Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews

Zhiyong Cheng
National University of Singapore
jason.zy.cheng@gmail.com

Ying Ding
Vipshop Inc., USA
ian.yingding@gmail.com

Lei Zhu
Shandong Normal University, China
leizhu0608@gmail.com

Mohan Kankanhalli
National University of Singapore
mohan@comp.nus.edu.sg

ABSTRACT

Although latent factor models (e.g., matrix factorization) achieve good accuracy in rating prediction, they suffer from several problems including cold-start, non-transparency, and suboptimal recommendation for local users or items. In this paper, we employ textual review information with ratings to tackle these limitations. Firstly, we apply a proposed aspect-aware topic model (ATM) on the review text to model user preferences and item features from different *aspects*, and estimate the *aspect importance* of a user towards an item. The aspect importance is then integrated into a novel aspect-aware latent factor model (ALFM), which learns user's and item's latent factors based on ratings. In particular, ALFM introduces a weighted matrix to associate those latent factors with the same set of aspects discovered by ATM, such that the latent factors could be used to estimate aspect ratings. Finally, the overall rating is computed via a linear combination of the aspect ratings, which are weighted by the corresponding aspect importance. To this end, our model could alleviate the data sparsity problem and gain good interpretability for recommendation. Besides, an aspect rating is weighted by an aspect importance, which is dependent on the targeted user's preferences and targeted item's features. Therefore, it is expected that the proposed method can model a user's preferences on an item more accurately for each user-item pair locally. Comprehensive experimental studies have been conducted on 19 datasets from Amazon and Yelp 2017 Challenge dataset. Results show that our method achieves significant improvement compared with strong baseline methods, especially for users with only few ratings. Moreover, our model could interpret the recommendation results in depth.

CCS CONCEPTS

• **Information systems** → **Social recommendation**; **Personalization**; **Recommender systems**; *Collaborative filtering*; • **Computing methodologies** → **Topic modeling**; **Factor analysis**;

KEYWORDS

Aspect-aware, Matrix Factorization, Recommendation, Review-aware, Topic Model

This paper is published under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW 2018, April 23–27, 2018, Lyon, France
© 2018 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC BY 4.0 License.
ACM ISBN 978-1-4503-5639-8/18/04.
https://doi.org/10.1145/3178876.3186145

ACM Reference Format:
Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186145

1 INTRODUCTION

When making comments on an item (e.g., *product*, *movie*, and *restaurant*) in the online review/business websites, such as Yelp and Amazon, reviewers also provide an overall rating, which indicates their overall preference or satisfaction towards the corresponding items. Hence, predicting users' overall ratings to unrated items or *personalized rating prediction* is an important research problem in recommender systems. Latent factor models (e.g., matrix factorization [9, 21, 37]) are the most widely used and successful techniques for rating prediction, as demonstrated by the Netflix Prize contest [3]. These methods characterize user's interests and item's features using *latent factors* inferred from rating patterns in user-item rating records. As a typical collaborative filtering technique, the performance of MF suffers when the ratings of items or users are insufficient (also known as the cold-start problem) [17]. Besides, a rating only indicates the overall satisfaction of a user towards an item, it cannot explain the underlying rationale. For example, a user could give a restaurant a high rating because of its delicious food or due to its nice ambience. Most existing MF models cannot provide such fine-grained analysis. Therefore, relying solely on ratings makes these methods hard to explicitly and accurately model users' preferences [17, 23, 26, 35, 36].

Moreover, MF cannot achieve optimal rating prediction locally for each user-item pair, because it learns the latent factors of users ($\boldsymbol{p_u}$) and items ($\boldsymbol{q_i}$) via a global optimization strategy [10]. In other words, $\boldsymbol{p_u}$ and $\boldsymbol{q_i}$ are optimized to achieve a global optimization over all the user-item ratings in the training dataset.[1] As a result, the performance could be severely compromised locally for individual users or items. MF predicts an unknown rating by the dot product of the targeted user u's and item i's latent factors (e.g., $\boldsymbol{p_u^T q_i}$). The overall rating of a user towards an item ($\hat{r}_{u,i}$) is decided by the importance/contribution of all factors. Take the k-th factor as an example, its contribution is $p_{u,k} * q_{i,k}$. For accurate prediction, it is important to accurately capture the importance of each latent factor for a user towards an item. It is well-known that different users may care about different *aspects* of an item. For example, in the domain

[1] In the paper, unless otherwise specified, notations in bold style denote matrices or vectors, and the ones in normal style denote scalars.

