PReFacTO: Preference Relations Based Factor Model with Topic Awareness and Offset

Priyanka Choudhary*
Indian Institute of Technology Hyderabad
Hyderabad, Telangana, India
cs16mtech11011@iith.ac.in

Maunendra Sankar Desarkar
Indian Institute of Technology Hyderabad
Hyderabad, Telangana, India
maunendra@iith.ac.in

ABSTRACT

Recommendation systems create personalized list of items that might interest the user by analyzing the user's history of past purchases and/or consumption. For rating based systems, most of the traditional methods for recommendation focus on the absolute ratings provided by the users to the items. In this paper, we extend the traditional Matrix Factorization approach for recommendation and propose pairwise relation based factor modeling. While modeling the items in the system, the use of pairwise preferences allow information flow between the items through the preference relations as an additional information. Item feedbacks are available in the form of reviews apart from the rating information. The reviews have textual information that can be really helpful to represent the item's latent feature vector appropriately. We perform topic modeling of the item reviews and use the topic vectors to guide the joint factor modeling of the users and items and learn their final representations. The proposed method shows promising results in comparison to the state-of-the-art methods in our experiments.

KEYWORDS

Recommendation System, Pairwise Preferences, Topic Modeling, Latent Factor Models

ACM Reference Format:
Priyanka Choudhary and Maunendra Sankar Desarkar. 2018. PReFacTO: Preference Relations Based Factor Model with Topic Awareness and Offset. In *Proceedings of ACM SIGIR Workshop on eCommerce (SIGIR 2018 eCom).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Users have access to large variety of items available online for purchase, subscription, consumption etc. Such a huge list of options often result in choice overload, where it becomes difficult to browse through and/or select the items of interest. Recommendation Systems (RS) make this task of selecting appropriate items easier by finding and suggesting subset of the items that might be of interest to the user. Many traditional recommendation techniques use only ratings to assess the users' taste and behavior. Given a small subset of rating data containing ratings given to the items by the users, Recommendation Systems try to predict the ratings of the items that are not yet rated/viewed by the user. Based on these predicted rating values, ranked list of the items that can be of user's interest are recommended to the users. Latent factor models [1, 8, 13] have been extensively used in the past for this purpose.

There are lot of recommendation systems where the user feedback come in the form of ratings. Majority of such recommendation systems use these absolute ratings entered by the users for modeling the users and items according to latent factor modeling, and use those models for recommendation. Latent factor models like Matrix Factorization [8] are commonly used to transform or represent the users and the items to latent feature spaces. These representations are helpful for explaining the observed ratings and predicting the unknown ratings. These latent factors, e.g. in case of movie recommendations, can be genres, actors or directors or something un-interpretable. These factors try to explain the aspects behind the liking of the items by a particular user. The items are modeled in a similar fashion by representation of the hidden factors possessed by them. This representation predicts the rating by possession of these factors in an item and affinity of users towards these hidden factors.

User feedback in the form of reviews along with the ratings is also available for many online systems like Amazon, IMDb, TripAdvisor etc. The review information can be really useful as it contains the users' perception about the items. There can be systems where the item description is also available. There are algorithms [14] which consider the item description as additional input for latent factor modeling. However, the descriptions are often entered by the item producers or sellers. On the other hand, the feedback in the form of reviews given by the user generally conveys these factors that are being liked or disliked in an item. An attempt to include these textual information can be helpful for better modeling, interpretation and visualization of the hidden dimensions [11].

An alternate form of recommendation system can be based on pairwise preferences of the user among the items [3, 4, 7]. Given a pair of items (i, j), user u may give feedback regarding which of the item he prefers over the other. Such type of feedback is referred to as *pairwise preference* or *pairwise preference based feedback*. A survey in [6] shows that users do prefer comparisons through pairwise scores rather than providing absolute ratings. Although there is no available dataset where the pairwise preferences were directly captured, many approaches in literature have induced pairwise preferences from absolute ratings [3, 7, 10] and used those relations for developing algorithms for recommendation.

