Xpl-CF: Explainable Embeddings for Feature-based Collaborative Filtering

Faisal M. Almutairi
University of Minnesota
almut012@umn.edu

Nicholas D. Sidiropoulos
University of Virginia
nikos@virginia.edu

Bo Yang
Amazon Alexa
yang4173@umn.edu

ABSTRACT

Collaborative filtering (CF) methods are making an impact on our daily lives in a wide range of applications, including recommender systems and personalization. Latent factor methods, e.g., matrix factorization (MF), have been the state-of-the-art in CF, however they lack interpretability and do not provide a straightforward explanation for their predictions. Explainability is gaining momentum in recommender systems for accountability, and because a good explanation can swing an undecided user. Most recent explainable recommendation methods require auxiliary data such as review text or item content on top of item ratings. In this paper, we address the case where no additional data are available and propose augmenting the classical MF framework for CF with a prior that encodes each user’s embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. Our Xpl-CF approach automatically reveals these user-item relationships, which underpin the latent factors and explain how the resulting recommendations are formed. We showcase the effectiveness of Xpl-CF on real data from various application domains. We also evaluate the explainability of the user-item relationship obtained from Xpl-CF through numeric evaluation and case study examples.

CCS CONCEPTS

• **Information systems → Collaborative filtering**.

KEYWORDS

collaborative filtering, matrix factorization, explainable recommendation

ACM Reference Format:
Faisal M. Almutairi, Nicholas D. Sidiropoulos, and Bo Yang. 2021. Xpl-CF: Explainable Embeddings for Feature-based Collaborative Filtering. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM ’21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459637.3482221

1 INTRODUCTION

In the context of recommendation engines, collaborative filtering (CF) is the process of filtering information using techniques involving collaboration among multiple viewpoints. CF models can be divided into *neighbor-based* and *feature-based* (e.g., latent factor) categories; latent factor methods have been the state-of-the art in CF. One of the very successful latent factor CF techniques is matrix factorization (MF) due to its ability to capture correlations and higher-order statistical dependencies across dimensions. MF automatically predicts a person’s affinity for items by connecting that person’s historical interests with the interests of similar users, while taking inter-dependencies among items into account. More specifically, given a sparse user × item rating matrix, MF uses the observed ratings to learn dense latent representations (embeddings) of users and items in a lower dimensional space. In the inference phase, the *unknown* entry corresponding to the i^{th} user and j^{th} item is predicted by the dot product of their embeddings. The more similar the user’s and item’s embeddings (closer to each other in the latent space), the larger their dot product (predicted rating). Although this provides a geometric interpretation of the prediction of MF, we still cannot explain how the latent vectors are formed. MF methods tend to be black-box machine learning models that lack interpretability and do not provide a straightforward explanation for their predictions; this is the main drawback of latent factor methods compared to neighbor-based CF.

Researchers have recently found that interpretations and explainability in recommendation systems play a significant role to improve the transparency, persuasiveness, effectiveness, trustworthiness, and user satisfaction [18]. They also enable system designers to diagnose, debug, and refine the recommendation algorithm. Interpretable recommendations are of interest in many applications, especially in business-to-business (B2B) scenarios where the recipient of the recommendation is a salesperson responsible for the client. A salesperson has to decide whether to pursue a sales opportunity (i.e., recommendation), and (s)he relies on evaluating the *reasoning* behind a generated recommendation [4]. Explainable recommendations have also been proven effective in business-to-client (B2C) e-commerce settings [19].

In this work, we propose Xpl-CF, a CF approach that augments the classical MF model with a new type of prior information. The proposed prior not only improves the prediction accuracy of MF, but it also underpins the latent factors and explains how the resulting recommendations are formed. Unlike most recent explainable recommendation methods, Xpl-CF does not require additional data. The main intuition behind our modeling is that a user preference profile (latent factor) is determined by their experience with a subset of items. The strength of this association can differ, e.g., a user might strongly associate herself with Sci-Fi movies and mildly with horror movies. Our proposed prior encodes a user’s embedding as a sparse linear combination of item embeddings. Conversely, an item’s embedding is determined by a subset of users (i.e., a sparse

CIKM ’21, November 1–5, 2021, Virtual Event, QLD, Australia
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482221

linear combination of user embeddings). We demonstrate the effectiveness of the proposed model on real datasets from investment and recommender system domains.