of restaurants, some users care more about the taste of *food* while others pay more attention to the *ambience*. Even for the same aspect, the preference of users could be different from each other. For example, in the *food* aspect, some users like *Chinese cuisines* while some others favor *Italian cuisines*. Similarly, the characteristics of items on an aspect could also be different from each other. Thus, it is possible that "a user u prefers item i but dislikes item j on a specific aspect", while "another user u' favors item j more than item i on this aspect". Therefore, in MF, the importance of a latent factor for users towards an item should be treated differently. At first glance, MF achieves the goal as the influence of a factor (e.g., k-th factor) is dependent on both $p_{u,k}$ and $q_{i,k}$ (i.e., $p_{u,k} * q_{i,k}$). However, it is suboptimal to model the importance of a factor by a fixed value of an item or a user. In fact, MF treats each factor of an item with the same importance to all users (i.e., $q_{i,k}$); and similarly, each factor of a user is equally important to all items (i.e., $p_{u,k}$) in rating prediction. Take the previous example, "*a user u prefers item i but dislikes item j on an aspect*", i.e., a factor (e..g, k) in MF), which means $p_{u,k} * q_{i,k}$ should be larger than $p_{u,k} * q_{j,k}$ (i.e., $p_{u,k} * q_{i,k} > p_{u,k} * q_{j,k}$). On the other hand, "*user u' favors item j more than item i on this aspect*", thus $p_{u',k} * q_{j,k}$ should be larger than $p_{u',k} * q_{i,k}$ (i.e., $p_{u',k} * q_{i,k} < p_{u',k} * q_{j,k}$). Because the values of $p_{u,k}$ and $p_{u',k}$ are kept the same when predicting ratings, it is impossible for MF to satisfy the local requirements $p_{u,k} * q_{i,k} > p_{u,k} * q_{j,k}$ and $p_{u',k} * q_{i,k} < p_{u',k} * q_{j,k}$ simultaneously for these user-item pairs. A straightforward solution is to assign different weights (e.g., $w_{u,i,k}$) to different user-item pairs (e.g., $p_{u,k} * q_{i,k}$). However, how to compute a proper weight for each user-item pair is challenging.

A large amount of research effort has been devoted to deal with these weaknesses of MF methods. For example, various types of side information have been incorporated into MF to alleviate the cold-start problem, such as tags [30, 38], social relations [24, 34], reviews [23, 26, 39], and visual features [16]. Among them, the accompanied review of a rating contains important complementary information. It not only encodes the information of user preferences and item features but also explains the underlying reasons for the rating. Therefore, in recent years, many models have been developed to exploit reviews with ratings to tackle the cold-start problem and also enhance the explainability of MF, such as HFT [26], CTR [32], RMR [23], and RBLT [31]. However, a limitation of these models is that they all assume an *one-to-one correspondence relationship* between latent topics (learned from reviews) and latent factors (learned from ratings), which not only limits their flexibility on modeling reviews and ratings but also may not be optimal. In addition, they cannot deal with the suboptimal recommendation for local users or items in MF. In fact, very few studies in literature have considered this problem.

In this paper, we focus on the problem of *personalized rating prediction* and attempt to tackle the above limitations together by utilizing reviews with ratings. Specifically, an *A*spect-aware *T*opic *M*odel (ATM) is proposed to extract *latent topics* from reviews, which are used to model users' preferences and items' features in different *aspects*. In particular, each *aspect* of users/items is represented as a probability distribution of latent topics. Based on the results, the relative importance of an aspect (i.e., *aspect importance*) for a user towards an item can be computed. Subsequently, the aspect importance is integrated into a developed *A*spect-aware *L*atent *F*actor *M*odel (ALFM) to estimate *aspect ratings*. In particular, a weight matrix is introduced in ALFM to associate the latent factors to the same set of aspects discovered by ATM. In this way, our model avoids referring to external sentiment analysis tools for aspect rating prediction as in [12, 42]. The overall rating is obtained by a linear combination of the *aspect ratings*, which are weighted by the importance of corresponding aspects (i.e., *aspect importance*). Note that the latent topics and latent factors in our model are not linked directly; instead, they are correlated via the *aspects* indirectly. Therefore, the number of latent topics and latent factors could be different and separately optimized to model reviews and ratings respectively, which is fundamentally different from the *one-to-one* mapping in previous models [2, 23, 26, 31, 32, 39]. Besides, our model could learn an aspect importance for each user-item pair, namely, assigning a different weight to each $p_{u,k} * q_{i,k}$, and thus could alleviate the suboptimal local recommendation problem and achieve better performance.

A set of experimental studies has been conducted on 19 real-world datasets from Yelp and Amazon to validate the effectiveness of our proposed model. Experimental results show that our model significantly outperforms the state-of-the-art methods which also use both reviews and ratings for rating prediction. Besides, our model also obtains better results for users with few ratings, demonstrating the advantages of our model on alleviating the cold-start problem. Furthermore, we illustrate the interpretability of our model on recommendation results with examples. In summary, the main contributions of this work include:

- We propose a novel aspect-aware latent factor model, which could effectively combine reviews and ratings for rating prediction. Particularly, our model relaxes the constraint of one-to-one mappings between the latent topics and latent factors in previous models and thus could achieve better performance.
- Our model could automatically extract explainable aspects, and learn the aspect importance/weights for different user-item pairs. By associating latent factors with aspects, the aspect weights are integrated with latent factors for rating prediction. Thus, the proposed model could alleviate the suboptimal problem of MF for individual user-item pairs.
- We conduct comprehensive experimental studies to evaluate the effectiveness of our model. Results show that our model is significantly better than previous approaches on tasks of rating prediction, recommendation for sparse data, and recommendation interpretability.

2 RELATED WORK

A comprehensive review on the recommender system is beyond the scope of this work. We mainly discuss the works which utilize both reviews and ratings for rating prediction. Some works assume that the review is available when predicting the rating score, such as SUIT [22], LARAM [33], and recent DeepCoNN [43]. However, in real world recommendation settings, the task should be predicting ratings for the uncommented and unrated items. Therefore, the review is unavailable when predicting ratings. We broadly classify the approaches for the targeted task in three categories: (1)

sentiment-based, (2) topic-based, and (3) deep learning-based. Our approach falls into the second category.

Sentiment-based. These works analyze user's sentiments on items in reviews to boost the rating prediction performance, such as [12, 27, 28, 42]. For example, [27] estimated a sentiment score for each review to build a user-item sentiment matrix, then a traditional collaborative filtering method was applied. Zhang et al. [42] analyzed the sentiment polarities of reviews and then jointly factorize the user–item rating matrix. These methods rely on the performance of external NLP tools for sentiment analysis and thus are not self-contained.

