Beyond Globally Optimal: Focused Learning for Improved Recommendations

Alex Beutel[1]*, Ed H. Chi[1], Zhiyuan Cheng[2]†, Hubert Pham[1], John Anderson[1]
[1]Google, Inc. [2]Pinterest
{alexbeutel, edchi, hubertpham, janders}@google.com

ABSTRACT

When building a recommender system, how can we ensure that *all* items are modeled well? Classically, recommender systems are built, optimized, and tuned to improve a global prediction objective, such as root mean squared error. However, as we demonstrate, these recommender systems often leave many items badly-modeled and thus under-served. Further, we give both empirical and theoretical evidence that no single matrix factorization, under current state-of-the-art methods, gives optimal results for each item.

As a result, we ask: how can we learn additional models to improve the recommendation quality for a specified subset of items? We offer a new technique called *focused learning*, based on hyperparameter optimization and a customized matrix factorization objective. Applying focused learning on top of weighted matrix factorization, factorization machines, and LLORMA, we demonstrate prediction accuracy improvements on multiple datasets. For instance, on MovieLens we achieve as much as a 17% improvement in prediction accuracy for niche movies, cold-start items, and even the most badly-modeled items in the original model.

Keywords

recommendation; regularization

1. INTRODUCTION

How can we predict what movies a user will like? Or to which users an app would be appealing? How can we ensure that all movies or apps have a good opportunity to be surfaced? Recommender systems have become an integral part of our everyday lives, from Netflix recommending movies to Yelp suggesting restaurants to Google Play offering music and apps. While these uses of recommender systems have clearly been successful, little research has focused on ensuring that *all* items in these recommender systems are modeled well. For these systems to be continually trusted, it is crucial that we understand where they fail and how to improve recommendation quality for all items.

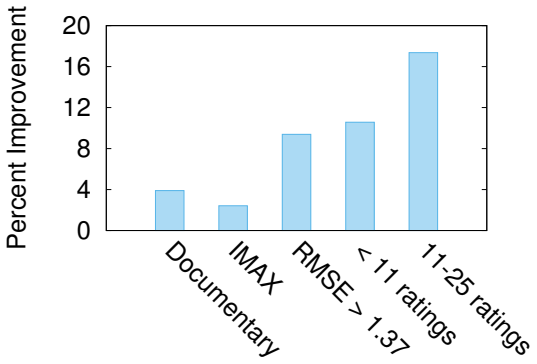


Figure 1: Focused learning improves MovieLens predictions for under-served categories of movies, including (1) niche genres, (2) items for which we have few observations, and (3) even the most badly-modeled items from our original model.

Concretely, much of the recent development in recommender systems research has been based on matrix factorization (MF) [17, 19, 32], where we use a database of user ratings of items to learn a latent bilinear model for predicting unobserved ratings. Following the Netflix Prize, these models generally aim to improve Root Mean Squared Error (RMSE) over a random holdout of ratings. While there are many advantages to such an approach, focusing on the average accuracy metrics, such as RMSE, leaves many items ill-served. In fact, as we will demonstrate, common properties of real-world data sets encourage a skewed recommendation policy where some items are modeled far worse than others.

Given this issue with classic factorization models, how can we learn a model, just using ratings data, that is focused on improving recommendation accuracy for badly-modeled items, or any subset of items? How can we use all observed ratings to improve the predictions of a subset? We call this the *focused learning* problem.

This problem is related to multiple previous lines of recommendation research. Research on the cold-start problem attempts to improve recommendation accuracy for items with few observed ratings, but often relies on side information, e.g., context [34] and review text [11], or probes users for more data [3, 2]. In our approach, we make use of *given rating data only.* Also related is research in transfer learn-

*A portion of this work was done while at Carnegie Mellon University.
†A portion of this work was done while at Google, Inc.

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052713

	Focused Learning	LCE [33]	Anava et al. [3]	ExcUseMe [2]	Park et al. [27]	Yin et al. [39]	CD-CCA [31]	CDTF [16]
Doesn't Require Side Information	✓	×	✓	✓	✓	✓	✓	✓
Doesn't Require User Interaction	✓	✓	×	×	✓	✓	✓	✓
Works on top of other CF methods	✓	×	×	×	×	×	×	×
Arbitrary focus groups	✓	×	×	×	×	×	×	×
– Long-tail	✓	✓	✓	✓	✓	✓	×	×
– Sub-domain (e.g., genre)	✓	×	×	×	×	×	✓	✓
– "Outliers"	✓	×	×	×	×	×	×	×

Table 1: Comparison with related work.

ing and cross-domain recommendation, which often designs models that include a transfer function between domains [31, 16]. Rather, we frame the problem as a hyperparameter optimization challenge: we focus on finding hyperparameters for our model that offer the best performance for a pre-specified subset of items. This allows our algorithm to continue to work with new, state-of-the-art recommendation models as they are developed.

Through this simple approach, we significantly improve recommendation accuracy for multiple challenging groups, including (A) niche genres, (B) items for which we have few observations, and (C) even the most badly-modeled items from the original model (items typically thought of as anomalous or outliers that are ignored). In particular, we achieve a greater than 17% improvement on items for which we have few ratings, as seen in Figure 1.

In this paper we offer the following contributions:

- **Problem formulation:** We define the focused learning problem, giving empirical and theoretical evidence that a single factorization, under current state-of-the-art methods, will under-serve some items and users.
- **Algorithm:** We offer a novel algorithm for focused learning that can work with multiple state-of-the-art recommender systems.
- **Real-world experiments:** We give empirical evidence, across multiple datasets and recommender models, that our focused learning algorithm improves prediction accuracy for a variety of focused items.

2. RELATED WORK

Our research, while providing a new perspective, is related to many other areas of data mining and machine learning. An overview of how the most closely related work relates to our research can be seen in Table 1.

Recommender systems: A plethora of research has focused on predicting preferences based on explicit feedback, e.g., user ratings of items, or implicit feedback, user interactions with items. Spurred by the Netflix Prize, much work was devoted to creating different factorization models that better fit ratings data [17, 18, 19]. In addition to modeling rating matrices, researchers have proposed learning preferences from more complex sets of user feedback [37, 41]. Some models combine clustering techniques with recommender systems [7, 40]. However, these changes are not enough to overcome the challenges of optimizing a single, global objective within one model.

Local Models & Ensembling: Recent work learns local models [10], and in some cases ensembles local learners [6, 21]. With respect to recommendation, we show experimentally in §7.1 that local models do not addresses the accuracy skew in our data, but that we can improve the accuracy by applying our focused learning algorithm to ensembles of local models [21]. More broadly, much of the work on ensemble learning has focused on backfitting [8] or functional gradient descent [24] to build combined models based on error in the training data. Here, we focus on building new models based on held-out prediction errors.

Cross-Domain Recommendation: An additional line of research has focused on transfer learning, cross-domain, and multi-task recommendation, all slight variants related to focused learning. These methods often use side information, such as item content [13], to improve recommendation. However, an additional line of work has used purely collaborative filtering (CF) approaches. This models often learn a transfer function between domains, such as with canonical correlation analysis (CCA) [31] or tensor factorization [16]. Our work differs in that it can work with many different recommender systems, including new ones such as LLORMA, rather than being tied to a specific model structure.