The existing methods from the literature that are based on pairwise preferences do not consider the item content information in

*This is the corresponding author.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGIR 2018 eCom, July 2018, Ann Arbor, Michigan, USA
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

the modeling process. In this paper, we propose approaches that combine the pairwise feedback with the additional review data available. We propose an algorithm to use Latent Factor modeling using the pairwise preferences to discover the latent dimensions, map users and items to joint latent feature vector space and produce recommendations for the end user. The latent feature vector space for the items are derived through topic modeling. In this approach, we construct a proxy document for each item by considering the reviews that it has got. If available, the descriptions of the items also can be used to populate this document. We performed probabilistic topic modeling on these documents representing items using Latent Dirichlet Allocation (LDA). These topics are then used to guide the factorization process for learning the latent representations of the users. We propose two different approaches for this purpose. One in which the LDA topic vectors for the items are directly used as the latent representations of the items, and another where these LDA representations are used to initialize the item vectors in the factorization process. For the second approach, the item-latent offset is introduced alongside the LDA representations. The offset is learned throughout the factorization process and tries to capture the deviations from the LDA representations of the items. We call our approach as ***P**reference **Re**lations Based **Fac**tor Model with **T**opic Awareness and **O**ffset* or PReFacTO in short. Experimental evaluation and analysis performed on a benchmark dataset helps to understand the strengths of the pairwise methods and their ability to generate efficient recommendations. We summarize the contribution of our work below:

- We use relative preferences over item pairs in a factor modeling framework for modeling users and items. The models are then used for generating recommendations.
- We incorporate item reviews in the factorization process.
- Detailed experimental evaluation is performed on a benchmark dataset. Analysis of the results are performed to understand the advantages and shortcomings of the methods.

The rest of the paper is organized as follows. After discussing the related work, we present the proposed methods in Section 3. We briefly talk about pairwise preferences and handling textual reviews and then provide detailed description about the methods being proposed in this paper. In Section 4, we define the four evaluation metrics used to measure the performance of the proposed methods with the baseline methods. We provide the detailed discussion and analysis of the results obtained. The conclusion and the future work of this paper has been summarized in Section 5.

2 RELATED WORK

Traditional recommendation systems have extensively used latent factor based modeling techniques. Many researches have been done that employ the use of Matrix Factorization(MF) [8, 9] techniques for the prediction of unknown rating values of items not seen by the user and providing recommendations by selecting top-N items. This basic MF model which corresponds to the pointwise method used in this paper. It acts as a baseline model to compare the proposed methods presented in this paper. The works of [11, 14] have included the content based modeling to interpret the textual labels for the rating dimensions. This justifies the reasons how the user assess the products. Similar kind of work has been done in [5]. It tries to improve the rating predictions and provide feature discovery. Different users give different weights to these features. For e.g., a user who loves horror movies and hates romantic genre will have high weightage to "Annabelle" movie than the "The Notebook" in contrary to a romantic movie lover. This weightage will affect the overall scores and explain the rating difference.

Recently researchers have shown keen interest in pairwise preferences based recommendation techniques. In [2] suitable graphical interface has been provided to the user to mark his choices over the pair of items. In [7] the pairwise preferences are induced from the available rating values of the items. Both implicit [12] and explicit feedback can be modeled using the pairwise preferences based latent factor models. In [3], the users motivate the use of preference relations or relative feedback for recommendation systems. Pairwise preferences have been used in [3, 4, 7, 10] in matrix factorization and nearest neighbor of latent factor modeling settings to generate recommendations. However, in none of these works, the user reviews are taken into account.

3 METHODOLOGY

In this section, we present our proposed recommendation methods that work with pairwise preference information from the user. Apart from the pairwise feedback, we also consider the reviews that are provided by the users for different items. The methods represent each user and item in a shared latent feature space, through factor modeling approach. Before discussing our proposed methods in detail, we briefly describe the concepts of pairwise preferences and also about the way in which we handle the textual reviews available for the items.

Pairwise Preferences: The ratings in recommendation systems are generally absolute in nature, often in the range of 1-5 or 1-10. However, users have different behavior while rating the items. The same rating value entered by two different users might be due to two different satisfaction levels. Moreover, the absolute rating entered by a user to an item may change over time, if the same user is asked to rate the same item again. Motivated by observations like this, pairwise preferences are introduced in modeling users and items in recommendation systems [3]. Pairwise relation based approaches try to capture the relative preference between the items. Such feedback, if directly obtained, removes the user bias that may correspond to the leniency or strictness of the users while assigning the absolute ratings.

Although pairwise preference relations can address some of the problems with absolute ratings mentioned above, there is no dataset (publicly available) with directly obtained pairwise preferences. In absence of such data, we consider in our work, datasets with absolute ratings as user feedback, and induce relative ratings from those absolute ratings. We then consider those relative pairwise preferences as input to the proposed methods.

Handling Textual Reviews - Topic modeling: If the item descriptions are available, then the system can identify more about the attributes or aspects that the items possess. This information can be useful in making the recommendations. In fact, content-based recommendation algorithms try to exploit these item attributes for generating the recommendations.

