recommender systems, this phenomenon is also related to the cold-start problem [6,5], where the cause of high error is limited to insufficient data. Other notions of fairness such as statistical parity [16] is also studied in recommender systems but is not the focus of this paper.

Differentiated regularization Besides separating the regularization parameter of users and items, Beutel et al. [2] proposes to assign the advantaged and disadvantaged subpopulation to different sets, and regularize them differently. Chen et al. [5] take a step further by computing a set of per-user regularization parameter by linear or logarithm functions of user sparsity. Our work is closely related to [2] and [5] because we also seek to differentiate regularization but through an optimization-based approach.

Hyperparameter search In practice, grid search is often used when the space of hyperparameters is small. Random search [1] can be more efficient because it randomly samples hyperparameter values instead of trying all combinations of the candidate hyperparameters. Bayesian hyperparameter optimization [13] speeds up random search by taking into consideration the past evaluations to decide what areas of hyperparameter space to search next. For the task of optimizing a large number of hyperparameters, [12] devise a reversible learning method to compute hyperparameter gradients by reversing the dynamics of gradient descent. Our approach also uses gradient-based hyperparameter tuning, but we leverage the differentiability of closed-form updates in alternating least squares for matrix factorization to directly back-propagate to the hyperparameters.

3 Problem Definition

Given a dataset $\mathcal{D}$ that contains ratings by M users on N items. which can be represented as an $M \times N$ sparse matrix R where each observed entry r_{ui} represents the rating user u gives to item i. Suppose each user is associated with a set of properties S, based on one or more such properties $s \in S$, we can split users into a set of subgroups G.

A rating prediction model predicts the missing values in the sparse rating matrix. We randomly split all observed ratings into R^{Train} and R^{Test}. We train a model on R^{Train}, and use root mean squared error (RMSE) to measure prediction error on R^{Test} as

$$RMSE = \sqrt{\frac{1}{|R^{\text{Test}}|} \sum_{r_{ui} \in R^{\text{Test}}} (r_{ui} - \hat{r}_{ui})^2)} \quad (1)$$

where $\hat{r}_{ui}$ is the predicted value of r_{ui}. With a matrix factorization model, users and items are projected as matrices $P \in \mathbb{R}^{M \times d}$ and $Q \in \mathbb{R}^{N \times d}$. The uth row of P, denoted as p_u, is the latent feature of user u; the ith row of Q, denoted as q_i, is the latent feature of item i. The ratings are predicted as $\hat{r}_{ui} = p_u q_i^\intercal$.

Problem Formulation Given a user subgroup of concern $\hat{g} \in G$, which has higher prediction error and is considered to be the disadvantage population. The goal is to find a model that reduces error for this subgroup. The error of a subgroup $\hat{g} \in G$ is denoted and measured as

$$RMSE_{\hat{g}} = \sqrt{\frac{1}{|R_{\hat{g}}^{\text{Test}}|} \sum_{r_{ui} \in R_{\hat{g}}^{\text{Test}}} (r_{ui} - \hat{r}_{ui})^2)} \tag{2}$$

where $R_{\hat{g}}^{\text{Train}}$ and $R_{\hat{g}}^{\text{Test}}$ denote the training and test data of $\hat{g}$ respectively.

4 Data Biases and Regularization

In this section, we discuss four types of data biases that contribute to higher prediction error in disadvantaged subgroups, and empirically show the consequences of these biases with synthetic datasets. We also discuss how these data biases are related to regularization and imply the need for personalized regularization.