Cold-Start Recommendation: One commonly studied case where classic CF fails is in the case of "cold-start" users or items, that is users or items for which we have very few or no observed ratings. Most work in this area relies on side information about the items [34, 4, 20, 33], users' social networks [23], multi-task learning to model review text [11, 25], or actively probing users for ratings [3, 2]. Recently, researchers have examined the cold-start problem using just ratings data [30, 39], but aren't able to build on state-of-the-art CF methods.

Another line of work has focused on recommending in the tail. Often this aims to surface novel recommendations as measured by coverage in tail [35], rather than accuracy as is our focus, and often addresses the challenge again with additional contextual data [15]. [27] takes a local model approach, focusing on how to cluster the items in the tail.

Hyperparameter Optimization: A significant amount of research has focused on using machine learning to optimize the hyperparameters of other machine learning models [5], e.g., using Bayesian learning and Gaussian processes to estimate model hyperparameters [9, 36]. Our method follows this hyperparameter optimization perspective but does not directly implement these algorithms. Rather, more complex hyperparameter optimization algorithms could be directly applied to improve the precision and speed of our method.

Handling noisy data: Our custom regularization is related, in intuition, to previous research on learning confidence, such as through directly measuring confidence [12, 38], Bayesian CF [32, 7], or transforming ratings in CF [17]. We believe incorporating these more complex techniques for measuring confidence could offer additional performance improvements when combined with our general focused learning system.

Symbol	Definition
$r_{i,j}$	Rating from user i to item j from set $\{1 \dots R\}$
$\mathcal{R}$	Set of ratings $r_{i,j}$
n, m	Number of users and items, respectively
$\mathcal{R}^{\text{Train}}$	Training data; subset of $\mathcal{R}$
$\mathcal{R}^{\text{Val.}}$	Validation data; subset of $\mathcal{R}$
$\mathcal{R}^{\text{Test}}$	Test data; subset of $\mathcal{R}$
$\mathcal{I}$	Set of items to focus on
$\mathcal{R}^{\text{Test}}_{\mathcal{I}}$	Set of ratings from $\mathcal{R}^{\text{Test}}$ for which $j \in \mathcal{I}$
n_j	Number of observed ratings for item j
$\text{RMSE}_{\mathcal{R}}$	Root mean squared error for observations in $\mathcal{R}$
k	Rank of the factorization
λ	Weight for regularization
$\hat{r}_{i,j}$	Model's prediction for $r_{i,j}$

Table 2: Notation used in this paper

3. PROBLEM DEFINITION

We begin with an overview of our notation and problem setup. A complete list of symbols used in this paper can be found in Table 2. We consider the case where we have ratings $r_{i,j} \in \{1 \dots R\}$ where $i \in \{1 \dots n\}$ indexes the users and $j \in \{1 \dots m\}$ indexes the items, where we say $r_{i,j} \in \mathcal{R}$ if we have an observed rating $r_{i,j}$. For convenience we can also view the data as a matrix $\mathcal{R} \in \mathbb{R}^{n \times m}$, where most values in the matrix are missing.

As with most recommenders, our general goal is to learn a model from training data that predicts the missing values in the matrix. Following convention in prediction tasks, we split our data into training $\mathcal{R}^{\text{Train}}$ and testing $\mathcal{R}^{\text{Test}}$ data, where $\mathcal{R}^{\text{Train}} \cup \mathcal{R}^{\text{Test}} = \mathcal{R}$ and $\mathcal{R}^{\text{Train}} \cap \mathcal{R}^{\text{Test}} = \emptyset$. We assume that both $\mathcal{R}^{\text{Train}}$ and $\mathcal{R}^{\text{Test}}$ come from the same distribution, i.e., from a random split of $\mathcal{R}$.

We will typically evaluate the quality of our model and its prediction based on its test RMSE:

$$\text{RMSE}_{\mathcal{R}^{\text{Test}}} = \sqrt{\frac{1}{|\mathcal{R}^{\text{Test}}|} \sum_{r_{i,j} \in \mathcal{R}^{\text{Test}}} (r_{i,j} - \hat{r}_{i,j})^2} \qquad (1)$$

Here $\hat{r}_{i,j}$ is the model's prediction for $r_{i,j}$.

Focused Learning Problem.

In focused learning, we want to have high prediction accuracy on a subset of the data. Given a set of items $\mathcal{I}$, we denote by $\mathcal{R}_{\mathcal{I}}$ the set of observations from items in $\mathcal{I}$; precisely $\mathcal{R}_{\mathcal{I}} = \{r_{i,j} | r_{i,j} \in \mathcal{R} \wedge j \in \mathcal{I}\}$. With this notation, we can precisely specify our problem:

Problem Definition 1 (Focused Learning)**.**
Given: *Data* $\mathcal{R}^{\text{Train}}$ *and a subset of the items to focus on* $\mathcal{I}$
Find: *A model that has high prediction accuracy on* $\mathcal{R}^{\text{Test}}_{\mathcal{I}}$*, the test data for the focus set.*

We will consider prediction error to be measured by RMSE, but the formulation could easily accommodate other metrics.

Implicit in the problem definition above is the focus set $\mathcal{I}$. For the focused learning problem we would like to be able to take any focus set. However, in practice, there are many ways to select the focus set, and we will analyze how different focus selection techniques impact the quality of results.

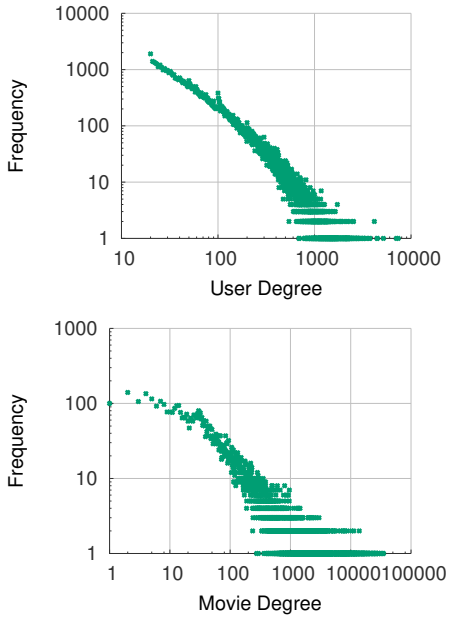


Figure 2: Log-log plot of the heavy-tail distribution of observations in MovieLens.

4. DATA EXPLORATION

While recommender systems are pervasive, they often treat the data as generic matrices, rather than accounting for many of the common biases in such data.

Heavy-Tail Distribution of Observations.

One of the most common properties throughout the web is heavy-tail distributions [14, 1, 26]. This pattern shows up in ratings data where most movies receive few ratings and a few popular movies receive many ratings; similarly, most users give few ratings and a few users give many ratings [27]. In Figure 2, we observe this distribution in MovieLens.

This pattern has significant implications: First, because most objectives optimize the average error across all observations, items with more observed ratings are considered more important than items with fewer observed ratings, and as a result, the allocation of model capacity is biased toward popular items. Second, because we want to have low RMSE for all observations in the test set, the heavy-tail distribution of test data biases the model selection.

Skewed Predictions.