Several systems allow the users to enter reviews for the items. Item reviews are very useful in making view/purchase decisions as they often contain reasons or explanations regarding why the item was liked or disliked by the user who wrote the review. The reviews often describe some additional details about the items, for example the aspects that they possess. An example review for a product from Amazon is given below.

> It seems like just about everybody has made a Christmas Carol movie. This one is the best by far! It seems more realistic than all the others and the time period seems to be perfect. The acting is also far better than any of the others I've seen; my opinion.

We hypothesize that even if item descriptions are not available, then also, the reviews reveal a great deal of information about the different attributes (specified or latent) that might be contained in the items[1]. These attributes can then be useful in modeling the items, and can further aid in generating efficient recommendations.

Based on this assumption, we use the reviews given by the users to different items as an additional source of information for learning the item representations. We use Latent Dirichlet Allocation (LDA) topic modeling technique to learn the topic representation of the items. LDA is an unsupervised method, which, given a document collection, identifies a fixed-number (say k, input to the algorithm) of latent topics present in the collection. Each document can then be represented as a k-dimensional vector in that topic space. LDA works on documents, so we need to represent each item as a document. For that purpose, we combine all the reviews assigned to an item to create a proxy document for that item. If d_{ui} represents a review given by a user u for an item i, then we denote the proxy document (d_i) for the item i as the concatenation of all the reviews given by the set of users U for i. Then, we can have a document collection d corresponding to the set of items I as $d = \cup_i(d_i)$ where $i = 1, \cdots, |I|$.

3.1 Preference Relation based Factor modeling (Pairwise)

Between the pair of items (i, j), users can express their relative preference if such a provision exists. This would allow the user to indicate, for the item pair, which item he prefers more. The user can also indicate if he favors both the items equally.

This pairwise preference can be captured through an interface where users mark their preferences over a small subset of data. However, as mentioned earlier, we are not aware of the existence of any such system that allows the users to enter the pairwise preferences directly. In absence of that, if the rating data is available, pairwise preferences can be obtained as: $r_{u_{ij}} = r_{ui} - r_{uj}$. Here, r_{ui} indicates the absolute rating given by user u to item i. If the sign of $r_{u_{ij}}$ is positive, we may consider that item i is preferred over item j by the user u. If the sign is negative we may consider that j is preferred over i. If the value of r_{uij} is zero, then it indicates that both the items are equally preferable to u. Similar kind of approach

[1]The dataset used in our experiments did not have the item descriptions, but contained the reviews

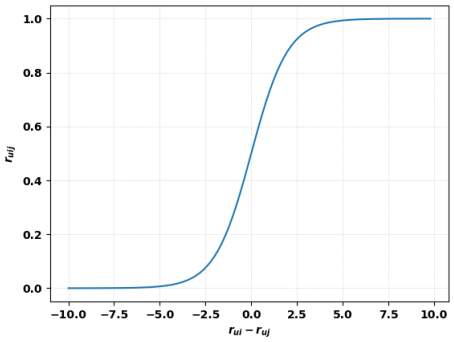

Figure 1: Graph showing pairwise relation between the items as a function of sigmoid.

was adopted in [4] for inducing pairwise preferences from absolute ratings.

We take a different approach for converting the absolute rating to relative preferences. If the ratings given by user u to the two items i and j are r_{ui} and r_{uj} respectively, then we define the (actual or ground truth) preference strength for the triplet (u, i, j) as

$$r_{uij} = \frac{\exp(r_{ui})}{\exp(r_{ui}) + \exp(r_{uj})} = \frac{1}{1 + \exp(-(r_{ui} - r_{uj}))} \tag{1}$$

The value of r_{uij} thus obtained can capture the strength of the preference relation as well. If the difference between r_{ui} and r_{uj} becomes larger, then the strength of this relation becomes stronger as shown in Figure 1.

We model the prediction of the unobserved r_{uij}'s as:

$$\hat{r}_{u_{ij}} = \frac{\exp(p_u(q_i - q_j) + (b_i - b_j))}{1 + \exp(p_u(q_i - q_j) + (b_i - b_j))} = \frac{1}{1 + \exp(-(p_u(q_i - q_j) + (b_i - b_j)))} \tag{2}$$

where the rating matrix R consisting of user-item interaction gives access to the values of r_{ui}, indicating the rating given to item i by user u. The quantity b_i represents the bias for the item. The method models each user u by a vector p_u. This vector space measures user's interest in the particular item based on affinity of user towards these factors. Similarly, each item i is represented by a feature vector q_i. This latent factor representation defines the degree to which these factors are possessed by the item.