2 RELATED WORK

Explainable recommendation methods can be grouped into two broad types: post-hoc and embedded methods. In post-hoc approaches, explanations and recommendations are generated from separate models [9, 10, 12]. Embedded methods, on the other hand, aim to explain the recommendation model itself [7, 19]. Here we focus on the embedded category; we refer the reader to [18] for an in-depth literature review. In the case of neighbor-based CF methods, the recommendations are directly based on similarities between users and/or items [13], which also serve to explain the recommendation in a rather straightforward way - but these methods are far from the state-of-art in terms of quality of recommendation. The explanation task is trickier with latent factor models. Their internal decision processes cannot be directly interpreted by humans, since by finding lower dimensional representations of users and items they abstract away from the interactions between users and items [12]. The two predominant approaches in the recommendation literature are: **i)** adding constraints to the latent factor models (our approach belongs to this class), and **ii)** using external data.

In the latter category, **external data** such as product reviews (e.g., in TF/IDF form) and rating data are jointly factored with shared latent factors via topic modelling [2] or coupled MF [19]. For instance, in the case of topic modeling, the learned latent topics can be leveraged to provide an interpretation of the latent factors. Although exploiting additional information provides valuable insight, such information is not always available - especially in B2B settings. Moreover, if the additional sources used for explanation are not correlated with the rating data, then the explanations will not accurately reflect the reasons for the recommendation and will degrade the rating prediction accuracy.

Closest to our work is the Explainable MF (EMF) approach in [1]. EMF is a **constrained latent factor** model that modifies the cost function of MF by penalizing the Euclidean distance of the latent vectors of similar users and items. The similarity is predefined by a user × item similarity matrix and is measured by the ratio of the neighbors of user i who have rated item j - the neighborhood is calculated using cosine similarity. EMF is essentially a hybrid method between neighbor- and feature-based CF. Although EMF has the advantage of not requiring extra data views to generate explanations, it still employs a rather restrictive predefined neighborhood model. We point out that EMF explains the recommendation via the distance in the latent space and does not attempt to explain the embeddings of users/items. Xpl-CF, on the other hand, explains the embedding of a user in relation to item embeddings and vice versa. In contrast to [1], the explainability relationships in Xpl-CF are automatically revealed by the model and not predefined apriori.

3 PROPOSED METHOD

3.1 Formulation

Assume we have a data matrix $\mathbf{X} \in R^{N\times M}$, with the user × item rating data. The matrix factorization CF models assume that $\mathbf{X}$ can be approximated using low-rank factor matrices, i.e., $\mathbf{X} \approx \mathbf{A}\mathbf{B}^T$, where rows of $\mathbf{A} \in \mathbb{R}^{N\times R}$ and $\mathbf{B} \in \mathbb{R}^{M\times R}$ are the embeddings of users and items, respectively, and $R \leq \min(N, M)$ is the matrix rank. After obtaining $\mathbf{A}$ and $\mathbf{B}$, the unknown rating of the i^{th} user for the j^{th} item is predicted by the dot product of their embeddings, i.e., $\mathbf{X}(i, j) = \mathbf{a}_i\mathbf{b}_j^T$. In other words, the MF model produces latent representations of users and items in a lower dimensional space. If a user likes an item, the distance between their embeddings will be small and therefore their dot product is larger.