For the reasons described above, the model with the best average predictive accuracy will leave meaningful subsets of items modeled significantly worse than other subsets. To demonstrate this, we calculate the *per-user* and *per-item* prediction error, $\text{RMSE}_{\mathcal{R}^{\text{Test}}_j}$. In Figure 3, we plot the number of users and items with each MSE (size step of 0.1). We observe a long tail of users and items with low prediction accuracy. In addition, this is not just based on degree; even for users or items with many observations, there is a long tail of prediction error. Prediction skew is particularly problematic because it means that these users and items will have a significantly worse experience under these models.

In addition, we find that some genres of movies receive significantly worse performance. As can be seen in Figure

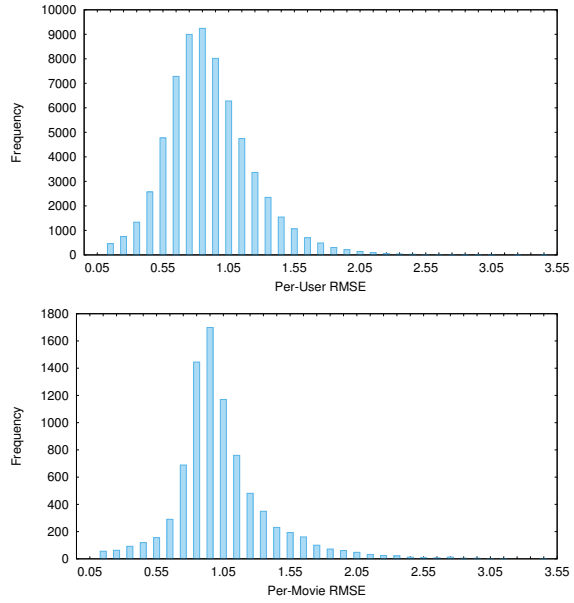


Figure 3: Many users and movies are badly-modeled.

4(a), IMAX movies and Musicals on MovieLens are modeled much worse than Film-Noir movies. Perhaps surprisingly, in Figure 4(b) we see that this pattern is not just an artifact of data sparsity, because Musicals actually have a relatively high number of observations per movie, but still have worse prediction accuracy compared to other genres. Having some genres badly-modeled may not be terrible, but we can imagine a much more problematic situation—If we were predicting users' preferences for apps, and apps for some language/country had significantly worse performance than other languages/countries, then the recommender system would offer worse performance to entire populations of people.

Theoretical Justification.

We can also understand the need for focused learning from a theoretical perspective. We have a model $\mathcal{M}$ with parameters θ trying to fit data $\mathcal{R}$ under loss $\mathcal{L}_{\mathcal{R}}(\mathcal{M}_\theta)$, and we assume our loss function is an average over instances in $\mathcal{R}$:

$$\arg\min_\theta \mathcal{L}_{\mathcal{R}}(\mathcal{M}_\theta) = \arg\min_\theta \frac{1}{|\mathcal{R}|} \sum_{(\mathbf{x},y)\in\mathcal{R}} L\left(y, \mathcal{M}_\theta(\mathbf{x})\right) \quad (2)$$

where $L : \mathbb{R}\times\mathbb{R} \to \mathbb{R}_{\geq 0}$ is the per-instance loss, and if $L(y, \mathcal{M}_\theta(\mathbf{x})) > 0$ then $\frac{\partial L(y,\mathcal{M}_\theta(\mathbf{x}))}{\partial\theta} \neq 0$.

We consider θ^* to be the parameters for the global optimal solution to eq. (2). We assume the optimal model has non-zero loss, i.e., $\mathcal{L}_{\mathcal{R}}(\mathcal{M}_{\theta^*}) > 0$.

Theorem 1 (Global optimal not locally optimal)**.** *For dataset $\mathcal{R}$ and loss function $\mathcal{L}_{\mathcal{R}}(\mathcal{M}_\theta)$ with optimal parameters θ^* and $\mathcal{L}_{\mathcal{R}}(\mathcal{M}_{\theta^*}) > 0$; there exists $\mathcal{R}' \subset \mathcal{R}$ such that θ^* is not the optimal solution to $\mathcal{L}_{\mathcal{R}'}(\mathcal{M}_\theta)$.*

Proof. Because θ^* is the global optimal, we know

$$\frac{\partial\mathcal{L}_{\mathcal{R}}(\mathcal{M}_{\theta^*})}{\partial\theta} = \sum_{(\mathbf{x},y)\in\mathcal{R}} \frac{\partial L(y,\mathcal{M}_{\theta^*}(\mathbf{x}))}{\partial\theta} = 0.$$

Because $\mathcal{L}_{\mathcal{R}}(\mathcal{M}_{\theta^*}) > 0$, there exists $p = (\hat{\mathbf{x}}, \hat{y}) \in \mathcal{R}$ such that $L(\hat{y}, \mathcal{M}_{\theta^*}(\hat{\mathbf{x}})) > 0$. From this, we find

$$\frac{\partial\mathcal{L}_{\mathcal{R}\setminus p}(\mathcal{M}_{\theta^*})}{\partial\theta} = \sum_{(\mathbf{x},y)\in\mathcal{R}} \frac{\partial L(y,\mathcal{M}_{\theta^*}(\mathbf{x}))}{\partial\theta} - \frac{\partial L(\hat{y},\mathcal{M}_{\theta^*}(\hat{\mathbf{x}}))}{\partial\theta} \neq 0$$

Therefore, θ^* is not optimal for $\mathcal{R}' = \mathcal{R}\setminus p$. ■

Therefore, the globally optimal model is typically not the best model for each part of the data. Rather, learning multiple models, with each model optimized to improve prediction quality for different subsets of data, should yield better results than relying on the global optimum.

5. OUR METHOD

We now describe our focused learning algorithm. For clarity our description uses the language and notation of classic MF, but we will demonstrate in §6.4 and §6.5 that the method can be easily applied to other CF models:

$$\arg\min_{U,V} \sum_{r_{i,j}\in\mathcal{R}^{\text{Train}}} w_j(r_{i,j} - \langle u_i, v_j\rangle)^2 + \lambda(\|U\|_2^2 + \|V\|_2^2) \quad (3)$$

Here $U \in \mathbb{R}^{n\times k}$ and $V \in \mathbb{R}^{m\times k}$, w_j weights column j, and λ weight the regularization. In some cases, we will separate the regularization into λ_u and λ_v. We consider w_j, k, λ_u and λ_v all as hyperparameters that can be tuned.

Given U and V we can calculate the RMSE over $\mathcal{R}^{\text{Test}}$ by eq. (1) where $\hat{r}_{i,j} = \langle u_i, v_j\rangle$. We use the notation $\mathcal{A}_{\mathcal{R}^{\text{Train}},\mathcal{R}^{\text{Test}}}(k, \lambda_u, \lambda_v, \vec{w})$ to denote an algorithm that learns U and V from $\mathcal{R}^{\text{Train}}$ and outputs $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$.

5.1 Focused Hyperparameter Optimization

To solve the focused learning problem, we formulate it as a hyperparameter optimization challenge. With a slight abuse of notation, for hyperparameter optimization we split our training data into $\mathcal{R}^{\text{Train}}$ and $\mathcal{R}^{\text{Val.}}$, such that we use $\mathcal{R}^{\text{Train}}$ to learn our model and $\mathcal{R}^{\text{Val.}}$ to check how well our model is performing before officially testing on $\mathcal{R}^{\text{Test}}$.