Given the training set, the mean-squared error (MSE) function on the training data (with suitable regularization) is used as the objective function. The optimization is generally performed using Stochastic Gradient Descent (SGD) and the algorithm outputs optimized values of the rating parameters $\Theta = \{B, P, Q\}$ where B represents the bias values (b_i) for all the items $i \in I$, P represents the user latent feature vector (p_u) for all the users $u \in U$ and Q represents the item latent feature vector (q_i) for all the items $i \in I$. The objective function is defined as :

$$\min_{\Theta} \sum_{(u,i,j)\in T} \left(r_{uij} - \frac{s_{ij}}{1+s_{ij}}\right)^2 + \lambda||p_u||^2 + \frac{\lambda}{2}||q_i||^2 + \frac{\lambda}{2}||q_j||^2 + \frac{\lambda}{2}||b_i||^2 + \frac{\lambda}{2}||b_j||^2 \quad (3)$$

where

$$s_{ij} = \exp\left(p_u(q_i - q_j) + (b_i - b_j)\right)$$

T represents the training set and λ is the regularization parameter. The update rules for optimizing the above objective function are given below:

Update rules :

$$p_u \leftarrow p_u + \alpha\left(\frac{2e_{uij}s_{ij}(q_i - q_j)}{(1+s_{ij})^2} - 2\lambda p_u\right) \quad (4)$$

$$q_i \leftarrow q_i + \alpha\left(\frac{2e_{uij}s_{ij}p_u}{(1+s_{ij})^2} - \lambda q_i\right) \quad (5)$$

$$q_j \leftarrow q_j - \alpha\left(\frac{2e_{uij}s_{ij}p_u}{(1+s_{ij})^2} + \lambda q_j\right) \quad (6)$$

$$b_i \leftarrow b_i + \alpha\left(\frac{2e_{uij}s_{ij}}{(1+s_{ij})^2} - \lambda b_i\right) \quad (7)$$

$$b_j \leftarrow b_j - \alpha\left(\frac{2e_{uij}s_{ij}}{(1+s_{ij})^2} + \lambda b_j\right) \quad (8)$$

where $e_{uij} = r_{uij} - \frac{s_{ij}}{(1+s_{ij})}$ and α is the learning rate.

After obtaining the model parameters through stochastic gradient descent, we can predict the personalized utility of the item i for the user u as:

$$\rho_{ui} = b_i + p_u q_i \quad (9)$$

The top-N items according to this predicted personalized utility are recommended to the user.

3.2 Preference Relation based Factor modeling with Topics (Pairwise+Topic)

As motivated in the previous section, the review comments about items can be useful in identifying the aspects that the items possess. Moreover, it also helps to understand the reasons behind the liking or disliking of the item by the user. Hence, we extend the previous method to incorporate the reviews about the items. The *proxy documents* for the items are passed through a Latent Dirichlet Allocation (LDA) framework to identify the latent topics present in the documents.

LDA is a probabilistic topic modeling technique that discovers latent topics in the documents. It represents each document d_i by k-dimensional topic distribution θ_i through Dirichlet distribution. The k-th dimension of the vector indicates the probability with which the k-th topic is being discussed in the document. Each topic is associated with the word distribution ϕ_k which is the probability of the word-topic association.

We pass the collection of documents $D = \cup_{i\in I} d_i$ to LDA. As an output, we get the topic vector q_i corresponding to each document $d_i \in D$. For each item i, the latent representation is now fixed at q_i, and these values of q_i's are fed to the factor modeling technique used in Section 3.1. The objective function for this method is given by Equation 10. The optimization variables (parameters) now become $\Theta = \{B, P\}$. The solution to this objective function is obtained through Stochastic Gradient Descent.

$$\min_{\Theta} \sum_{(u,i,j)\in T} \left(r_{uij} - \frac{s_{ij}}{1+s_{ij}}\right)^2 + \lambda||p_u||^2 + \frac{\lambda}{2}||b_i||^2 + \frac{\lambda}{2}||b_j||^2 \quad (10)$$

Here q_i remains fixed throughout the learning process. Hence, we do not have regularization term for q_i in the objective function. The update rules remain same for p_u, b_i and b_j as in Equation 4, 7 and 8 respectively. Personalized utility scores of the items are computed using Equation 9 and recommendations are generated.