4.1 Data Biases

We first consider a group-level bias called *population bias*, which refers to the discrepancy among the size of subgroups. The subgroups with smaller populations are more likely to be compromised in modeling, especially when the data of these minority groups have a very different structure from the other groups. We also consider three individual-level biases. The first one is *sparsity bias*, which refers to the difference in per-user data sparsity. A model with a particular complexity requires a corresponding amount of data to overcome the curse of dimensionality, which creates a disadvantage for users who are new or less active. The second one is *rank bias*, it refers to the situation that some users' preferences are more complicated than others. Therefore, a higher-dimensional model is required to capture their preferences. The third one is *noise bias*, which suggests different levels of data quality and the situation where some users' data is more noisy than the others. We believe these four types of data biases lead to increased prediction error for the subgroups that are being biased against.

4.2 Validation

We validate our heuristics on the effect of data bias on synthetic datasets where we explicitly inject two types of data bias among users. We create the synthetic data by first generating a twenty-dimensional user and item feature matrices for 100 users and 600 items. Then we compute the rating matrix as their dot product. To make the datasets more realistic, we also add Gaussian noise to these ratings and clamp them within the range of 1.0 to 5.0.

We assign users to two subgroups A and B. To inject data biases, without loss of generosity, we choose group B to be the disadvantaged group and the users from group B to be the disadvantaged users. For population bias, we lower the population of group B to be the minority group; for rank bias, we force some columns of the latent features of users from group A to be zero, so that group B has higher dimension than group A; for sparsity bias, we mask more ratings from group B than group A; for noise bias, we add a higher level of Gaussian noise to the rating of group B. Specifically, to create the biased settings, we set $|B| = 30, |A| = 70$; $d_A = 5, d_B = 20$; $5\times$ amount of ratings are observed from group A than group B; $2\times$ amount of noise is added to the ratings of group B[3].

We train matrix factorization models on these datasets and measure $RMSE_B$, the error of group B. The results are shown in Figure 1. The blue bar represents a bias-free dataset; the orange bars represent datasets with each individual type of bias; the green, red, and purple bars represent datasets with 2, 3, and 4 types of biases respectively. R, N, P, S are short for rank bias, noise bias, population bias, and sparsity bias respectively. First, by comparing the fair setting (the blue bar) and the settings with each individual biases (the orange bars), we observe that, except population bias, each of the discussed biases alone directly leads to higher $RMSE_B$. Population bias is the exception because when the other three biases are not present, the two subpopulations have exactly the same data structure. Second, in general, $RMSE_B$ increases as more types of biases are injected, suggesting a compound impact of data biases. Third, the effect of these data biases are not independent but can enhance each other. For example, we observe a noticeable increase in $RMSE_B$ from setting "R, S, N" to "R, S, N, P" when population bias is added, which by itself does not have the same effect.

4.3 Relation to Regularization

The data biases discussed above are directly related to data properties that, if not carefully handled when building a model, will lead to overfitting or underfitting. Overfitting or underfitting are two forms of mistakes that a model could make to mishandle training data and increase error. Overfitting happens when a model attends to too much detail and noise, and is more likely to occur when data is insufficient due to increase variance; underfitting, on the other hand, happens when a model oversimplifies and fail to capture the underlying structure of the data.

An important component for balancing underfitting and overfitting in machine learning models is regularization. The strength of regularization needs to be tuned to best fit the learning task and the data. Since we discussed that data properties such as quality, sparsity, and complexity may not be universal across all users, tuning a global regularization parameter λ^*, as is done in standard matrix factorization models, becomes insufficient to accommodate the important differences in per-user data, leading to poor model performance on certain user subgroups.

[3] Note that to avoid the effect of irrelevant factors, we normalize the ratings so that the ratings of group A and B follow the same distribution. We also keep the overall level of sparsity and noise unchanged by rescaling the configuration within each subgroup.