Hyperparameter optimization typically optimizes:

$$\min_{k,\lambda_u,\lambda_v,\vec{w}} \mathcal{A}_{\mathcal{R}^{\text{Train}},\mathcal{R}^{\text{Val.}}}(k, \lambda_u, \lambda_v, \vec{w}) \quad (4)$$

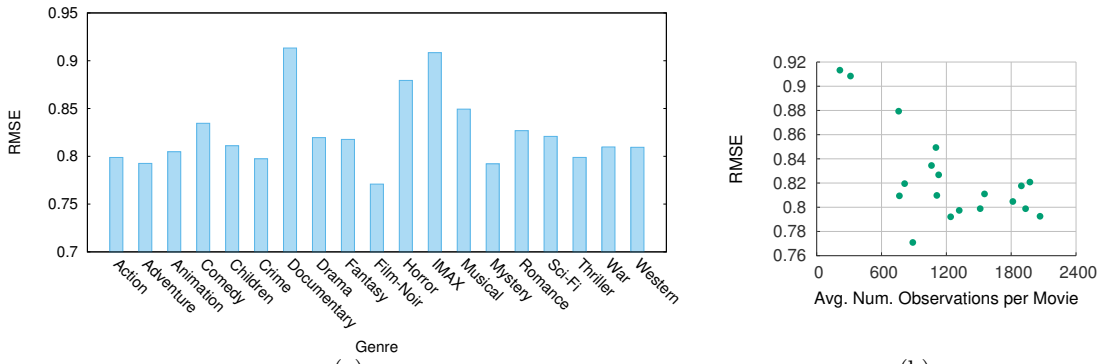
The goal of this optimization is that if the accuracy on $\mathcal{R}^{\text{Val.}}$ improves through changing k, λ_u, λ_v and $\vec{w}$, then the accuracy on $\mathcal{R}^{\text{Test}}$ should improve too.

For focused learning, our objective is to improve the prediction accuracy on $\mathcal{R}_{\mathcal{I}}^{\text{Test}}$, for a predefined set of items $\mathcal{I}$. Therefore, we change our hyperparameter optimization to:

$$\min_{k,\lambda_u,\lambda_v,\vec{w}} \mathcal{A}_{\mathcal{R}^{\text{Train}},\mathcal{R}_{\mathcal{I}}^{\text{Val.}}}(k, \lambda_u, \lambda_v, \vec{w}) \quad (5)$$

While it is difficult to prioritize $\mathcal{R}_{\mathcal{I}}^{\text{Train}}$ in eq. (3), it is clear that we can optimize our hyperparameters for $\mathcal{R}_{\mathcal{I}}^{\text{Val.}}$ with the goal of improving the accuracy for $\mathcal{R}_{\mathcal{I}}^{\text{Test}}$.

We will demonstrate that a simple grid search yields improvements in prediction accuracy for our focus sets $\mathcal{I}$. Of course, one could also apply research on hyperparameter optimization [9, 5]. In grid search, each run is independent and thus we can trivially parallelize our learning over different hyperparameter settings and different focus sets $\mathcal{I}$.



(a) (b)

Figure 4: In a standard model, we observe that (a) some genres are modeled significantly better than others for the MovieLens data, and (b) these patterns do not just follow number of observations (degree).

5.2 Focused Learning

The above approach already offers a framework for focused learning, and as we will see, improves model accuracy. However, we find that by slightly modifying our underlying objective, we obtain hyperparameters more useful for focused learning. We allow for different regularization of items that are our focus and items that are not our focus:

$$\arg\min_{U,V} \sum_{r_{i,j}\in\mathcal{R}^{\text{Train}}} w_j(r_{i,j} - \langle u_i, v_j\rangle)^2 + \lambda_u\|U\|_2^2 + \lambda_{\text{focus}}\sum_{j\in\mathcal{I}}\|v_j\|_2^2 + \lambda_{\text{unfocus}}\sum_{j\notin\mathcal{I}}\|v_j\|_2^2 \quad (6)$$

We denote this algorithm by $\mathcal{A}'_{\mathcal{R}^{\text{Train}},\mathcal{R}^{\text{Val.}},\mathcal{I}}$. Thus, our new hyperparameter optimization is:

$$\min_{k,\lambda_u,\lambda_{\text{focus}},\lambda_{\text{unfocus}}} \mathcal{A}'_{\mathcal{R}^{\text{Train}},\mathcal{R}^{\text{Val.}}_{\mathcal{I}},\mathcal{I}}(k,\lambda_u,\lambda_{\text{focus}},\lambda_{\text{unfocus}}) \quad (7)$$

The intuition behind this new objective is that the focus set may need a different regularization than the unfocused set, since regularization controls the degree in which the model generalizes. The advantage of this formulation is that our parameterization is customized to the focus set, and as we will see this single extra parameter gives an additional significant gain in prediction accuracy, along with interesting insights into the role of regularization in MF.

6. EXPERIMENTS

We now demonstrate the success of our approach through a variety of experiments on real-world data. In testing focused learning, we apply it to weighted alternating least squares (ALS) MF [17], factorization machines [28] and LLORMA [21]. We primarily test our method on the MovieLens data set [29], where there are over 10 million ratings from 71567 users for 10681 movies. In this data set, all users have given at least 20 ratings. We split the ratings into training, validation, and test sets using a random 80%-10%-10% split.

Global Baseline Model. For the global baseline model from eq. (3), all hyperparameters were first hand-tuned to give optimal global results on the holdout data, obtaining regularization $\lambda_u = \lambda_v = 30$ and rank $k = 35$. The column weights w_j are tuned to be proportional to $\frac{1}{(n_j+1)^{0.3}}$, where n_j is the number of observations in column j.

Focusing Models. Our goal is to improve and report the test RMSE for focus sets. For most of our experiments we focus the hyperparameter search on λ_{focus} and λ_{unfocus}. For both we perform a grid search over $\lambda_{\text{focus}}, \lambda_{\text{unfocus}} \in \{3, 15, 30, 60, 150, 300\}$. When testing LLORMA, we also search over rank k and the number of local models.

We use three different methods for creating focus sets $\mathcal{I}$. We primarily compare the results of focused hyperparameter search from eq. (5) and focused learning in eq. (7), both learned using weighted ALS [17], against the test RMSE for the focus set from the globally optimal model. We test focused learning with LibFM in §6.4 and LLORMA in §6.5; in §7.1 we compare to intuitive but less successful baselines.

6.1 Focusing on Cold-Start Items

Because one of the motivating observations is the heavy-tail distribution of ratings, we begin with trying to improve the prediction accuracy for items with few ratings. To test this, we group movies into deciles, i.e., 10-percentile buckets, based on each movie's number of ratings (degree) n_j, i.e., the first group contains all movies with $n_j \in [1, 11)$, our second contains all movies with $n_j \in [11, 26)$, etc.

As we see in Table 3 and Figure 5, we achieve large accuracy improvements with degree-based focus selection. For the items with the fewest observed ratings, we are able to improve prediction accuracy by over 13% and for the second group we achieve an improvement of over 16%. This is particularly notable because we are not using any additional data or context, as is common in research on improving prediction quality for cold-start items.

Furthermore, Figure 5 shows how focused learning achieves better performance than focused hyperparameter search alone. This demonstrates the importance of not just finding the right hyperparameters, but also giving the model the necessary flexibility. We will further explore the role of regularization in §7.2.