3.3 Pairwise Relation based Factor modeling with Topics and Offset (PreFacTO)

In the previous method described in Section 3.2, the topic modeling provides the seed information for the item latent vector representations obtained from the reviews. These representations were fixed throughout the learning process. In our next method, we allow the item representations to take deviations from their LDA topic vectors. If ϵ_i is the deviation of the item i's representation from its topic vector q_i, then the pairwise ratings can be modeled as:

$$\begin{aligned}\hat{r}_{u_{ij}} &= \frac{\exp\left(p_u((q_i+\epsilon_i) - (q_j+\epsilon_j)) + (b_i - b_j)\right)}{1+\exp\left(p_u((q_i+\epsilon_i) - (q_j+\epsilon_j)) + (b_i - b_j)\right)} \\ &= \frac{1}{1+\exp\left(-(p_u((q_i+\epsilon_i) - (q_j+\epsilon_j)) + (b_i - b_j))\right)}\end{aligned} \quad (11)$$

The parameters for this model are $\Theta = \{B, P, \mathcal{E}\}$. As earlier, B and P are the collection of item-bias vectors and user vectors. $\mathcal{E}$ is the collection of deviations or offsets of the items from their LDA topic vectors. The objective function for this model can be written as:

$$\min_{\Theta} \sum_{(u,i,j)\in T} \left(r_{uij} - \frac{s_{ij}}{1+s_{ij}}\right)^2 + \lambda||p_u||^2 + \frac{\lambda}{2}||b_i||^2 + \frac{\lambda}{2}||b_j||^2 + \frac{\lambda}{2}||\epsilon_i||^2 + \frac{\lambda}{2}||\epsilon_j||^2 \quad (12)$$

where $s_{ij} = \exp\left(p_u(q_i+\epsilon_i) - (q_j+\epsilon_j)) + (b_i - b_j)\right)$ and r_{uij} is already defined in Equation 1.

The model parameters are learned using Stochastic Gradient Descent. The update rules are given below:

$$p_u \leftarrow p_u + \alpha\left(\frac{2e_{uij}s_{ij}((q_i+\epsilon_i) - (q_j+\epsilon_j))}{(1+s_{ij})^2} - 2\lambda p_u\right) \quad (13)$$

$$\epsilon_i \leftarrow \epsilon_i + \alpha\left(\frac{2e_{uij}s_{ij}p_u}{(1+s_{ij})^2} - \lambda\epsilon_i\right) \quad (14)$$

$$\epsilon_j \leftarrow \epsilon_j - \alpha\left(\frac{2e_{uij}s_{ij}p_u}{(1+s_{ij})^2} + \lambda\epsilon_j\right) \quad (15)$$

where $e_{uij} = r_{uij} - \frac{s_{ij}}{(1+s_{ij})}$.

The update rules for the bias terms remain same as specified in Equations 7 and 8. After the optimized values of the parameters are obtained, personalized utility of the item i for user u is computed

using following equation and Top-N recommendations are made for each user.

$$\rho_{ui} = b_i + p_u(q_i + \epsilon_i) \tag{16}$$

4 PERFORMANCE EVALUATION

4.1 Dataset

We use the Amazon product review dataset[2] for our experiments. This dataset contains reviews and ratings given to different items by different users. We consider items from the *Movies and TV* category. All items in this category were released between 1999 to 2013. We divided this timeline into three blocks each consisting of 5 year span: (A) 1999-2003, (B) 2004-2008, and (C) 2009-2013. From each block, we removed the items which have less than 10 reviews in that block and the users who have given less than 5 reviews in that block. After this filtering to remove these non-prolific users and items, we have 3,513 items, 85,375 users, 725198 ratings and 725176 reviews in our dataset. We have used 70% of this data for training and the remaining 30% for testing purposes.

4.2 Baseline Methods

We compare our preference relation based models to the following baselines:

(a) **Absolute Rating based Factor modeling (Pointwise):** In analogous to the standard latent-model [8], we convert the absolute rating values using the sigmoid function. The sigmoid function is then used to make predictions using the following objective function:

$$\min_{\Theta} \sum_{(u,i)\in T} \left(\rho_{ui} - \frac{s_i}{1+s_i}\right)^2 + \lambda||p_u||^2 + \frac{\lambda}{2}||q_i||^2 + \frac{\lambda}{2}||b_i||^2$$

where

$$\rho_{ui} = \frac{\exp(r_{ui})}{1+\exp(r_{ui})}$$
$$s_i = \exp(p_u q_i + b_i)$$

(b) **Absolute Rating based Factor modeling with Topics (Pointwise+Topics) :** We combine the topic modeling technique with the latent factor modeling. The latent vector representations of the items are drawn from the reviews (by passing the reviews of the items as an input to the LDA) and fed to latent factor model. Here the item representations will remain fixed and the user-latent space will be learned using the Stochastic Gradient Descent.