In the original data domain, the user-item relationships are clear: users are represented by their ratings of a subset of items, and items are represented by ratings given by a subset of users. However, the user-item relationships are not clear in the latent space. Why is the embedding of a user (or item) more/less similar to certain items (users)? Our framework addresses this question. In our proposed formulation, we rely on MF to obtain user and item embeddings and impose a prior on these embeddings. The prior encodes each user's embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. This leads to the following problem formulation.

$$\begin{aligned} \min_{\mathbf{A},\mathbf{B},\mathbf{S},\mathbf{Z}} \quad & \|\mathbf{\Omega} \odot (\mathbf{X} - \mathbf{A}\mathbf{B}^T)\|_F^2 + \mu_a\|\mathbf{A} - \mathbf{S}\mathbf{B}\|_F^2 \\ & + \mu_b\|\mathbf{B} - \mathbf{Z}\mathbf{A}\|_F^2 + \lambda \mathbf{1}^T(\mathbf{S} + \mathbf{Z}^T)\mathbf{1} \qquad (1) \\ \text{s.t.} \quad & \mathbf{S}, \mathbf{Z}^T \geq \mathbf{0} \end{aligned}$$

where $\odot$ is the element-wise product, and $\mathbf{\Omega}$ is a zero-one matrix indicating the availability of the corresponding entries in $\mathbf{X}$. $\mathbf{1}$ is a vector of all ones of the appropriate size and $\mu_a \geq 0$, $\mu_b \geq 0$, and $\lambda \geq 0$ are regularization hyper-parameters. The first term in (1) is the least squares data fitting, while the second and third terms represent the user-item relationships in the latent space. The last term is introduced to promote sparsity in $\mathbf{S}$ and $\mathbf{Z}$ (l_1 norm with non-negativity boils down to the sum of entries).

The variables $\mathbf{S}$ and $\mathbf{Z}$ reveal the user-item relationships in the latent space and explain the resulting recommendations. For easier interpretability, we model $\mathbf{S}$ and $\mathbf{Z}$ as element-wise non-negative. Assume that $\mathbf{a}_i$ and $\mathbf{s}_i$ are rows in $\mathbf{A}$ and $\mathbf{S}$, respectively. Then, $\mathbf{a}_i = \mathbf{s}_i\mathbf{B}$ and $\mathbf{s}_i$ is a *sparse* vector that selects (and scales) some item embeddings to form user i's embedding. The motivation behind this assumption is that the features that the user cares about are characterized by her experience and knowledge about a subset of items. Similarly, $\mathbf{b}_j = \mathbf{z}_j\mathbf{A}$ assumes that the item embedding is characterized by a subset of user embeddings.

3.2 Explainability Analysis

In this subsection, we present how Xpl-CF can be used to explain a recommendation. The prediction of a value in $\mathbf{X}$, $\hat{x}_{ij}$, is

$$\hat{x}_{ij} = \mathbf{a}_i\mathbf{b}_j^T \approx \mathbf{s}_i\mathbf{B}\mathbf{b}_j^T \qquad (2a)$$

$$\approx \mathbf{s}_i\mathbf{Z}\mathbf{A}\mathbf{A}^T\mathbf{z}_j^T = \mathbf{u}_i\mathbf{A}\mathbf{A}^T\mathbf{z}_j^T \qquad (2b)$$

where in (2a), the recommendation boils down to the similarity between the target item $\mathbf{b}_j$ and a subset of items selected by $\mathbf{s}_i$. Because $\mathbf{s}_i$ is fixed across all items for user i, we can interpret this subset of items as the "lens" that user i sees all items through. Equation (2b) provides another intriguing insight by explaining the prediction as a (sparse) linear combination of user × user similarity encoded in $\mathbf{A}\mathbf{A}^T$ - note that vector $\mathbf{u}_i := \mathbf{s}_i\mathbf{Z}$ may be dense. In the same vein, we can write

$$\hat{x}_{ij} = \mathbf{b}_j\mathbf{a}_i^T \approx \mathbf{z}_j\mathbf{A}\mathbf{a}_i^T \tag{3a}$$