6.2 Focusing on Outliers

We next test our ability to improve the prediction quality for items that were originally modeled badly. Typically, these items would be considered outliers or too noisy because they do not conform to the model. However, we demonstrate that we are able to create models that greatly improve prediction quality for these items.

To create our focus sets we take our global optimal model and get the validation error $\text{RMSE}_{\mathcal{R}^{\text{Val.}}_j}$ for each item j. We then group items into deciles based on their validation RMSE.

			Focused Hyperparameter Search			**Focused Learning**			
Degree Range	Percentile Range	Original $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Optimal $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Percent Improved	Optimal λ_v	Optimal $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Percent Improved	Optimal λ_{focus}	Optimal λ_{unfocus}
$[1, 11)$	0%–10%	1.229433	1.1513	6.3572%	3	**1.099444**	**10.5731%**	3	30
$[11, 26)$	10%–20%	1.355938	1.2633	6.8348%	3	**1.120485**	**17.3646%**	3	30
$[26, 45)$	20%–30%	1.254085	1.1713	6.6016%	3	**1.078881**	**13.9707%**	3	60
$[45, 78)$	30%–40%	1.127054	1.0529	6.5804%	3	**0.976097**	**13.3939%**	3	30
$[78, 135)$	40%–50%	1.053422	1.0044	4.6507%	3	**0.982389**	**6.7431%**	15	150
$[135, 233)$	50%–60%	0.970897	0.9460	2.5622%	15	**0.918135**	**5.4344%**	15	60
$[233, 444)$	60%–70%	0.921469	0.9062	1.6603%	15	**0.888497**	**3.5782%**	15	60
$[444, 926)$	70%–80%	0.885365	0.8786	0.7647%	15	**0.870866**	**1.6376%**	15	60
$[926, 2388)$	80%–90%	0.846424	0.8464	0%	30	**0.842873**	**0.4195%**	15	60
$[2388, \infty)$	90%–100%	0.800346	0.8003	0%	30	**0.799343**	**0.1253%**	30	60

Table 3: Focusing on Cold-Start Items: Improvements in test RMSE from focused learning on items with fewest observations in MovieLens.

			Focused Hyperparameter Search			**Focused Learning**			
Per-Movie $\text{RMSE}_{\mathcal{R}^{\text{Val.}}}$	Percentile Range	Original $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Optimal $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Percent Improved	Optimal λ_v	Optimal $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Percent Improved	Optimal λ_{focus}	Optimal λ_{unfocus}
$[1.37, \infty)$	0%–10%	1.1804	1.1475	2.7835%	3	**1.0696**	**9.3903%**	3	150
$[1.13, 1.37)$	10%–20%	1.0739	1.0733	0.0579%	15	**1.0405**	**3.1095%**	15	150
$[1.01, 1.13)$	20%–30%	1.0014	1.0014	0%	30	**0.9970**	**0.4429%**	30	150
$[0.93, 1.01)$	30%–40%	0.9367	0.9367	0%	30	0.9378	-0.1218%	30	60
$[0.87, 0.93)$	40%–50%	0.8850	0.8850	0%	30	0.8850	0%	30	30
$[0.83, 0.87)$	50%–60%	0.8450	0.8450	0%	30	0.8450	0%	30	30
$[0.78, 0.83)$	60%–70%	0.8063	0.8063	0%	30	0.8063	0%	30	30
$[0.72, 0.78)$	70%–80%	0.7605	0.7605	0%	30	**0.7594**	**0.1402%**	30	60
$[0.62, 0.72)$	80%–90%	0.7142	0.7132	0.1400%	15	**0.7101**	**0.5634%**	30	60
$[0, 0.62)$	90%–100%	0.7947	0.7802	1.8279%	15	**0.7688**	**3.2601%**	15	30

Table 4: Better modeling of Outliers: Focused learning improves prediction error (test RMSE) for the worst modeled movies in MovieLens.

		Focused Hyperparameter Search			**Focused Learning**			
Genre	Original $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Optimal $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Percent Improved	Optimal λ_v	Optimal $\text{RMSE}_{\mathcal{R}^{\text{Test}}}$	Percent Improved	Optimal λ_{focus}	Optimal λ_{unfocus}
Action	0.7988	0.7988	0%	30	**0.7959**	**0.3643%**	30	60
Adventure	0.7926	0.7926	0%	30	**0.7905**	**0.2643%**	30	60
Animation	0.8048	0.8048	0%	30	**0.7966**	**1.0110%**	30	150
Comedy	0.8345	0.8345	0%	30	**0.8340**	**0.0650%**	30	60
Children	0.8110	0.8110	0%	30	**0.8054**	**0.6984%**	30	150
Crime	0.7974	0.7974	0%	30	**0.7961**	**0.1639%**	30	60
Documentary	0.9133	0.9060	0.7922%	15	**0.8692**	**4.8238%**	15	150
Drama	0.8195	0.8195	0%	30	**0.8187**	**0.0952%**	30	60
Fantasy	0.8177	0.8177	0%	30	**0.8165**	**0.1422%**	30	60
Film-Noir	0.7709	0.7702	0.08821%	15	**0.7665**	**0.5727%**	15	150
Horror	0.8794	0.8858	-0.7270%	15	**0.8771**	**0.2582%**	30	60
IMAX	0.9084	0.9013	0.7866%	15	**0.8865**	**2.4165%**	15	60
Musical	0.8493	0.8493	0%	30	**0.8452**	**0.4937%**	30	150
Mystery	0.7921	0.7921	0%	30	0.7921	0%	30	30
Romance	0.8268	0.8268	0%	30	0.8268	0%	30	30
Sci-Fi	0.8208	0.8208	0%	30	**0.8191**	**0.2136%**	30	60
Thriller	0.7988	0.7988	0%	30	**0.7979**	**0.1198%**	30	60
War	0.8098	0.8098	0%	30	**0.8083**	**0.1818%**	30	150
Western	0.8094	0.8094	0%	30	**0.8014**	**0.9934%**	15	150

Table 5: Focusing with Side Information: Improvements in test RMSE from focused learning on each genre in MovieLens.

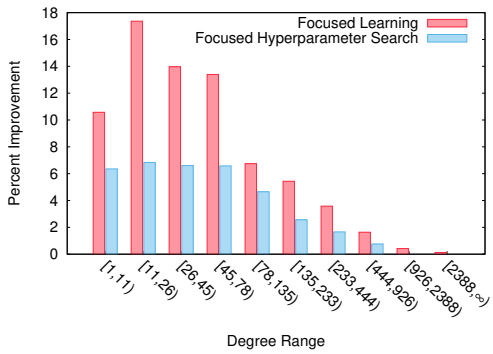


Figure 5: Focusing on Cold-Start Items: Improvement from focused learning on cold-start items in MovieLens.

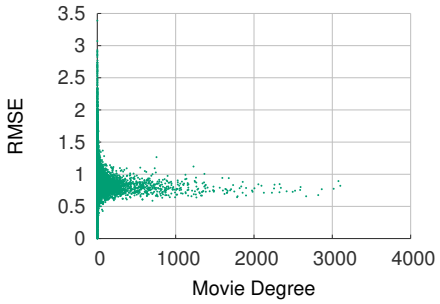


Figure 6: We observe especially high variance in prediction errors for movies with fewer observations.