(c) **Absolute Rating based Factor modeling with Topics And Offset (Pointwise+Topics+Offset) :** Along with the factor and the topic modeling, we introduce *item latent vector offset* which captures the deviations of the item feature vector representations drawn from the LDA. The objective function to model the system and learn the user-latent and the item-offset representations can be written as:

$$\min_{\Theta} \sum_{(u,i)\in T} \left(\rho_{ui} - \frac{s_i}{1+s_i}\right)^2 + \lambda||p_u||^2 + \frac{\lambda}{2}||b_i||^2 + \frac{\lambda}{2}||\epsilon_i||^2$$

where $s_i = \exp(p_u(q_i + \epsilon_i) + b_i)$

[2] http://jmcauley.ucsd.edu/data/amazon/

4.3 Evaluation

For evaluation of the models presented in Section 3, we compare those three algorithms with the baseline methods mentioned in Section 4.2. We use Precision@k, Recall@k, IRecall and URecall as the evaluation metrics. We took $k = 100$. The IRecall and the URecall metrics are described below.

IRecall: IRecall of an item is computed using the following equation:

$$IRecall_i = \frac{|Rec(i) \cap Rated(i)|}{|Rated(i)|}, \tag{17}$$

where $Rec(i)$ denotes the sets of users to whom item i is recommended. $Rated(i)$ denotes the set of users who have i in their test set. Thus this metric measures the algorithm's ability to recommend items to the users who have actually rated it. IRecall for an algorithm is defined as the average of the item-wise IRecall values over the set of concerned items.

URecall: URecall of a user is computed as:

$$URecall_u = \frac{|Rec(u) \cap Rated(u)|}{|Rated(u)|}, \tag{18}$$

where $Rec(u)$ denotes the sets of items that have been recommended to user u. $Rated(u)$ denotes the set of items present in the test set of user u.

For the experimentation and evaluation purposes, we have divided the items into bins. These bins are created based on the number of reviews. For each block, we maintain item review count written by the user during that time span (block range). We define two bins for each block as follows: Bin-0 consists of the items having review count less than 40 and Bin-1 contains the items having review count greater than or equal to 40. We consider the Bin-0 as a collection of *sparse* items, and the items from Bin-1 as *dense* items. For each bin, we compute the average of the IRecall value of all the items present in the corresponding bin. Analogous to the items, we divide the users as well into the bins based on the number of reviews given by the user. Also, we take the average of the URecall value of all the users falling into the corresponding bin. We then compare the IRecall and URecall values of the different methods mentioned in this paper with the baseline approaches.

4.4 Experimental Analysis And Discussion

Setting the parameters for the proposed method: The model hyperparameters λ (regularization parameter) and k (number of topics) need to be determined in order to produce best models for recommendation. Experiments were conducted with different values of λ and k on a small subset of the data. From the experiments, the combination of $\lambda = 4E-05$ and $k = 10$ were found to be the best values for the parameters. Hence, we select these two values of the hyperparameters for further experimentation. Performance of the algorithm on the test set for different values of λ (keeping k fixed at 10) and different values of k (keeping λ fixed at $4E-05$ are shown in Table 1 and Table 2 respectively.

Comparison with other methods and discussion: For each method, we run the experiments for the three blocks, and compute the average value of each metric over these three blocks. These average values are reported in Table 3. It can be seen from the experimental results that pairwise methods and in particular, PreFacTO

Table 1: Values of the evaluation metrics for different values of λ. Number of topics were fixed at 10.

λ	Precision	Recall	IRecall(reviews<40)	IRecall(reviews>40)	URecall(reviews<40)	URecall(reviews>40)
4.00E-02	0.0076	0.1045	**0.0117**	0.0673	0.1074	0.0863
4.00E-03	0.0122	0.1451	0.0013	0.0793	**0.1448**	0.1456
4.00E-04	0.0120	0.1398	0.0012	0.0789	0.1390	0.1435
4.00E-05	**0.0125**	**0.1457**	0.0012	0.0792	**0.1448**	**0.1504**
4.00E-06	0.0124	0.1448	0.0011	**0.0797**	0.1438	0.1495

Table 2: Values of the evaluation metrics for different values of k: the number of topics. The value of λ was fixed at $4.00E-05$.