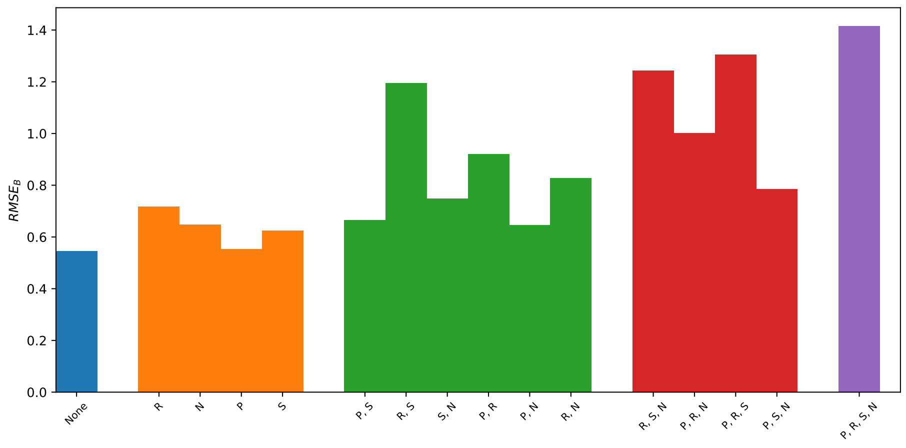

Fig. 1: The measured $RMSE_B$ of models trained on synthetic datasets with different data biases injected. Here we use R, N, P, S to indicate rank bias, noise bias, population bias, and sparsity bias respectively. The models are grouped based on the number of injected data biases and are presented in different colors.

5 Personalized Regularization Learning

We have discussed that a globally tuned regularization parameter λ^* neglects the differences in per-user data. Therefore, we believe a model could benefit from a set of personalized regularization parameters. With this expanded set of hyperparameters, the objective function of training a matrix factorization model is modified by replacing the globally tuned regularization hyperparameter λ^* with user-personalized regularization parameters $\Lambda = \{\lambda_u\}_{u=1}^{M}$. The user and item latent features are learned as

$$P^*, Q^* = \min_{P,Q} \sum_{r_{ui} \in R^{\text{Train}}} (r_{ui} - P_u Q_i^\intercal)^2 + \frac{1}{2}\left(\sum_{u=1}^{M} \boldsymbol{\lambda_u}(P_u P_u^\intercal) + \sum_{i=1}^{N} \lambda^*(Q_i Q_i^\intercal)\right) \tag{3}$$

Since personalized regularization grows the space of hyperparameters from $\mathbb{R}$ to $\mathbb{R}^M$, traditional tuning procedures such as grid search become insufficient in such a high-dimensional space. Also, in joint embedding models like matrix factorization, the personalized regularization parameters are not independent of each other, therefore, it is challenging to derive the parameters from heuristics.

5.1 Personalized Regularization Learning

To efficiently search for the optimal personalized hyperparameters, we propose *personalized regularization learning* (PRL), which poses the hyperparameter search problem as a secondary learning task. We denote the primary learning problem in Equation (3) as L, which returns the learned P, Q for a given hyperparameter set Λ

$$P, Q = L(\Lambda) \tag{4}$$

We then make predictions using the learned latent features P and Q through a predictor function H,

$$\hat{R} = H(P, Q) \tag{5}$$

and evaluate a secondary objective, which in our running example, is the subgroup error $RMSE_{\hat{g}}$ through function E,

$$RMSE_{\hat{g}} = E(\hat{R}) \tag{6}$$

Combining equation Equations (4) to (6), we get $RMSE_{\hat{g}} = E(H(L(\Lambda))) = F(\Lambda)$ where $F = E(H(L))$. The secondary learning problem is formulated as

$$\Lambda^* = \min_{\Lambda \in \mathbb{R}^M} F(\Lambda) \tag{7}$$

F is a differentiable function if E, H, and L are all differentiable. Then we can directly backpropagate through F to compute gradients of the secondary objective with respect to Λ.