$$\approx \mathbf{z}_j\mathbf{S}\mathbf{B}(\mathbf{s}_i\mathbf{B})^T = \mathbf{v}_j\mathbf{B}\mathbf{B}^T\mathbf{s}_i^T \tag{3b}$$

where $\mathbf{v}_j := \mathbf{z}_j\mathbf{S}$. The prediction in (3a) is explained as the similarity between the target user with a subset of users selected by the model, whereas (3b) explains the prediction as a (sparse) linear combination of item × item similarity. Combining (2a) and (3a), we can say

$$\hat{x}_{ij} = \mathbf{a}_i\mathbf{b}_j^T \approx \mathbf{s}_i\mathbf{B}(\mathbf{z}_j\mathbf{A})^T = \mathbf{s}_i\mathbf{B}\mathbf{A}^T\mathbf{z}_j^T \tag{4}$$

where each prediction is explained through a sparse linear combination of user-item similarity encoded in $\mathbf{B}\mathbf{A}^T$. Thus, the explanation associated with a recommendation can list the items (and users if applicable) that contribute to the prediction the most (i.e., items with highest values in $\mathbf{s}_i$). Another benefit of $\mathbf{S}$ and $\mathbf{Z}$ is that they can be used to extract communities *in the latent space*. For instance, a community includes users whose embeddings are characterized by the same items; however, this is out of the scope of this paper and we leave it for future work.

3.3 Model Engineering

We add two modifications to the problem formulation in (1). The first point we address is the scaling between the low-rank factors $\mathbf{A}$ and $\mathbf{B}$. Since the embedding of a user is a linear combination of item embeddings and vice versa, it is important for $\mathbf{A}$ and $\mathbf{B}$ to be within the same scale. Thus, we constrain the columns of $\mathbf{A}$ and $\mathbf{B}$ to be on the unit l_2 norm ball. We introduce a diagonal matrix to allow us to fix the scale without loss of generality of the factorization model [16], i.e., $\mathbf{X} \approx \mathbf{A}\mathbf{D}\mathbf{B}^T$, where $\mathbf{D}$ is a diagonal matrix.

The second addition to the model is the user and item bias terms. These biases capture how well an item is rated compared to the average, across all items. Similarly, a user's bias corresponds to the user's tendency to give better/worse ratings relative to the average. Taking these points into account, we obtain the following:

$$\begin{aligned} \min_{\mathbf{A},\mathbf{B},\mathbf{d},\mathbf{a},\mathbf{b},\mathbf{S},\mathbf{Z}} \quad & \|\mathbf{\Omega} \odot (\mathbf{X} - \mathbf{A}\mathbf{D}\mathbf{B}^T - \mathbf{a}\mathbf{1}^T - \mathbf{1}\mathbf{b}^T)\|_F^2 + \lambda\mathbf{1}^T(\mathbf{S} + \mathbf{Z}^T)\mathbf{1} \\ & + \mu_a\|\mathbf{A} - \mathbf{S}\mathbf{B}\|_F^2 + \mu_b\|\mathbf{B} - \mathbf{Z}\mathbf{A}\|_F^2 + \eta(\|\mathbf{d}\|^2 + \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2) \\ \text{s.t.} \quad & \mathbf{S}, \mathbf{Z}^T \geq \mathbf{0},\ \mathbf{D} = \text{Diag}(\mathbf{d}) \\ & \|\mathbf{A}(:, r)\|_2 = \|\mathbf{B}(:, r)\|_2 = 1,\ \forall r \in [R] \end{aligned} \tag{5}$$