No. of Topics	Precision	Recall	IRecall(reviews<40)	IRecall(reviews>40)	URecall(reviews<40)	URecall(reviews>40)
5	0.0107	0.1229	0.0008	0.0781	0.1221	0.1302
10	**0.0125**	**0.1457**	**0.0012**	**0.0792**	**0.1448**	**0.1504**
15	0.0108	0.1246	0.0011	0.0778	0.1238	0.1324
20	0.0108	0.1244	0.0008	0.0784	0.1233	0.1331

Table 3: Comparing performances of different algorithms. The best values for each metric across the algorithms are marked in bold.

Method	Precision	Recall	IRecall(reviews<40)	IRecall(reviews>40)	URecall(reviews<40)	URecall(reviews>40)
Pointwise	0.0106	0.1267	0.0141	0.0635	0.1271	0.1210
Pointwise+Topics	0.0048	0.0555	0.0256	0.0551	0.0551	0.0568
Pointwise+Topics+Offset	0.0055	0.0650	0.0252	0.0514	0.0651	0.0632
Pairwise	0.0021	0.0254	**0.0420**	0.0312	0.0255	0.0252
Pairwise+Topics	0.0038	0.0485	0.0378	0.0399	0.0491	0.0448
PreFacTO	**0.0125**	**0.1457**	0.0012	**0.0792**	**0.1448**	**0.1504**

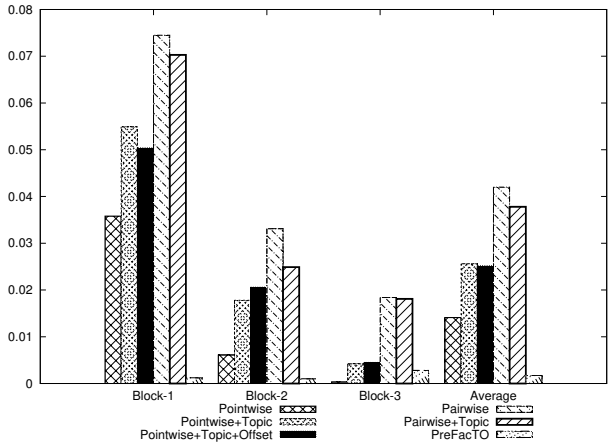

Figure 2: Comparison of IRecall values of different algorithms taking into consideration the items having review count less than 40.

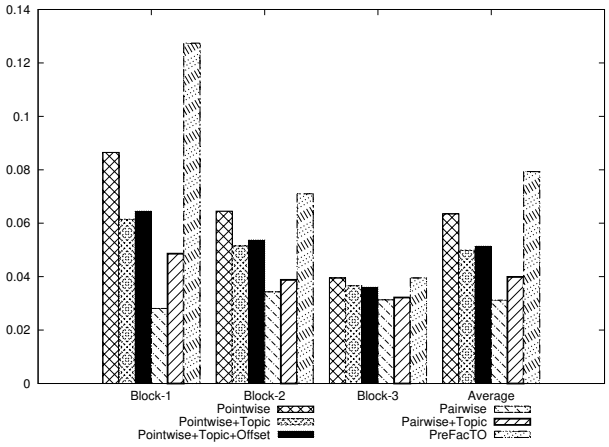

Figure 3: Comparison of IRecall values of different algorithms with review count of the items greater than or equal to 40.

gives the best results compared to other algorithms for the complete dataset. Although the PreFacTO and pointwise are at par based on their performance but the PreFacTO slightly surpasses the pointwise in terms of overall precision and recall values. If we compare the IRecall values for the sparse items, the Pairwise method outperforms all other approaches. The IRecall values for dense items shows that PreFacTO performs very well for dense items. The IRecall values for the sparse and dense items for different blocks are compared in Figure 2 and Figure 3 respectively. There are four groups of columns in both the figures. The first three represent

the three blocks, and the last one represents the average value over the three blocks.

The superior performance of Pairwise and worst performance of PreFacTO in case of the sparse items might be due to the sparseness of the reviews. The LDA representation for the sparse items having very less reviews and further learning in the form of deviations on top of the LDA vectors do not provide any additional benefit. On the contrary, it might have led to overfitting. But on the other hand, Pairwise tries to model the system only through rating information. The preference relations provide some additional information to the item in the process of comparing it with the other items. There is no overfitting in the process and modeling the system for the sparse items works well. If we look at URecall values for the sparse users, the PreFacTO actually performs well.

However, in case of dense items, the PreFacTO outperforms every other method including Pointwise. Along with the pairwise preference based learning, the item vector representation from the rich-textual information of the reviews and learning the deviations from these item vectors help in better prediction with reasoning as to why the item will be likeable or dislikable to the user.