5.2 Leveraging ALS

Solving the factorization problem L involves minimizing a non-convex regularized squared reconstruction error. One approach for optimizing this objective is through gradient descent [3], which has been made especially convenient with the advance of automatic differentiation tools [4]. However, the gradient of hyperparameters are usually unavailable [12]. Therefore, we instead use alternating least squares (ALS) [14] to solve L, which alternates between optimizing P and Q by iteratively applying a closed-form solution

$$\begin{aligned} P_u &\leftarrow \Bigg(\sum_{i:(u,i)\in\mathcal{D}} \tilde{q_i}\tilde{q_i}^T + \lambda^* I_d \Bigg)^{-1} \sum_{i:(u,i)\in\mathcal{D}} r_{ui}\tilde{q_i} \\ Q_i &\leftarrow \Bigg(\sum_{u:(u,i)\in\mathcal{D}} \tilde{p_u}\tilde{p_u}^T + \lambda^* I_d \Bigg)^{-1} \sum_{u:(u,i)\in\mathcal{D}} (r_{ui} - b_u)\tilde{p_u} \end{aligned} \tag{8}$$

The closed-form updates are differentiable, so we can conveniently back-propagate through F to compute gradients of the secondary objective with respect to Λ and learn Λ with a standard gradient-based optimizer. Since the time complexity of computing partial derivative is the same as forward passing [7], the time it takes to back-propagate to Λ is the same as forward ALS, thus the time complexity of PRL is $\mathcal{O}(T)$, where T is the number of epochs ALS takes to converge. This is on par with the state-of-the-art hyperparameter optimization techniques [12].

5.3 Data Split

During learning, we must use different datasets for training the MF model and measuring subpopulation error. This is because our goal is to decrease generalization error, which needs to be evaluated on data unseen by the training algorithm. If we measure subpopulation error on the same data that the MF model is trained on, we may simply incentivize the learning optimization to overfit the data as much as possible.

Therefore, after we split data $\mathcal{R}$ into $\mathcal{R}^{\text{Train}}$ and $\mathcal{R}^{\text{Test}}$, we further split $\mathcal{R}^{\text{Train}}$ into $\mathcal{R}^{\text{Train}-\text{Primary}}$ and $\mathcal{R}^{\text{Train}-\text{Secondary}}$. In each PRL iteration, we train a matrix factorization model on $\mathcal{R}^{\text{Train}-\text{Primary}}$ and compute $RMSE_{\hat{g}}$ on $\mathcal{R}^{\text{Train}-\text{Secondary}}$, which is used to update Λ. After we obtained Λ^*, we apply it to train a final matrix factorization model on the full training set $\mathcal{R}^{\text{Train}}$. We then evaluate $RMSE_{\hat{g}}$ on $\mathcal{R}^{\text{Test}}$. The full algorithm is listed as Algorithm 1. In practice, we recommend creating multiple primary-secondary splits of $\mathcal{R}^{\text{Train}}$ so that the learned Λ^* is not overfitted to one particular split.

Algorithm 1: Personalized Regularization Learning

```
Given dataset $\mathcal{R}$, global optimal lambda $\lambda^*$, MF model $L$, error metric $E$,
  disadvantaged subpopulation $\hat{g}$. Split $\mathcal{R}$ into $\mathcal{R}^{\text{Train}}$ and $\mathcal{R}^{\text{Test}}$, further split
  $\mathcal{R}^{\text{Train}}$ into $\mathcal{R}^{\text{Train}-\text{Primary}}$ and $\mathcal{R}^{\text{Train}-\text{Secondary}}$.
Initialize $\Lambda \leftarrow \{\lambda_i = \lambda^*\}_{i=1}^{N}$, randomly initialize $P$ and $Q$
while not converged do
    $P^*, Q^* \xleftarrow{\mathcal{R}^{\text{Train}-\text{Primary}}} L(\Lambda)$
    $\hat{R} \xleftarrow{\mathcal{R}^{\text{Train}-\text{Secondary}}} H(P^*, Q^*)$
    $RMSE_{\hat{g}} \xleftarrow{\mathcal{R}^{\text{Train}-\text{Secondary}}} E(\hat{R}_{\hat{g}})$
    compute gradient $\nabla_{\Lambda} RMSE_{\hat{g}}$ through backpropagation
    update $\Lambda$ with $\nabla_{\Lambda} RMSE_{\hat{g}}$
end
Re-initialize $P$ and $Q$
$P^*, Q^* \xleftarrow{\mathcal{R}^{\text{Train}}} L(\Lambda^*)$
```