3.4 Optimization

The formulation in (5) is non-convex and a very challenging optimization problem. An additional challenge stems from the fact that $\mathbf{X}$ is partially observed. We employ a carefully designed alternating optimization (AO) algorithm. The proposed algorithm leverages the Alternating Direction Method of Multipliers (ADMM) and utilizes parallel computing, computation caching, and warm-start to provide a scalable and efficient implementation. The high level algorithmic strategy is to employ AO to update $\mathbf{A}$, $\mathbf{B}$, $\mathbf{d}$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{S}$ and $\mathbf{Z}$ one at a time, while fixing the others. Let us consider the subproblem w.r.t. $\mathbf{A}$. We introduce an auxiliary variable $\widetilde{\mathbf{A}}$ to handle the unit $l2$ norm ball constraint. The ADMM updates for $\mathbf{A}$ are:

$$\begin{aligned} \widetilde{\mathbf{A}}(:, i) \leftarrow \min_{\widetilde{\mathbf{A}}(:,i)} \quad & \frac{1}{2}\|\mathbf{X}_s(i, \mathcal{J}_i)^T - \widetilde{\mathbf{B}}(\mathcal{J}_i, :)\widetilde{\mathbf{A}}(:, i)\|_F^2 + \frac{\mu_a}{2}\|\widetilde{\mathbf{A}}(:, i)^T - \\ & \mathbf{S}(i, :)\mathbf{B}\|_F^2 + \frac{\rho_a}{2}\|\mathbf{A}(i, :) - \widetilde{\mathbf{A}}(:, i)^T + \mathbf{U}(i, :)\|_F^2,\ \forall i \in [N] \end{aligned} \tag{6a}$$

$$\mathbf{A}(:, r) \leftarrow \mathbf{A}_u(:, r)/\|\mathbf{A}_u(:, r)\|_2,\ \forall r \in [R] \tag{6b}$$

$$\mathbf{U} \leftarrow \mathbf{U} + \mathbf{A} - \widetilde{\mathbf{A}}^T \tag{6c}$$

where $\mathbf{X}_s = \mathbf{X} - \mathbf{a}\mathbf{1}^T - \mathbf{1}\mathbf{b}^T$, $\widetilde{\mathbf{B}} = \mathbf{B}\mathbf{D}$, $\mathbf{A}_u = \widetilde{\mathbf{A}}^T - \mathbf{U}$ and $\mathcal{J}_i$ is the set of items that have observations for user i. Equation (6b) is a simple column scaling, whereas (6c) is the dual variable update. Problem (6a) is a *weighted* least squares problem (weighted by the binary matrix $\mathbf{\Omega}$). An important implication is that (6a) corresponds to solving N separable least squares problems, which enables parallel computation. One point that requires more care is handling the missing entries in $\mathbf{X}$ (or the zeros in $\mathbf{\Omega}$). The way we handle this is by removing the equations that correspond to the indices of the missing entries, i.e., we remove rows in $\widetilde{\mathbf{B}}$ and entries in $\mathbf{X}_s(i, :)$ when solving for each $\widetilde{\mathbf{A}}(:, i)$. Moreover, for each least squares problem, we do not compute the matrix inversion explicitly. Instead, the Cholesky decomposition of a Gram matrix is computed. Then, back and forward substitution steps are performed to obtain $\widetilde{\mathbf{A}}(:, i)$. Matrix $\mathbf{B}$ is updated using the ADMM in the same fashion as $\mathbf{A}$ with the appropriate transpose.

Next, we update vector $\mathbf{d}$ by by minimizing $(\|\mathbf{x}(\mathcal{T}) - \mathbf{K}(\mathcal{T}, :)\mathbf{d}\|_F^2 + \eta\|\mathbf{d}\|_2^2)$ w.r.t $\mathbf{d}$, where $\mathbf{K} = \mathbf{B} \otimes \mathbf{A}$, $\otimes$ is the Khatri–Rao product, $\mathbf{x} = \text{vec}(\mathbf{X}_s)$ and $\mathcal{T}$ is the set of observed entries in $\mathbf{x}$.

Next, we update the bias variables for users and items. The update for the bias of user i, $\mathbf{a}(i)$, corresponds to solving:

$$\min_{\mathbf{a}(i)} \quad \frac{1}{2}\|\mathbf{X}_b(i, \mathcal{J}_i)^T - \mathbf{1}\mathbf{a}(i)\|_F^2 + \frac{\eta}{2}(\mathbf{a}(i))^2 \tag{7}$$

where $\mathbf{X}_b = \mathbf{X} - \mathbf{A}\mathbf{D}\mathbf{B}^T - \mathbf{1}\mathbf{b}^T$. The items' biases in $\mathbf{b}$ are updated similarly. Note that the updates of the bias variables across users (and items) are independent; thus, they can be computed in parallel.