As can be seen in Table 4, we are able to improve prediction quality for the previously badly-modeled movies. We improve the RMSE for the movies in the first group ($\text{RMSE}_{\mathcal{R}_j^{\text{Val.}}} \geq 1.37$) by nearly 10%.

Surprisingly, we also observe a 3.26% improvement in test RMSE for the previously *best* modeled movies—those with $\text{RMSE}_{\mathcal{R}_j^{\text{Val.}}} < 0.62$. Upon further investigation we find that at either extreme, movies with very high or very low RMSE, are mostly movies with fewer ratings in our hold-out set. This is more clearly visualized in Figure 6, which shows that movies with low degree have high variance in prediction error.[1] Because our focused learning approach is based on its regularization, it is most apt to handle data sparsity challenges.

6.3 Focusing with Side Information

In many real-world applications, we have additional information about the items on which we are predicting. Using this side information as our focus selection criteria is a natural choice. Here, we make use of the genre for each movie. For example, we might want to specifically improve the predictions for Documentaries due to product direction concerns. For each genre we consider all movies that are in that genre and use the typical focused learning algorithm.

[1]This also helps explain why the original test RMSE for the 90%-100% decile group (0.7947) is higher than for next two "worse"-modeled decile groups (0.7142 and 0.7605). That is, grouping is performed based on validation RMSE $\text{RMSE}_{\mathcal{R}^{\text{Val.}}}$, but we measure the test RMSE. When there are sparser observations for a focus set, the difference between validation and test RMSEs is higher.

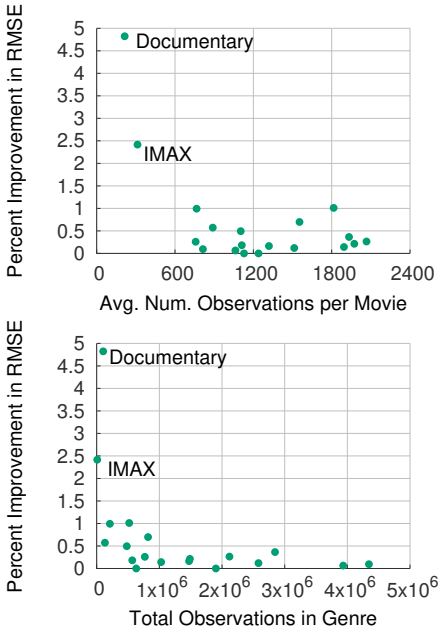


Figure 7: Analysis of the improvement from focusing on genres in MovieLens.

In MovieLens, because each movie can be part of more than one genre, a movie may be in more than one focus set.

As we see in Table 5, we observe improvements in the prediction error for all genres, with the largest improvements for IMAX films and Documentaries. Beyond the raw improvements, we observe a few interesting patterns. For all genres, we find that the optimal regularization for the focus set (λ_{focus}) is low while the optimal regularization for the rest of the movies (λ_{unfocus}) is high.

We also plot, in Figures 7, the improvement in performance as a function of the total number of observations in each movie as well as the data sparsity in each genre. Unsurprisingly, documentaries and IMAX movies, the two genres with the largest improvement, have the fewest observations and the most sparse data. This makes sense because genres with few ratings have the least influence in the selection of a global optimal regularization parameter. In case of data sparsity, we know that regularization is important to set correctly. However, beyond those two genres, we observe that improvements in accuracy are largely independent of data size and sparsity. Therefore, while our method excels at helping cold-start movies, as we saw above, it also helps large genres with popular movies.

6.4 Focusing in more complex models: LibFM

To demonstrate that focused learning will continue to be useful under more complex models, we test our approach on multiple model structures to improve predictions for MovieLens split by item degree (as in §6.1). We build on LibFM [28], with three increasingly complex settings.

A summary of our results with LibFM can be seen in Table 6. Each test consists of a new variation of offsets for prediction and set of hyperparameters that we tune during focused learning. In each case we compare our model with globally optimized hyperparameters to our model with focused hyperparameters. Across these experiments we find that focused learning can still offer significant improvements.

Prediction $\hat{r}_{i,j}$	Regularization	Focus Parameters	Max. Improvement
$\mu+\langle u_i, v_j\rangle$	$\lambda_\mu\mu^2+\lambda_u\|U\|_2^2+\lambda_v\|V\|_2^2$	$\lambda_{\text{focus } v}, \lambda_{\text{unfocus } v}$	8.9%
$\mu+a_i+b_j+\langle u_i, v_j\rangle$	$\lambda_\mu\mu^2+\lambda_a\|a\|_2^2+\lambda_b\|b\|_2^2+\lambda_u\|U\|_2^2+\lambda_v\|V\|_2^2$	$\lambda_{\text{focus } v}, \lambda_{\text{unfocus } v}$	1.85%
$\mu+a_i+b_j+\langle u_i, v_j\rangle$	$\lambda_\mu\mu^2+\lambda_a\|a\|_2^2+\lambda_b\|b\|_2^2+\lambda_u\|U\|_2^2+\lambda_v\|V\|_2^2$	$\lambda_{\text{focus } v}, \lambda_{\text{unfocus } v}, \lambda_{\text{focus } b}$	2.25%

Table 6: Maximum improvements of focused learning under different objective functions.

When we add per-user and per-item offsets, the baseline model is improved, covering some of the improvements previously offered by focused learning. With these more complex models, focused learning overfits for items with the fewest observations due to the small validation set. However, once the validation set is larger, focused learning still improves over even the more complex model. Last, by performing focused learning on the regularization of the item-offsets, we gain *additional improvements* in prediction accuracy. This demonstrates that even when using a more complex model, focused learning can improve prediction accuracy and focusing additional parts of your model can provide *additional* improvements.

6.5 Focused Learning in LLORMA

We apply the focused hyperparameter optimization algorithm of §5.1 on top of the LLORMA implementation in PREA [22]. Here, we use as a baseline the hyperparameter settings from [21], which we also verify to be optimal under a global objective. We test regularization $\lambda \in \{0.1, 0.01, 0.001, 0.0001, 0.00001\}$ and rank $\in \{5, 10, 20\}$; the implementation automatically chooses a number of local models with the best validation accuracy for $\leq q = 50$. We run our experiments on the MovieLens 1M dataset [29], due to speed and memory limitations of the PREA implementation. We achieve an improvement in test accuracy of 3.7% for the items with the 10% fewest ratings and of 1.4% for items in the second decile. This further demonstrates that our focused learning algorithm benefits from further optimizing other hyperparameters, and still offers significant improvements, even on top of new, more complex models.

We also test our method to improve the prediction accuracy for users with the fewest ratings. We achieve a 0.27% improvement for users with the 10% fewest ratings and a 0.65% improvement for users in the second decile. We believe this is smaller than the item improvements because the dataset only includes users with ≥ 20 ratings, thus cutting off much of the tail. Therefore, while the magnitude of the improvements on these users is smaller, it still presents strong evidence of the generalizability of the method.

6.6 Recommendation at Google

To demonstrate how well focused learning works beyond MovieLens, we test our framework to improve a collaborative filtering model at Google. The model is currently serving over one billion monthly active users in a recommender system. Our training matrix, constructed from user engagement data, has 3 million rows, 1 million columns and *79 million* observed values. There is a heavy-tail distribution of observations among both the rows and the columns, and the rows were previously filtered so that each row contains at least 11 observations. We use Google's production settings for the recommender system as the baseline and the initial hyperparameters, with $\lambda_u = \lambda_v = 15$, rank $k = 100$, and weights w_j proportional to $\frac{1}{(n_j+1)^{0.5}}$. We take an 80%-10%-10% split of the data and create focus groups by partitioning the columns according to their degree.