5.4 Interpretability

A key advantage of PRL is that it provides interpretable feedback in the magnitude of the learned per-user regularization parameters. Compared to regularization-based methods that directly manipulate user and item latent representations, PRL's learned parameters indicate the level of regularization, which is comparatively well-understood and can help us understand how the model is improved. We can interpret them by comparing their values against the globally tuned value. If a user's parameter increases, it suggests that this user would have been overfitted. Conversely, if a user receives lower regularization from PRL, they were prone to underfit and needed a more complex model.

6 Experiments

6.1 Datasets

The choice of public real datasets that provide user demographic information is very limited. We use the benchmark MovieLens 100k dataset [8], which contains 100,000 ratings from 1,000 users on 1,700 movies, and conveniently provides multiple user demographic features. Specifically, we consider demographic information such as gender, age, zip code; we also consider user degree–the number of ratings each user has, and user error–the error of each user with a vanilla matrix factorization model. For gender, we split users by category and create two subgroups (female and male users); for zipcode, we split by the first digit of zip code and create 10 subgroups, representing users from different regions in the US; for age, degree, and error, we split by percentile and each split creates 10 equal size subgroups.

We randomly sample 10% of data as holdout set for testing and use the rest as training set. Then we do 10-fold cross-validation on the training set to select the best global regularization weight λ and the rank d. The optimal combination we found is $d^* = 30$ and $\lambda^* = 20.0$. We train a standard matrix factorization model and measure the subgroup errors under all user splits. For each split, we pick the subgroup with the highest error as the disadvantaged group, denoted as $\hat{g}$ and seek to reduce $RMSE_{\hat{g}}$. The disadvantaged subgroups are listed in the second row of Table 1.

6.2 Baselines

Focused learning (FL) FL [2] assigns users to only two subgroups, a focused set, and an unfocused set. The two sets of users are regularized differently to optimize the model performance on the focused set of users. The optimal hyperparameter pair is searched via grid search.

Differentiated regularization (DR) DR [5] is motivated to alleviate the cold-start problem and regularize every user differently. The regularization parameters are computed from three functions (one linear and two logarithmic) of user degree. We denote the three formulas as DR-linear, DR-Log-1, and DR-Log-2.

Unfairness-regularized matrix factorization (URMF) URMF [15] is designed to optimize a secondary fairness in matrix factorization models. The strategy is to add the optimized secondary objective as a penalty term to the standard matrix factorization objective, weighted by a weight parameter. URMF directly manipulates the fitted latent embeddings instead of through regularization.

6.3 Specifications and Results

For DR, we directly apply the three formula (Equation 5 in [5]) to compute personalized regularization parameters. For FL, we follow the same procedure as proposed by the authors and try a range of regularization values on the focused and unfocused set, $\{0.001, 0.01, 0.1, 1, 1, 10, 20, 30, 50, 100\}$, which gives 100 combinations. For URMF, we try 10 different unfairness pernalty weights $\{1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 5, 10, 20\}$. For FL and UR, we identify the optimal setting or weight via cross-validation, then apply the same setting or weight to train a final model on the full training set. For all trained models, we measure $RMSE_{\hat{g}}$ on the holdout test set.