In any real recommendation system, there are sparse items, and there are dense items as well. Depending on the exact system or domain, the ratio of sparse to dense items can vary. In this study, we have explored few algorithms that consider pairwise feedback instead of absolute ratings. Among the proposed methods, Pairwise works well for the sparse items and PreFacTO works well for the dense items. The experiments show the power of preference relation based feedback for recommendation. However, it does not establish the superiority of any single algorithm that works across the entire range of data (both sparse and dense zones). Nonetheless, we believe that it might be possible to design such algorithms that works well for the entire range of data. It might be an interesting research direction to develop hybrid methods that consider both Pairwise and PreFacTo for fusing the recommendations from sparse and dense zones to generate the final recommendations.

5 CONCLUSIONS AND FUTURE WORK

We have presented the PReFacTO approach in this paper, which aligns the latent factor modeling between the users and the item pairs with the hidden topics in the reviews of the item. The pairwise relation adds significant information for the sparse items and provides better modeling of the user-item interaction, and the item hidden dimensions are effectively drawn from the reviews. The topic modeling based latent factors of the items along with the pairwise relation between these items (where the latent feature space of the items drawn from the LDA are allowed to change through offset during the learning process) provides significant improvement over the methods considered in isolation. Our algorithm runs very effectively on large dataset and comparable with the pointwise approach. In fact, PreFacTO method gives marginal improvements over the pointwise methods. It is also shown that Pairwise method works well for the sparse items and PreFacTO provides better performance in case of dense items.

It was observed in the experimental results that Pairwise works well for sparse items and PreFacTO works well for dense items. It might be possible to develop hybrid methods that consider both Pairwise and PreFacTo and fuse the recommendations generated by them from sparse and dense zones to come up with the final recommendations. It might also be possible to develop parameterized algorithms that automatically switch between Pairwise (no consideration of reviews) and PreFacTo (considering the reviews) depending on the availability of data for the item under consideration during the modeling.

REFERENCES

[1] Deepak Agarwal and Bee-Chung Chen. 2009. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 19–28.

[2] Laura Blédaité and Francesco Ricci. 2015. Pairwise preferences elicitation and exploitation for conversational collaborative filtering. In *Proceedings of the 26th ACM Conference on Hypertext & Social Media*. ACM, 231–236.

[3] Maunendra Sankar Desarkar, Sudeshna Sarkar, and Pabitra Mitra. 2010. Aggregating Preference Graphs for Collaborative Rating Prediction. In *Proceedings of the Fourth ACM Conference on Recommender Systems (RecSys '10)*. ACM, New York, NY, USA, 21–28. https://doi.org/10.1145/1864708.1864716

[4] Maunendra Sankar Desarkar, Roopam Saxena, and Sudeshna Sarkar. 2012. Preference relation based matrix factorization for recommender systems. In *International conference on user modeling, adaptation, and personalization*. Springer, 63–75.

[5] Gayatree Ganu, Noemie Elhadad, and Amélie Marian. 2009. Beyond the stars: improving rating predictions using review text content.. In *WebDB*, Vol. 9. Citeseer, 1–6.

[6] Nicolas Jones, Armelle Brun, and Anne Boyer. 2011. Comparisons instead of ratings: Towards more stable preferences. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*. IEEE Computer Society, 451–456.

[7] Saikishore Kalloori, Francesco Ricci, and Marko Tkalcic. 2016. Pairwise preferences based matrix factorization and nearest neighbor recommendation techniques. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 143–146.

[8] Robert Bell Koren, Yehuda and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer (Long Beach, Calif.)* 42 (Aug. 2009), 30–37. https://doi.org/10.1109/MC.2009.263

[9] Yehuda Koren and Robert Bell. 2015. Advances in collaborative filtering. In *Recommender systems handbook*. Springer, 77–118.

[10] Shaowu Liu, Gang Li, Truyen Tran, and Yuan Jiang. 2016. Preference Relation-based Markov Random Fields for Recommender Systems. In *Asian Conference on Machine Learning (Proceedings of Machine Learning Research)*, Geoffrey Holmes and Tie-Yan Liu (Eds.), Vol. 45. PMLR, Hong Kong, 157–172. http://proceedings.mlr.press/v45/Liu15.html

[11] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. *Proceedings of the 7th ACM conference on Recommender systems. ACM, 2013* (Oct. 2013), 165–172. https://doi.org/10.1145/2507157.2507163

[12] Ladislav Peska and Peter Vojtas. 2017. Using implicit preference relations to improve recommender systems. *Journal on Data Semantics* 6, 1 (2017), 15–30.

[13] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on Machine learning*. ACM, 880–887.

[14] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 448–456.