As seen in Table 7, we observe that our approach offers consistent improvements over Google's production system, with up to a 4% improvement in accuracy. These experiments offer a strong reaffirmation of our approach.

7. WHY DOES THIS WORK?

In this section, we explore a wide variety of perspectives to better understand why focused learning works.

7.1 Alternative Baselines

So far we have primarily compared our focused learning algorithm to a globally-optimized model. However, our focused learning approach was not the first idea attempted. Therefore, we now compare against other potential approaches.

Doubling the model size. By making additional focused models, we are increasing the total model size. To demonstrate that focused learning increases model size in a principled way, we made a model twice as large as our globally optimal model and compared the RMSE on specific focus sets. To be precise, we tuned our λ (jointly $\lambda_u = \lambda_v$) for a global model of rank 70 and evaluated the test RMSE for documentaries, movies with degree $[1, 11)$, and movies with degree $[11, 26)$. For these three categories we observe a test RMSE of 0.9169, 1.2359, and 1.3615, respectively. In all three cases the RMSE from the doubly-large but globally optimized model is *worse* than our original rank-35 global model and also significantly worse than our focused learning models. Therefore, focused learning is not improving results merely by having a larger model, but because the additional model size is allocated well for the focused set of items.

Training local models. Second, we test how well local models perform on this problem. Intuitively, local models might work well for semantically similar content. Therefore, for a given focus set $\mathcal{I}$, we only use $\mathcal{R}_{\mathcal{I}}^{\text{Train}}$ when training the model, we tune λ_v based on the RMSE of $\mathcal{R}_{\mathcal{I}}^{\text{Val.}}$, and then we test against $\mathcal{R}_{\mathcal{I}}^{\text{Test}}$. We test this on Action, Animation, Documentaries and Westerns genres, with test RMSE for the four groups of 0.8473, 0.9368, 1.1133, 1.0925, respectively. Across all four genres, the test RMSE from the local model is worse than the RMSE from the baseline model, and much worse than the focused learning RMSE.

7.2 Exploring Regularization

Given focused learning modifies the model regularization, it is worthwhile to explore the resulting regularization patterns. Most notably, we find that across nearly every experiment, $\lambda_{\text{focus}} \leq \lambda_v$ and $\lambda_{\text{unfocus}} \geq \lambda_{\text{focus}}$, where λ_v is the globally optimal regularization. This is particularly interesting when we consider groups of items for which we observe few ratings. From one perspective, we would expect high regularization for items with few observations to prevent overfitting. From another perspective, an observation for a less-popular item is more valuable than an observation for a popular item, so we expect less regularization so as

Degree Range	Percentile Range	Original $\text{RMSE}_{\mathcal{R}\text{Test}}$	Focused Hyperparameter Search: Optimal $\text{RMSE}_{\mathcal{R}\text{Test}}$	Percent Improved	Optimal λ_v	**Focused Learning**: Optimal $\text{RMSE}_{\mathcal{R}\text{Test}}$	Percent Improved	Optimal λ_{focus}	Optimal λ_{unfocus}
$[1, 2]$	0%–40%	1.4155	1.4113	0.30%	7.5	**1.3777**	**2.67%**	1.5	7.5
$[3, 3]$	40%–50%	1.5097	1.5063	0.23%	7.5	**1.4818**	**1.85%**	1.5	15
$[4, 6]$	50%–60%	1.5795	1.5722	0.46%	7.5	**1.5308**	**0.46%**	1.5	7.5
$[7, 12]$	60%–70%	1.6788	1.6785	0.02%	7.5	**1.6101**	**4.09%**	1.5	15
$[13, 27)$	70%–80%	1.6895	1.6841	0.32%	7.5	**1.6501**	**2.33%**	1.5	15
$[28, 82]$	80%–90%	1.6832	1.6829	0.01%	7.5	**1.6369**	**2.75%**	1.5	15
$[83, \infty)$	90%–100%	2.1862	2.1662	0.91%	30	**2.1553**	**1.41%**	30	150

Table 7: Focused learning to improve Google's recommender system.

Movie Cluster	Test RMSE for $\lambda_{\text{unfocus}} =$ 3	15	30	60	150	300
$\text{RMSE}_{\mathcal{R}\text{Val.}} \in [1.37, \infty)$	1.1475	1.1152	1.0991	1.0776	1.0696	1.0876
$\text{RMSE}_{\mathcal{R}\text{Val.}} \in [1.13, 1.37)$	1.1103	1.0733	1.0660	1.0521	1.0405	1.0583
$\text{RMSE}_{\mathcal{R}\text{Val.}} \in [0.83, 0.87)$	0.8733	0.8486	0.8450	0.8473	0.8541	0.8658
$\text{RMSE}_{\mathcal{R}\text{Val.}} \in [0, 0.62)$	0.8366	0.7802	0.7688	0.7665	0.7681	0.7812
Degree $\in [1, 11)$	1.1513	1.1051	1.0994	1.0987	1.1246	1.1650
Degree $\in [11, 25)$	1.2633	1.1510	1.1205	1.1146	1.1586	1.1987
Degree $\in [135, 233)$	1.0209	0.9460	0.9275	0.9181	0.9190	0.9297
Degree $\in [2388, \infty)$	0.8571	0.8464	0.8459	0.8473	0.8520	0.8613
Action	0.8188	0.8040	0.7988	0.7959	0.7974	0.8025
Documentaries	0.9624	0.9060	0.8920	0.8806	0.8692	0.8782
IMAX	0.9626	0.9013	0.8958	0.8865	0.8846	0.9077
Western	0.8313	0.8169	0.8261	0.8276	0.8014	0.8122

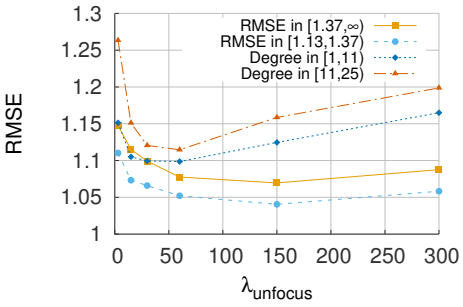


Figure 8: Regularization of one item has an effect on the accuracy of the rest of the model.

Focus Groups	λ_{unfocus}	$\lambda_{\text{focus}-1}$	$\lambda_{\text{focus}-2}$	Test RMSE Focus-1	Focus-2
$[1, 11)$; $[11, 26)$	30	3	3	1.1045	1.1201
	30	3	150	**1.0973**	1.5974
	30	300	3	1.3929	**1.1196**
	30	300	150	1.3922	1.5977
$[45, 78)$; $[233, 444)$	30	3	15	0.9788	0.8934
	30	3	30	**0.9761**	0.9213
	60	3	15	0.9733	**0.8884**
	60	3	30	0.9716	0.9068

Table 8: Multiple focuses give incompatible settings. (bold values are those selected by cross validation).

to not drown out that information. Based on the results in Table 3, the second perspective seems to hold more often.