Finally, we update the latent mapping variables $\mathbf{S}$ and $\mathbf{Z}$ using the ADMM (we present the update of $\mathbf{S}$ as a running example). We omit the terms in (5) that do not include $\mathbf{S}$ and introduce an auxiliary variable $\widetilde{\mathbf{S}}$ to split the effort of handling the least squares terms and the non-negativity constraint. The ADMM updates for the resulting problem are the following:

$$\widetilde{\mathbf{S}} \leftarrow \min_{\widetilde{\mathbf{S}}} \quad \frac{\mu_a}{2}\|\mathbf{A}^T - \mathbf{B}^T\widetilde{\mathbf{S}}\|_F^2 + \lambda\mathbf{1}^T\widetilde{\mathbf{S}}\mathbf{1} + \frac{\rho_s}{2}\|\mathbf{S} - \widetilde{\mathbf{S}}^T + \mathbf{V}\|_F^2 \tag{8a}$$

$$\mathbf{S} \leftarrow \arg\min_{\mathbf{S} \geq \mathbf{0}} \quad \|\mathbf{S} - \widetilde{\mathbf{S}}^T + \mathbf{V}\|_F^2 \tag{8b}$$

$$\mathbf{V} \leftarrow \mathbf{V} + \mathbf{S} - \widetilde{\mathbf{S}}^T \tag{8c}$$

Equation (8b) is a simple element-wise non-negative projection (i.e., zero out the negative elements in $\widetilde{\mathbf{S}}^T - \mathbf{V}$). Equation (8c) is the dual variable update. Similar to the case of $\mathbf{A}$, the update in (8a) corresponds to solving N separable least squares problems that can be solved in parallel. Unlike (6a), the N problems in (8a) share the same mixing matrix $\mathbf{B}^T$. This means that we need to compute the Cholesky decomposition of $(\mu_a\mathbf{B}\mathbf{B}^T + \rho_s\mathbf{I})$ only once[1].

[1] Code is available at https://github.com/FaisalAlmutairi/explainable_recommendation.

Table 1: Matrix completion error of all methods.

		R = 10		R = 50		R = 100	
Data	Model	RMSE	MAE	RMSE	MAE	RMSE	MAE
B2B	BMF	**1.2026**	**0.8547**	1.2100	0.8565	1.2334	0.8704
	AdaErr	1.4536	1.0230	1.4391	1.0142	1.4593	1.0496
	EMF	1.2373	0.8843	1.2360	0.8821	1.2231	0.8598
	Xpl-CF	1.2482	0.8779	**1.1915**	**0.8392**	**1.1892**	**0.8339**
ML100K	BMF	0.9228	0.7261	0.9188	0.7245	0.9158	0.7228
	AdaErr	0.9432	0.7514	0.9326	0.7422	1.1119	0.9244
	EMF	0.9393	0.7504	0.9355	0.7491	0.9339	0.7479
	Xpl-CF	**0.9123**	**0.7150**	**0.9132**	**0.7182**	**0.9156**	**0.7166**

Table 2: Explanability evaluation using ML100K (R = 10).

	k = 10		k = 15		k = 20	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
BMF	0.9228	0.7261	0.9228	0.7261	0.9228	0.7261
BMF-RandU	0.9342	0.7348	0.9433	0.7412	0.9521	0.7476
BMF-S	0.9452	0.7432	0.9598	0.7544	0.9756	0.7660
BMF-RandI	0.9414	0.7383	0.9550	0.7467	0.9705	0.7565
BMF-Z	0.9569	0.7498	0.9822	0.7659	1.0064	0.7815

Table 3: List of movies that explain the prediction of "Get Shorty".