Topic-based. These approaches extract latent topics or aspects from reviews. An early work [14] in this direction relied on domain knowledge to manually label reviews into different aspects, which requires expensive domain knowledge and high labor cost. Later on, most works attempt to extract latent topics or aspects from reviews automatically [2, 12, 17, 23, 26, 26, 31, 36, 39]. A general approach of these methods is to extract latent topics from reviews using topic models [23, 26, 31, 32, 39] or non-negative MF [2, 29] and learn latent factors from ratings using MF methods. HFT [26] and TopicMF [2] link the latent topics and latent factors by using a defined transform function. ITLFM [39] and RBLT [31] assume that the latent topics and latent factors are in the same space, and linearly combine them to form the latent representations for users and items to model the ratings in MF. CTR [32] assumes that the latent factors of items depend on the latent topic distributions of their text, and adds a latent variable to offset the topic distributions of items when modeling the ratings. RMR [23] also learns item's features using topic models on reviews, while it models ratings using a mixture of Gaussian rather than MF methods. Diao et al. [12] propose an integrated graphical model called JMARS to jointly model aspects, ratings and sentiments for movie rating prediction. Those models all assume an one-to-one mapping between the learned latent topics from reviews and latent factors from ratings. Although we adopt the same strategy to extract latent topics and learn latent factors, our model does not have the constraint of one-to-one mapping. Besides, Zhang et al. [42] extracted aspects by decomposing the user–item rating matrix into item–aspect and user–aspect matrices. He et al. [17] extracted latent topics from reviews by modeling the user-item-aspect relation with a tripartite graph.

Deep learning-based. Recently, there has been a trend of applying deep learning techniques in recommendation [11, 19]. For example, He et al. generalized matrix factorization and factorization machines to neural collaborative filtering and achieved promising performance [18, 19]. Textual reviews have also been used in deep learning models for recommendation [6, 40, 41, 43]. The most related works in this direction are DeepCoNN [43] and TransNet [6], which apply deep techniques to reviews for rating prediction. In DeepCoNN, reviews are first processed by two CNNs to learn user's and item's representations, which are then concatenated and passed into a regression layer for rating prediction. A limitation of DeepCoNN is that it uses reviews in the testing phase. [6] shows that the performance of DeepCoNN decreases greatly when reviews are unavailable in the testing phase. To deal with the problem, TransNet [6] extends DeepCoNN by introducing an additional layer to simulate the review corresponding to the target user-item pair. The generated review is then used for rating prediction.

Table 1: Notations and their definitions

Notation	Definition
$\mathcal{D}$	corpus with reviews and ratings
$d_{u,i}$	review document of user u to item i
s	a sentence in a review $d_{u,i}$
$\mathcal{U}$, $\mathcal{I}$, $\mathcal{A}$	user set, item set, and aspect set, respectively
M, N, A	number of users, items, and aspects, respectively
$N_{w,s}$	number of words in a sentence s
K	number of latent topics in ATM
y	an indicator variable in ATM
a_s	assigned aspect a to sentence s
π_u	the parameter of Bernoulli distribution $P(y = 0)$
η	Beta priors ($\eta = \{\eta_0, \eta_1\}$)
$\boldsymbol{\alpha_u}$, $\boldsymbol{\alpha_i}$	Dirichlet priors for aspect-topic distributions
$\boldsymbol{\gamma_u}$, $\boldsymbol{\gamma_i}$	Dirichlet priors for aspect distributions
$\boldsymbol{\beta_w}$	Dirichlet priors for topic-word distributions
$\boldsymbol{\theta_{u,a}}$	user's aspect-topic distribution: denoting user's preference on a
$\boldsymbol{\psi_{i,a}}$	item's aspect-topic distribution: denoting item's features on a
$\boldsymbol{\lambda_u}$, $\boldsymbol{\lambda_v}$	aspect distributions of user and item, respectively
$\boldsymbol{\phi_w}$	topic-text word distribution
f	number of latent factors in ALFM
$\mu_\cdot$	regularization coefficients
$b_\cdot$	bias terms, e.g., b_u, b_i, b_0
w_a	weight vector for aspect a
p_u, q_i	latent factors of user u and item i, respectively
$r_{u,i}$	rating of user u to item i
$r_{u,i,a}$	aspect rating of user u towards item i on aspect a
$\rho_{u,i,a}$	aspect importance of a for u with respect to i
$s_{u,i,a}$	the degree of item i's attributes matching user u's preference on aspect a

3 PROPOSED MODEL

3.1 Problem Setting

Let $\mathcal{D}$ be a collection of reviews of item set $\mathcal{I}$ from a specific category (e.g., restaurant) written by a set of users $\mathcal{U}$, and each review comes with an overall rating $r_{u,i}$ to indicate the overall satisfaction of user u to item i. The primary goal is to predict the unknown ratings of items that the users have not reviewed yet. A review $d_{u,i}$ is a piece of text which describes opinions of user u on different aspects $a \in \mathcal{A}$ towards item i, such as *food* for *restaurants*. In this paper, we only consider the case that all the items are from the same category, i.e., they share the same set of aspects $\mathcal{A}$. Aspects that users care for items are latent and learned from reviews by our proposed topic model, in which each aspect is represented as a distribution of the same set (e.g., K) of latent topics. Table 1 lists the key notations. Before introducing our method, we would like to first clarify the concepts of *aspects*, *latent topics*, and *latent factors*.