We show the results of all compared models on different user splits in Table 1. We first compare the performance of PRL against the standard matrix factorization model. We observed that PRL successfully reduces $RMSE_{\hat{g}}$ on all user splits, and achieves more than 10% improvement on subgroups split by Zip Code and Error. We also observe that PRL outperforms all baseline models by a convincing margin. Focused learning is the second-best method, this further suggests that fitting different regularization is effective in optimizing subpopulation error. We believe PRL wins over Focused learning due to the expanded set of hyperparameters and smart search through optimization. We observe a big fluctuation in the performance of URMF, it is possibly because URMF still can easily overfit to the training data since it measures both primary and secondary objectives on the same data. Lastly, DR performs the worst. It rarely reduces $RMSE_{\hat{g}}$ and even when it does, the improvements are trivial. This pattern aligns with the results and conclusion in the original paper that DR sometimes makes things worse and especially so on the MovieLens dataset. The poor performance of DR suggests it is challenging to find a one-fits-all heuristic for setting the personalized regularizations.

Table 1: Comparison of all model performance in reducing $RMSE_{\hat{g}}$. The rows are the $RMSE_{\hat{g}}$ of a standard matrix factorization model and all compared models, the lower the better. The columns are different user group splits. Bold values are the most significant improvement in each column.

User Split	Gender	Age	Zip Code	Degree	Error
$\hat{g}$	F	52-59	0	0%-10%	90%-100%
MF	1.029	1.045	1.062	1.118	1.612
PRL	**0.967(-6.0%)**	**0.983(-5.9%)**	**0.952(-10.4%)**	**1.043(-6.7%)**	**1.367(-15.2%)**
FL	0.998(-3.0%)	1.001(-4.2%)	0.989(-6.9%)	1.094(-2.1%)	1.479(-8.2%)
URMF	1.013(-1.6%)	1.089(+4.2%)	0.956(-10.0%)	1.102(-1.4%)	1.579(-2.0%)
DR-Linear	1.041(+1.2%)	1.127(+7.8%)	1.047(-1.4%)	1.114(-0.4%)	1.601(-0.7%)
DR-Log-1	1.067(+3.7%)	1.113(+6.5%)	1.082(+1.9%)	1.109(-0.8%)	1.641(+1.7%)
DR-Log-2	1.059(+2.9%)	1.114(+6.6%)	1.098(+3.9%)	1.112(-0.5%)	1.632(+1.2%)

We next examine the personalized regularization parameter fitted through PRL to understand how it reduces $RMSE_{\hat{g}}$. We compute the mean and standard deviation of the regularization parameters in each subgroup throughout PRL optimization. We show the plot in gender subgroups as an example in Figure 2. In this case, female users is the disadvantaged subgroup. We observe that female users, on average, have been assigned lower regularization, and male users' regularization has been increased. This provides an interesting insight that the complexity of the originally tuned global model was lower than what is need for the disadvantaged group. PRL allows these users to enjoy a more complex model that better captures their preferences. We also noticed an increased variance in the regularization parameter, and surprisingly even more so in the advantaged group. As we discussed in Section 5, the personalized regularization parameters are not independent of each other in joint embedding models, therefore, the regularization of all users are shifted even though the objective is to only optimize the prediction error of the disadvantaged group. Further, we found the same direction of change in the pair of regularization parameters identified via focused learning ($\lambda_F = 10$ and $\lambda_M = 30$). This suggests an alignment between the two methods in reducing high subpopulation error through adjusted regularization.

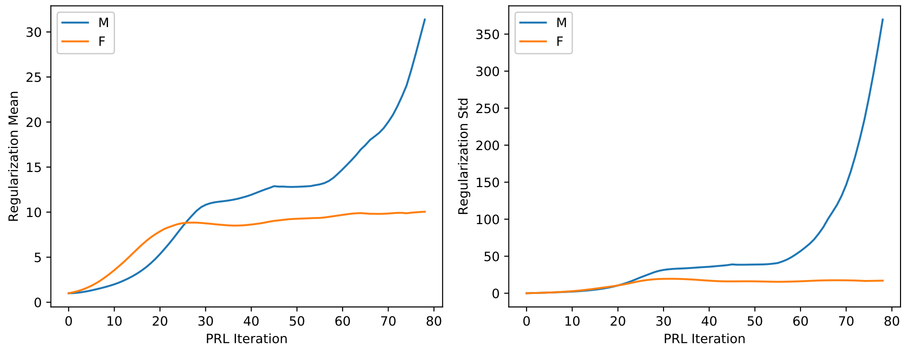

Fig. 2: The curve of mean and standard deviation of personalized regularization values during PRL. F means female users and M means male users.