Second, we observe that regularization is not independent across focused and unfocused items. That is, the regularization for the parameters of item i can have a significant impact on the prediction error for item j. In Figure 8, we see that even when λ_{focus} is selected according to focused learning, changing λ_{unfocus} has a significant impact on the test RMSE for these focused groups. In particular, we observe in each row, regardless of focus selection technique, a generally convex curve as we increase λ_{unfocus}.

Last, we perform an additional experiment to explore if there exists an optimal $\vec{\lambda}^*$ vector that would give the best results for all items. That is, should each movie have its own regularization λ_j? To investigate this idea, instead of having just one focus set, we expand our formulation slightly to have two focus sets, each with its own regularization $\lambda_{\text{focus}-1}$ and $\lambda_{\text{focus}-2}$, along with λ_{unfocus}. We then search to find the optimal hyperparameters for validation RMSE for both focus-1 and focus-2. We find that in some cases, as seen in two examples in Table 8, the optimal hyperparameters for focus-1 and focus-2 are incompatible. For example, when focus-1 is the set of movies with degree $\in [45, 78)$ and focus-2 is the set of movies with degree $\in [233, 444)$, then increasing $\lambda_{\text{focus}-2}$ from 15 to 30 consistently helps focus-1 and hurts focus-2. This suggests that there may not be a single correct setting of $\vec{\lambda}$ that performs optimally for all items. However, because the change in RMSE values from this additional regularization term is marginal, we are hesitant to draw any strong conclusions.

8. CONCLUSION

In this paper we explored focused situations when classic CF systems fail and how we can improve prediction quality in these cases. We made the following contributions:

- **Problem Formulation**, including empirical and theoretical evidence that a single globally-optimal model is not necessarily optimal for subsets of the data.
- **Algorithm** for focused learning with state-of-the-art models to improve recommendations for a pre-specified subset of items.
- **Real-world experiments** demonstrating the success of focused learning.

While these contributions are successful on their own, we believe this research opens exciting new directions for future research on focused learning.

Acknowledgements: The authors would like to thank the many people in Google Research that provided valuable feedback throughout the research process. In particular, we would like to thank Li Zhang, Nic Mayoraz, Sally Goldman, Rasmus Larsen, Brandon Dutra, Madhavan Kidambi, and Sarvjeet Singh.

9. REFERENCES

[1] L. A. Adamic and B. A. Huberman. Power-law distribution of the world wide web. *Science*, 287(5461):2115–2115, 2000.

[2] M. Aharon, O. Anava, N. Avigdor-Elgrabli, D. Drachsler-Cohen, S. Golan, and O. Somekh. ExcUseMe: Asking users to help in item cold-start recommendations. In *RecSys*. ACM, 2015.

[3] O. Anava, S. Golan, N. Golbandi, Z. Karnin, R. Lempel, O. Rokhlenko, and O. Somekh. Budget-constrained item cold-start handling in collaborative filtering recommenders via optimal design. In *WWW*, pages 45–54, 2015.

[4] I. Barjasteh, R. Forsati, F. Masrour, A.-H. Esfahanian, and H. Radha. Cold-start item and user recommendation with decoupled completion and transduction. In *RecSys*, 2015.

[5] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *NIPS*, pages 2546–2554, 2011.

[6] A. Beutel, A. Ahmed, and A. J. Smola. ACCAMS: Additive Co-Clustering to Approximate Matrices Succinctly. In *WWW*, 2015.

[7] A. Beutel, K. Murray, C. Faloutsos, and A. J. Smola. CoBaFi: Collaborative Bayesian Filtering. In *WWW*, pages 97–108, 2014.

[8] L. Breiman and J. H. Friedman. Estimating optimal transformations for multiple regression and correlation. *JASA*, 80(391):580–598, 1985.

[9] E. Brochu, V. Cora, and N. De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv:1012.2599*, 2010.

[10] E. Christakopoulou and G. Karypis. Local item-item models for top-n recommendation. In *RecSys*, 2016.

[11] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *KDD*, pages 193–202. ACM, 2014.

[12] M. Dredze, K. Crammer, and F. Pereira. Confidence weighted linear classification. In *ICML*. ACM, 2008.

[13] A. M. Elkahky, Y. Song, and X. He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *WWW*. ACM, 2015.

[14] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262, 1999.

[15] P. Hamel. To have a tiger by the tail: Improving music recommendation for international users. In *ICML workshop*, 2015.

[16] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu. Personalized recommendation via cross-domain triadic factorization. In *WWW*. ACM, 2013.

[17] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, 2008.

[18] Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *KDD*, pages 426–434, New York, NY, USA, 2008.

[19] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[20] X. N. Lam, T. Vu, T. D. Le, and A. D. Duong. Addressing cold-start problem in recommendation systems. In *IMCOM*, pages 208–211. ACM, 2008.

[21] J. Lee, S. Kim, G. Lebanon, and Y. Singer. Local low-rank matrix approximation. In *ICML*, 2013.

[22] J. Lee, M. Sun, and G. Lebanon. PREA: Personalized recommendation algorithms toolkit. *JMLR*, 13(Sep), 2012.

[23] H. Ma, H. Yang, M. R. Lyu, and I. King. SoRec: social recommendation using probabilistic matrix factorization. In *CIKM*, pages 931–940. ACM, 2008.

[24] L. Mason, J. Baxter, P. L. Bartlett, and M. R. Frean. Boosting algorithms as gradient descent. *NIPS*, 2000.

[25] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*, pages 165–172. ACM, 2013.

[26] M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet mathematics*, 1(2):226–251, 2004.

[27] Y. J. Park and A. Tuzhilin. The long tail of recommender systems and how to leverage it. In *RecSys*, pages 11–18. ACM, 2008.

[28] S. Rendle. Factorization machines with libFM. *ACM TIST*, 3(3):57:1–57:22, May 2012.

[29] J. Riedl and J. Konstan. Movielens dataset, 1998.

[30] Y. Rong, X. Wen, and H. Cheng. A monte carlo algorithm for cold start recommendation. In *WWW*, pages 327–336. ACM, 2014.

[31] S. Sahebi and P. Brusilovsky. It takes two to tango: An exploration of domain pairs for cross-domain collaborative filtering. In *RecSys*. ACM, 2015.

[32] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, pages 880–887. ACM, 2008.

[33] M. Saveski and A. Mantrach. Item cold-start recommendations: learning local collective embeddings. In *RecSys*. ACM, 2014.

[34] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR*. ACM, 2002.

[35] L. Shi. Trading-off among accuracy, similarity, diversity, and long-tail: a graph-based recommendation approach. In *RecSys*. ACM, 2013.

[36] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *NIPS*, pages 2951–2959, 2012.

[37] C. Tan, E. H. Chi, D. Huffaker, G. Kossinets, and A. J. Smola. Instant foodie: Predicting expert ratings from grassroots. In *CIKM*, 2013.

[38] L. Vilnis and A. McCallum. Word representations via gaussian embedding. *arXiv:1412.6623*, 2014.

[39] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen. Challenging the long tail recommendation. *Proc. VLDB Endow.*, 5(9):896–907, May 2012.

[40] X. Yu, H. Ma, B.-J. P. Hsu, and J. Han. On building entity recommender systems using user click log and freebase knowledge. In *WSDM*. ACM, 2014.

[41] Z. Zhao, Z. Cheng, L. Hong, and E. H. Chi. Improving user topic interest profiles by behavior factorization. In *WWW*, pages 1406–1416, 2015.