- **Aspect** - it is a high-level semantic concept, which represents the attribute of items that users commented on in reviews, such as *"food"* for *restaurant* and *"battery"* for *mobile phones*.
- **Latent topic & latent factor** - in our context, both concepts represent a more fine-grained concept than *"aspect"*. A latent topic or factor can be regarded as a *sub-aspect* of an item. For instance, for the "food" aspect, a related latent topic could be "*breakfast*" or "*Italian cuisine*". We adopt the terminology of

latent topic in topic models and *latent factor* in matrix factorization. Accordingly, "latent topics" are discovered by topic model on reviews, and "latent factors" are learned by matrix factorization on ratings.

3.2 Aspect-aware Latent Factor Model

Based on the observations that (1) different users may care for different aspects of an item and (2) users' preferences may differ from each other for the same aspect, we claim that the overall satisfaction of a user u towards an item i (i.e., the overall rating $r_{u,i}$) depends on u's satisfaction on each aspect a of i (i.e., *aspect rating* $r_{u,i,a}$) and the importance of each aspect (of i) to u (i.e., *aspect importance* $\rho_{u,i,a}$). Based on the assumptions, the overall rating $r_{u,i}$ can be predicted as:

$$\hat{r}_{u,i} = \sum_{a \in \mathcal{A}} \overbrace{\rho_{u,i,a}}^{\text{aspect importance}} \quad \underbrace{r_{u,i,a}}_{\text{aspect rating}} \tag{1}$$

3.2.1 Aspect rating estimation. Aspect rating (i.e., $r_{u,i,a}$) reflects the satisfaction of a user u towards an item i on the aspect a. To receive a high aspect rating $r_{u,i,a}$, an item should at least possess the characteristics/attributes in which the user is interested in this aspect. Moreover, the item should satisfy user's expectations on these attributes in this aspect. In other words, the item's attributes on this aspect should be of high quality such that the user likes it. Take the "food" aspect as an example, for a user who likes Chinese cuisines, to receive a high rating on the *"food"* aspect from the user, a restaurant should provide Chinese dishes and the dishes should suit the user's taste. Based on user's text reviews, we can learn users' preferences and items' characteristics on each aspect and measure *how the attributes of an item i on aspect "a" suit a user u's requirements on this aspect*, denoted by $s_{u,i,a}$. We compute $s_{u,i,a}$ based on results of the proposed Aspect-aware Topic Model (ATM) (described in Sect. 3.3), in which user's preferences and item's characteristics on each aspect are modeled as multinomial distributions of latent topics, denoted by $\boldsymbol{\theta_{u,a}}$ and $\boldsymbol{\psi_{i,a}}$, respectively. $s_{u,i,a} \in [0,1]$ is then computed as :

$$s_{u,i,a} = 1 - JSD(\boldsymbol{\theta_{u,a}}, \boldsymbol{\psi_{i,a}}) \tag{2}$$

where $JSD(\boldsymbol{\theta_{u,a}}, \boldsymbol{\psi_{i,a}})$ denotes the Jensen–Shannon divergence [13] between $\boldsymbol{\theta_{u,a}}$ and $\boldsymbol{\psi_{i,a}}$. Notice that a large value of $s_{u,i,a}$ does not mean a high rating $r_{u,i,a}$ - an item providing all the features that a user u requires does not mean that it satisfies u's expectations, since the provided ones could be of low quality. For instance, a restaurant provides all the Chinese dishes the user u likes (i.e., high score $s_{u,i,a}$), but these dishes taste bad from u's opinion (i.e., low rating $r_{u,i,a}$). Therefore, we can expect that for this restaurant: users discuss its Chinese dishes in reviews with negative opinions and thus give low ratings. Instead of analyzing the review sentiments for aspect rating estimation by using external NLP tools (such as [42]), we refer to the matrix factorization (MF) [21] technique.

MF maps users and items into a latent factor space and represents users' preferences and items' features by f-dim latent factor vectors (i.e., $\boldsymbol{p_u} \in \mathbb{R}^{f \times 1}$ and $\boldsymbol{q_i} \in \mathbb{R}^{f \times 1}$). The dot product of the user's and item's vectors ($\boldsymbol{p_u^T q_i}$) characterizes the user's overall interests on the item's characteristics, and is thus used to predict the rating $r_{u,i}$. To extend MF for aspect rating prediction, we introduce a binary matrix $\boldsymbol{W} \in \mathbb{R}^{f \times A}$ to associate the latent factors to different aspects, where A is the number of aspects considered. We call this model aspect-aware latent factor model (ALFM), in which the weight vector $\boldsymbol{w_a}$ in the a-th column of $\boldsymbol{W}$ indicates which factors are related to the aspect a. Thus, $\boldsymbol{p_{u,a}} = \boldsymbol{w_a} \odot \boldsymbol{p_u}$ denotes user's interests in the aspect a, where $\odot$ represents element-wise product between vectors. Therefore, $(\boldsymbol{p_{u,a}})^T(\boldsymbol{q_{i,a}})$ represents the aspect rating of user u to item i on aspect a. Finally, we integrate the matching results of aspects (i.e., $s_{u,i,a}$) into ALFM to estimate the aspect ratings:

$$r_{u,i,a} = s_{u,i,a} \cdot (\boldsymbol{w_a} \odot \boldsymbol{p_u})^T(\boldsymbol{w_a} \odot \boldsymbol{q_i}) \tag{3}$$

As a high aspect rating $r_{u,i,a}$ requires large values of both $s_{u,i,a}$ and $(\boldsymbol{w_a} \odot \boldsymbol{p_u})^T(\boldsymbol{w_a} \odot \boldsymbol{q_i})$, it is expected that the results learned from reviews could guide the learning of latent factors.