7 Discussion

In this work, we address the problem of error disparity in matrix factorization models. We discuss four types of biases that contribute to higher subpopulation error and validate their effect on synthetic datasets. We presented *personalized regularization learning* (PRL), a method that learns to regularize users differently to improve prediction performance for disadvantaged subgroups of users. PRL solves a secondary learning problem to minimize validation unfairness by back-propagating through alternating least squares. In experiments, PRL outperforms existing methods for reducing error disparity in recommendations. Moreover, the learned per-user regularization parameters are interpretable and provide insight into how fairness is improved. For future work, we are interested in investigating the effectiveness of PRL in other variants of matrix factorization, such as SVD++, factorization machine. We are also interested in further exploring the learned regularization parameters to uncover richer group structures.

Acknowledgement

This work was partially supported by a Deloitte Data Analytics Fellowship and an Amazon Research Award.

References

1. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of machine learning research **13**(Feb), 281–305 (2012)
2. Beutel, A., Chi, E.H., Cheng, Z., Pham, H., Anderson, J.: Beyond globally optimal: Focused learning for improved recommendations. In: Proceedings of the 26th International Conference on World Wide Web. pp. 203–212 (2017)
3. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010, pp. 177–186. Springer (2010)
4. Bücker, H.M., Corliss, G., Hovland, P., Naumann, U., Norris, B.: Automatic differentiation: applications, theory, and implementations, vol. 50. Springer Science & Business Media (2006)
5. Chen, H.H., Chen, P.: Differentiating regularization weights–a simple mechanism to alleviate cold start in recommender systems. ACM Transactions on Knowledge Discovery from Data (TKDD) **13**(1), 1–22 (2019)
6. Ferraro, A.: Music cold-start and long-tail recommendation: bias in deep representations. In: Proceedings of the 13th ACM Conference on Recommender Systems. pp. 586–590 (2019)
7. Griewank, A., Walther, A.: Evaluating derivatives: principles and techniques of algorithmic differentiation. SIAM (2008)
8. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems **5**(4), 1–19 (2015)
9. Hashimoto, T.B., Srivastava, M., Namkoong, H., Liang, P.: Fairness without demographics in repeated loss minimization. arXiv preprint arXiv:1806.08010 (2018)
10. Kamishima, T., Akaho, S., Asoh, H., Sakuma, J.: Recommendation independence. In: Conference on Fairness, Accountability and Transparency. pp. 187–201 (2018)
11. Kim, M.P., Ghorbani, A., Zou, J.: Multiaccuracy: Black-box post-processing for fairness in classification. In: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society. pp. 247–254 (2019)
12. Maclaurin, D., Duvenaud, D., Adams, R.: Gradient-based hyperparameter optimization through reversible learning. In: International Conference on Machine Learning. pp. 2113–2122 (2015)
13. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. In: Advances in Neural Information Processing Systems. pp. 2951–2959 (2012)
14. Takács, G., Tikk, D.: Alternating least squares for personalized ranking. In: Proc. of the ACM Conference on Recommender Systems. pp. 83–90 (2012)
15. Yao, S., Huang, B.: Beyond parity: Fairness objectives for collaborative filtering. In: Advances in Neural Information Processing Systems. pp. 2921–2930 (2017)
16. Zhu, Z., Hu, X., Caverlee, J.: Fairness-aware tensor-based recommendation. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 1153–1162 (2018)