User 1 ($U1$)	User 2 ($U2$)
Alphaville	Showgirls
Showgirls	Ready to Wear (Pret-A-Porter)
Striking Distance	Vampire in Brooklyn
Dead Presidents	Miami Rhapsody
Bloodsport 2	Party Girl
Fair Game	The Fog
High School High	Four Days in September
Steel	Little Big League
The Jackal	Free Willy 3: The Rescue
April Fool's Day	Exit to Eden

4 EXPERIMENTAL RESULTS

Datasets: We evaluate Xpl-CF using the following datasets. **i) B2B** [17], an investor holding-position dataset (an example of B2B applications). The data are organized into a company vs investor matrix where the entries are the percentage of shares that one investor holds in each company among all the shares issued. We use the data as collected and preprocessed in [17]. **ii) ML100K [3]**, a movie rating dataset and a popular baseline in recommender systems literature. It contains $\sim 10^5$ ratings. The original data only include users with at least 20 ratings. We also filter out movies with less than 20 ratings.

Baselines: We evaluate Xpl-CF against the following baselines. **i) BMF**, a matrix factorization approach with rank-1 factors specified to capture items' and users' biases [6, 11] implemented using Stochastic Gradient Descent (SGD). Our approach in (5) boils down to BMF when $\mu_a = \mu_b = 0$. **ii) AdaErr**, a CF model based on MF with a learning rate that adaptively adjusts based on the prediction error [8]. **iii) EMF**, an explainable CF model based on MF [1]; see Sec. 2 for more details.

Matrix Completion: In order to evaluate the quality of the embeddings, we take a generic approach by evaluating the embedding quality on the matrix completion task. The philosophy is: *if the embeddings predict missing data with high accuracy, then they must be good representations of items and users.* Accurate prediction, e.g., predicting holding-positions in the B2B dataset, not only gives a relative ranking of the likelihood of interest, but it also enables deriving useful information (e.g., percentage of investment). We split each dataset into 5 equal folds. After training the models on 4 folds, we test the trained models on the held-out fold. The hyper-parameters of all methods are chosen via cross validation (10% of training data). Due to random initialization, the results can differ for different runs; thus, after choosing the hyper-parameters, we run the training and testing on each fold 20 times and report the average error of the total 100 experiments. Table 1 shows the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) with various ranks R (number of features). Explainable methods usually suffer from accuracy-interpretability trade-off, which can be seen by comparing the explainable method EMF and BMF. Nevertheless, Xpl-CF significantly improves all the baselines, especially when $R = 10$ with ML100K and when $R = \{50, 100\}$ with B2B. The fact that Xpl-CF improves BMF suggests that the data follow the proposed prior.

Explainability Evaluation: There is no well-defined methodology for evaluating the model's explainability. There are two main approaches in the literature: online and offline. Online evaluation tests the performance by adding explanation to the recommendation loop on a live recommendation platform, e.g., e-commerce website [15, 19]. Offline evaluation usually either quantifies the importance of the explanation provided by the model [7], or demonstrates the quality of the explainability by examples [5, 14] - we adopt both strategies. Following the approach in [7], we remove the k ratings in the training data with the highest values in $\mathbf{s}_i$ (for each user). Then, we train a BMF model using the resulting training set - we call this model BMF-S. We perform the same strategy and remove the k ratings with the highest values in $\mathbf{z}_j$ for each item (we call it BMF-Z). Table 2 shows that the performance degradation of BMF-S and BMF-Z (relative to BMF) is significantly higher compared to when we *randomly* remove k training ratings from each user (BMF-RandU) or from each item (BMF-RandI). This suggests that the items (users) identified by $\mathbf{S}$ ($\mathbf{Z}$) are important in defining a user (item).