3.2.2 Aspect importance estimation. We rely on user reviews to estimate $\rho_{u,i,a}$, as users often discuss their interest topics of aspects in reviews, such as different *cuisines* in the *food* aspect. In general, the more a user comments on an aspect in reviews, the more important this aspect is (to this user). Thus, we estimate the importance of an aspect according to the possibility of a user writing review comments on this aspect. When writing a review, some users tend to write comments from the aspects according to their own preferences, while others like commenting on the most notable features of the targeted item. Based on this consideration, we introduce (1) π_u to denote the probability of user u commenting an item based on his own preference and (2) $\lambda_{u,a}$ ($\sum_{a \in \mathcal{A}} \lambda_{u,a} = 1$) to denote the probability of user u commenting on the aspect a based on his own preference. Accordingly, $(1 - \pi_u)$ denotes the probability of the user commenting from the item i's characteristics ($\sum_{a \in \mathcal{A}} \lambda_{i,a} = 1$), and $\lambda_{i,a}$ is the probability of user u commenting item i from the item's characteristics on the aspect a. Thus, the probability of a user u commenting an item i on an aspect a (i.e., $\rho_{u,i,a}$) is:

$$\rho_{u,i,a} = \pi_u \lambda_{u,a} + (1 - \pi_u)\lambda_{i,a} \tag{4}$$

$\lambda_{u,a}$, $\lambda_{i,a}$, and π_u are estimated by ATM, which simulates the process of a user writing a review, as detailed in the next subsection.

3.3 Aspect-aware Topic Model

Given a corpus $\mathcal{D}$, which contains reviews of users towards items $\{d_{u,i} | d_{u,i} \in \mathcal{D}, u \in \mathcal{U}, i \in \mathcal{I}\}$, we assume that a set of latent topics (i.e., K topics) covers all the topics that users discuss in the reviews. $\boldsymbol{\lambda_u}$ is a probability distribution of aspects in user u's preferences, in which each value $\lambda_{u,a}$ denotes the relative importance of an aspect a to the user u. Similarly, $\boldsymbol{\lambda_i}$ is the probability distribution of aspects in item i's characteristics, in which each value $\lambda_{i,a}$ denotes the importance of an aspect a to the item i. As the K latent topics cover all the topics discussed in reviews, an aspect will only relate to some of the latent topics closely. For example, topic "*breakfast*" is closely related to aspect "food", while it is not related to aspects like "*service*" or "*price*". The relation between aspects and topics is also represented by a probabilistic distribution, i.e., $\boldsymbol{\theta_{u,a}}$ for users and $\boldsymbol{\psi_{i,a}}$ for items. More detailedly, the interests of a user u in a specific aspect a is represented by $\boldsymbol{\theta_{u,a}}$, which is a multinomial distribution of the latent topics; the characteristics of an item i in a specific aspect a is represented by $\boldsymbol{\psi_{i,a}}$, which is also a multinomial distribution of the same set of latent topics. $\boldsymbol{\theta_{u,a}}$ is determined

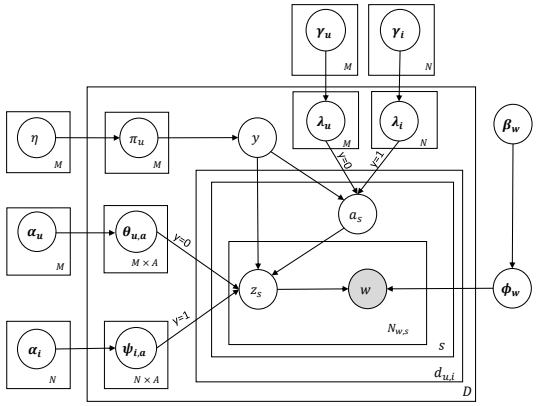


Figure 1: The graphical representation of the ATM model.

based on all the reviews $\{d_{u,i}|i \in \mathcal{I}\}$ of user u writing for items. $\boldsymbol{\psi_{i,a}}$ is learned from all the reviews $\{d_{u,i}|u \in \mathcal{U}\}$ of i written by users. A latent topic is a multinomial distribution of text words in reviews. Based on these assumptions, we propose an aspect-aware topic model ATM to estimate the parameters $\{\boldsymbol{\lambda_i}, \boldsymbol{\lambda_i}, \boldsymbol{\theta_{u,a}}, \boldsymbol{\psi_{i,a}}, \pi_u\}$ by simulating the generation of the corpus $\mathcal{D}$.

The graphical representation of ATM is shown in Fig. 1. In the figure, the shaded circles indicate observed variables, while the unshaded ones represent the latent variables. ATM mimics the processing of writing a review sentence by sentence. A sentence usually discusses the same topic z, which could be from user's preferences or from item's characteristics. To decide the topic z_s for a sentence s, our model introduces an indicator variable $y \in \{0, 1\}$ based on a Bernoulli distribution, which is parameterized by π_u. Specifically, when $y = 0$, the sentence is generated from user's preference; otherwise, it is generated according to item i's characteristics. π_u is user-dependent, indicating the tendency to comment from u's personal preferences or from the item i's characteristics is determined by u's personality. The generation process of ATM is shown in Algorithm 1. Let a_s denote the aspect assigned to a sentence s. If $y = 0$, a_s is drawn from λ_u and z_s is then generated from u's preferences on aspect a_s: $\boldsymbol{\theta_{u,a_s}}$; otherwise, if $y = 1$, a_s is drawn from λ_i and z_s is then generated from i's characteristics on aspect a_s: $\boldsymbol{\psi_{i,a_s}}$. Then all the words w in sentence s is generated from z_s according to the word distribution: $\boldsymbol{\phi_{z_s,w}}$.

In ATM, $\boldsymbol{\alpha_u}, \boldsymbol{\alpha_i}, \boldsymbol{\gamma_u}, \boldsymbol{\gamma_i}, \boldsymbol{\beta}$, and η are pre-defined hyper-parameters and set to be symmetric for simplicity. Parameters need to be estimated including $\boldsymbol{\lambda_i}, \boldsymbol{\lambda_i}, \boldsymbol{\theta_{u,a}}, \boldsymbol{\psi_{i,a}}$, and π_u. Different approximate inference methods have been developed for parameter estimation in topic models, such as variation inference [5] and collapsed Gibbs sampling [15]. We apply collapsed Gibbs sampling to infer the parameters, since it has been successfully applied in many large scale applications of topic models [7, 8]. Due to the space limitation, we omit the detailed inference steps in this paper.

3.4 Model Inference

With the results of ATM, $\rho_{u,i,a}$ and $s_{u,i,a}$ can be computed using Eq. 4 and 2, respectively. With the consideration of bias terms (i.e., b_u, b_i, b_0) in ALFM, the overall rating can be estimated as[2],

$$\hat{r}_{u,i} = \sum_{a \in \mathcal{A}} (\rho_{u,i,a} \cdot s_{u,i,a} \cdot (\boldsymbol{w_a} \odot \boldsymbol{p_u})^T (\boldsymbol{w_a} \odot \boldsymbol{q_i})) + b_u + b_i + b_0 \quad (5)$$

where b_0 is the average rating, b_u and b_i are user and item biases, respectively. The estimation of parameters is to minimize the rating prediction error in the training dataset. The optimization objective function is

$$\min_{p*,q*} \frac{1}{2} \sum_{u,i} (r_{u,i} - \hat{r}_{u,i})^2 + \frac{\mu_u}{2} ||\boldsymbol{p_u}||_2^2 + \frac{\mu_i}{2} ||\boldsymbol{q_i}||_2^2 + \mu_w \sum_a ||\boldsymbol{w_a}||_1 + \frac{\mu_b}{2} (||b_u||_2^2 + ||b_i||_2^2); \quad (6)$$

where $||\cdot||_2$ denotes the ℓ_2 norm for preventing model overfitting, and $||\cdot||_1$ denotes the ℓ_1 norm. μ_u, μ_i, μ_w, and μ_b are regularization parameters, which are tunable hyper-parameters. In practice, we relax the binary requirement of $\boldsymbol{w_a}$ by using ℓ_l norm. It is well known that ℓ_l regularization yields sparse solution of the weights [25]. The ℓ_2 regularization of $\boldsymbol{p_u}$ and $\boldsymbol{q_i}$ prevents them to have arbitrarily large values, which would lead to arbitrarily small values of $\boldsymbol{w_a}$.

Optimization. We use the stochastic gradient descent (SGD) algorithm to learn the parameters by optimizing the objective function in Eq. 6. In each step of SGD, the localized optimization is performed on a rating $r_{u,i}$. Let L denote the loss, and the gradients of parameters are given as follows:

$$\frac{\partial L}{\partial p_u} = \sum_{i=1}^{N} (\sum_a \rho_{u,i,a} s_{u,i,a} w_a^2)(\hat{r}_{u,i} - r_{u,i}) q_i + \mu_u p_u$$

$$\frac{\partial L}{\partial q_i} = \sum_{u=1}^{M} (\sum_a \rho_{u,i,a} s_{u,i,a} w_a^2)(\hat{r}_{u,i} - r_{u,i}) p_u + \mu_i q_i$$

$$\frac{\partial L}{\partial w_a} = \sum_{u=1}^{M} \sum_{i=1}^{N} \rho_{u,i,a} s_{u,i,a} (\hat{r}_{u,i} - r_{u,i}) p_u q_i w_a + \frac{\mu_w w_a}{\sqrt{(w_a^2 + \epsilon)}}$$

Algorithm 1: Generation Process of ATM

```
for each topic k = 1, ..., K do
    Draw φ_{k,w} ~ Dir(·|β_w);
for each user u ∈ U do
    Draw λ_u ~ Dir(·|γ_u);
for each item i ∈ I do
    Draw λ_i ~ Dir(·|γ_i);
for each user u ∈ U, each aspect a ∈ A do
    Draw θ_{u,a} ~ Dir(·|α_u);
for each item i ∈ I, each aspect a ∈ A do
    Draw ψ_{i,a} ~ Dir(·|α_i);
for each review d_{u,i}, u ∈ U, i ∈ I do
    for each sentence s ∈ d_{u,i} do
        Draw y ~ Bernoulli(·|π_u);
        if y_s == 0 then
            Draw a_s ~ Multi(λ_u) and then draw z_s ~ Multi(θ_{u,a_s});
        if y_s == 1 then
            Draw a_s ~ Multi(λ_i) and then draw z_s ~ Multi(ψ_{i,a_s});
        for each word w ∈ s do
            Draw w ~ Multi(φ_{z_s,w})
```

[2]In our experiments, we tried to normalize $\rho_{u,i,a}$ or $\rho_{u,i,a} \cdot s_{u,i,a}$ in Eq. 5, but no improvement has been observed.