We chose two users from ML100K data: $U1$ who has a clear interest in action, adventure and thriller movies and $U2$ who is more interested in comedy and romance - we determine their interest based on movies they have rated. Table 3 shows the list of movies that explain the rating prediction for "Get Shorty" for these two users. To generate these explanations, we selected the top 20 movies with the highest values in $\mathbf{s}_i$ for $U1$ and $U2$ (we denote these sets as $\mathcal{S}_{U1}$ and $\mathcal{S}_{U2}$, respectively). Then, in Table 3, we list the top 10 movies with the highest values in $\mathbf{B}(\mathcal{S}_{U1}, :)\mathbf{b}_j^T$ for $U1$ (and similarly for $U2$). These explanations are based on (2a); note that in this case $\mathbf{s}_i$ is user-specific, while $\mathbf{b}_j$ is item-specific. Get Shorty is an action *and* comedy movie. We highlight action/adventure/thriller movies in red, while comedy movies are in blue. One can see that the prediction is explained from an action viewpoint for $U1$, while it is explained by comedy movies for $U2$. Note that our model uses the rating data only and does not have access to the movies' genres.

5 CONCLUSION

In this paper, we proposed Xpl-CF, a CF model that augments the classical MF framework for CF with a prior that encodes each user's embedding as a sparse linear combination of item embeddings, and vice versa for each item embedding. Xpl-CF not only improves the prediction accuracy of MF, but it also automatically reveals the user-item relationships in the latent space (without requiring additional data). These relationships underpin the latent factors and explain how the resulting recommendations are formed.

Acknowledgments: N. Sidiropoulos was partially supported by NSF IIS-1908070.

REFERENCES

[1] Behnoush Abdollahi and Olfa Nasraoui. 2016. Explainable matrix factorization for collaborative filtering. In *Proceedings of the 25th International Conference Companion on World Wide Web*. 5–6.

[2] Yang Bao, Hui Fang, and Jie Zhang. 2014. Topicmf: Simultaneously exploiting ratings and reviews for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 28.

[3] 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.

[4] Reinhard Heckel, Michail Vlachos, Thomas Parnell, and Celestine Dünner. 2017. Scalable and interpretable product recommendations via overlapping co-clustering. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 1033–1044.

[5] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 263–272.

[6] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 426–434.

[7] Carolin Lawrence, Timo Sztyler, and Mathias Niepert. 2020. Explaining Neural Matrix Factorization with Gradient Rollback. *arXiv preprint arXiv:2010.05516* (2020).

[8] Dongsheng Li, Chao Chen, Qin Lv, Hansu Gu, Tun Lu, Li Shang, Ning Gu, and Stephen M Chu. 2018. Adaerror: An adaptive learning rate method for matrix approximation-based collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference*. 741–751.

[9] Huafeng Liu, Jingxuan Wen, Liping Jing, Jian Yu, Xiangliang Zhang, and Min Zhang. 2019. In2Rec: Influence-based interpretable recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1803–1812.

[10] Namyong Park, Andrey Kan, Christos Faloutsos, and Xin Luna Dong. 2020. J-Recs: Principled and Scalable Recommendation Justification. *arXiv preprint arXiv:2011.05928* (2020).

[11] Arkadiusz Paterek. 2007. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, Vol. 2007. 5–8.

[12] Georgina Peake and Jun Wang. 2018. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2060–2069.

[13] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[14] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the eleventh ACM conference on recommender systems*. 297–305.

[15] Yiyi Tao, Yiling Jia, Nan Wang, and Hongning Wang. 2019. The facT: Taming latent factor models for explainability with factorization trees. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 295–304.

[16] Bo Yang, Xiao Fu, and Nicholas D Sidiropoulos. 2016. Learning from hidden traits: Joint factor analysis and latent clustering. *IEEE Transactions on Signal Processing* 65, 1 (2016), 256–269.

[17] Bo Yang, Kejun Huang, and Nicholas D Sidiropoulos. 2020. Identifying Potential Investors with Data Driven Approaches. In *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 235–243.

[18] Yongfeng Zhang and Xu Chen. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192* (2018).

[19] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 83–92.