Table 2: Statistics of the evaluation datasets

Datasets	#users	#items	#ratings	Sparsity
Instant Video	4,902	1,683	36,486	0.9956
Automotive	2,788	1,835	20,218	0.9960
Baby	17,177	7,047	158,311	0.9987
Beauty	19,766	12,100	196,325	0.9992
Cell Phones	24,650	10,420	189,255	0.9993
Clothing	34,447	23,026	277,324	0.9997
Digital Music	5,426	3,568	64,475	0.9967
Grocery	13,979	8,711	149,434	0.9988
Health	34,850	18,533	342,262	0.9995
Home & Kitchen	58,901	28,231	544,239	0.9997
Musical Instruments	1,397	900	10,216	0.9919
Office Products	4,798	2,419	52,673	0.9955
Patio	1,672	962	13,077	0.9919
Pet Supplies	18,070	8,508	155,692	0.9990
Sports & Outdoors	31,176	18,355	293,306	0.9995
Tools & Home	15,438	10,214	133,414	0.9992
Toys & Games	17,692	11,924	166,180	0.9992
Video Games	22,348	10,672	228,164	0.9990
Yelp 2017	169,257	63,300	1,659,678	0.9998

Here, we omit the gradients of b_u and b_i, as they are the same as in the standard biased MF [21]. M and N are the total number of users and items in the dataset. Notice that in the deriving of the gradient for w_a, we use $\sqrt{w_a^2 + \epsilon}$ in place of $||w_a||_1$, because ℓ_1 norm is not differentiable at 0. ϵ can be regarded as a "smoothing parameter" and is set to 10^{-6} in our implementation.

4 EXPERIMENTAL STUDY

To validate the assumptions when designing the model and evaluate our proposed model, we conducted comprehensive experimental studies to answer the following questions:

- **RQ1:** How do the important parameters (e.g., the number of latent topics and latent factors) affect the performance of our model? More importantly, is the setting $f = K$ optimal, which is a default assumption for many previous models? (Sect. 4.3)
- **RQ2:** Can our ALFM model outperform the state-of-the-art recommendation methods, which consider both ratings and reviews, on rating prediction? (Sect. 4.4)
- **RQ3:** Compared to other methods which also use textual reviews and ratings, how does our ALFM model perform on the cold-start setting when users have only few ratings? (Sect. 4.5)
- **RQ4:** Can our model explicitly interpret the reasons for a high or low rating? (Sect. 4.6)

4.1 Dataset Description

We conducted experiments on two publicly accessible datasets that provide user review and rating information. The first dataset is Amazon Product Review dataset collected by [26][3], which contains product reviews and metadata from Amazon. This dataset has been widely used for rating prediction with reviews and ratings in previous studies [6, 23, 26, 31]. The dataset is organized into 24 product categories. In this paper, we used 18 categories (See Table 2) and focus on the 5-core version, with at least 5 reviews for each user or item. The other dataset is from Yelp Dataset Challenge 2017[4], which includes reviews of local business in 12 metropolitan areas across 4 countries. For the Yelp 2017 dataset, we also processed it to keep users and items with at least 5 reviews. From each review in these datasets, we extract the corresponding "userID", "itemID", a rating score (from 1 to 5 rating stars), and a textual review for experiments. Notice that for all the datasets, we checked and removed the duplicates, and then filtered again to keep them as 5-core. Besides, we removed the infrequent terms in the reviews for each dataset.[5] Some statistics of the datasets are shown in Table 2.

4.2 Experimental Settings

For each dataset, we randomly split it into training, validation, and testing set with ratio 80:10:10 for each user as in [6, 23, 26]. Because we take the 5-core dataset where each user has at least 5 interactions, we have at least 3 interactions per user for training, and at least 1 interaction per user for validation and testing. Note that we only used the review information in the training set, because the reviews in the validation or testing set are unavailable during the prediction process in real scenarios. The number of aspect is set to 5 in experiments.[6]

Baselines: We compare the proposed **ALFM** model with the following baselines. It is worth noting that these methods are tuned on the validation dataset to obtain their optimal hyper-parameter settings for fair comparisons.

- **BMF [21].** It is a standard MF method with the consideration of bias terms (i.e., user biases and item biases). This method only leverages ratings when modeling users' and items' latent factors. It is typically a strong baseline model in collaborative filtering [21, 23].
- **HFT [26].** It models ratings with MF and review text with latent topic model (e.g., LDA [5]). We use it as a representative of the methods which use an exponential transformation function to link the latent topics with latent factors, such as TopicMF [2]. The topic distribution can be modeled on either users or items. We use the topic distribution based on items, since it achieves better results. Note that in experiments, we add bias terms into HFT, which can achieve better performance.
- **CTR [32].** This method also utilizes both review and rating information. It uses a topic model to learn the topic distribution of items, which is then used as the latent factors of items in MF with an addition of a latent variable.

[3] http://jmcauley.ucsd.edu/data/amazon/
[4] http://www.yelp.com/dataset_challenge/
[5] The thresholds of infrequent terms varied across different datasets. For example, for the "Yelp 2017" dataset, which is relatively large, a term that appears less than 10 times in reviews is defined as an infrequent term; and the thresholds are smaller for relatively small datasets (e.g., the threshold is 5 for the "Music Instruments" dataset.)
[6] We tuned the number of aspects from 1 to 8 for all the datasets, and found that the performance does not change much unless setting the aspect number to 1 or 2.

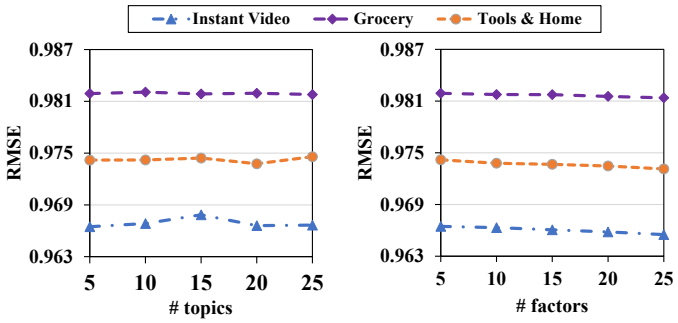
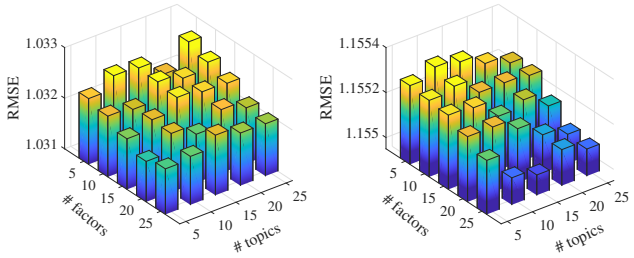


Figure 2: Effects of factors and topics in our model.



(a) Clothing **(b) Yelp 2017**

Figure 3: Effects of #factors v.s. #topics.

- **RMR [23].** This method also uses both ratings and reviews. Different from HFT and CTR, which use MF to model rating, it uses a mixture of Gaussian to model the ratings.
- **RBLT [31].** This method is the most recent method, which also uses MF to model ratings and LDA to model review texts. Instead of using an exponential transformation function to link the latent topics and latent factors (as in HFT [26]), this method linearly combines the latent factors and latent topics to represent users and items, with the assumption that the dimensions of topics and latent factors are equal and in the same latent space. The same strategy is also adopted by ITLFM [39]. Here, we use RBLT as a representative method for this strategy.
- **TransNet [6].** This method adopts neural network frameworks for rating prediction. In this model, the reviews of users and items are used as input to learn the latent representations of users and items. More descriptions about this method could be found in Section 2. We used the codes published by the authors in our experiments and tuned the parameters as described in [6].

The standard root-mean-square error (**RMSE**) is adopted in evaluation. A smaller RMSE value indicates better performance.

4.3 Effect of Important Parameters (RQ1)

In this subsection, we analyze the influence of *the number of latent factors* and *the number of latent topics* on the final performance of ALFM. As we know, in MF, more latent factors will lead to better performance unless overfitting occurs [20, 21]; while the optimal number of latent topics in topic models (e.g., LDA) is dependent on the datasets [1, 4]. Accordingly, the optimal number of latent topics in topic model and the optimal number of latent factors in MF should be tuned separately. However, in the previous latent factor models (e.g., HFT, TopicMF [2], RMR, CTR, and RBLT), the number of factors (i.e., #factors) and the number of topics (i.e., #topics) are assumed to be the same, and thus cannot be optimized separately. Since our model does not have such constraint, we studied the effects of #factors and #topics individually. Fig. 2 show the performance variations with the change of #factors and #topics by setting the other one to 5. We only visualize the performance variations of three datasets, due to the space limitation and the similar performance variation behaviors of other datasets. From the figure, we can see that with the increase of #factors, RMSE consistently decreases although the degree of decline is small. Notice that in our model, the rating prediction still relies on MF technique (Eq. 5). Therefore, the increase of #factors could lead to better representation capability and thus more accurate prediction. In contrast, the optimal number of latent topics is different from dataset to dataset.

To better visualize the impact of #factors and #topics, we also present 3D figures by varying the number of factors and topics in $\{5, 10, 15, 20, 25\}$, as shown in Fig. 3. In this figure, we use the performance of three datasets as illustration. From the figure, we can see that the optimal numbers of topics and latent factors are varied across different datasets. In general, more latent factors usually lead to better performance, while the optimal number of latent topics is dependent on the reviews of different datasets. This also reveals that setting #factors and #topics to be the same may not be optimal.

4.4 Model Comparison (RQ2)

We show the performance comparisons of our ALFM with all the baseline methods in Table 3, where the best prediction result on each dataset is in bold. For fair comparison, we set the number of latent factors (f) and the number of latent topics (K) to be the same as $f = K = 5$. Notice that our model could obtain better performance when setting f and K differently. Still, ALFM achieves the best results on 18 out of the 19 datasets. Compared with BMF, which only uses ratings, we achieve much better prediction performance (16.49% relative improvement on average). More importantly, our model outperforms CTR and RMR with large margins - 6.28% and 8.18% relative improvements on average, respectively. Compared to the recently proposed RBLT and TransNet, ALFM can still achieve 3.37% and 4.26% relative improvement on average respectively with significance testing. It is worth mentioning that HFT achieves better performance than RMR and comparable performance with recent RBLT, because we added bias terms to the original HFT in [26]. TransNet applies neural networks, which has exhibited strong capabilities on representation learning, in reviews to learn users' preferences and items' characteristics for rating prediction. However, it may suffer from (1) noisy information in reviews, which would deteriorate the performance; and (2) errors introduced when generating fake reviews for rating prediction, which will also cause bias in the final performance. Compared to those baselines, the advantage of ALFM is that it models users' preferences on different aspects; and more importantly, it captures a user's specific attention on each aspect of a targeted item. The substantial improvement of ALFM over those baselines demonstrates the benefits of modeling users' specific preferences on each aspect of different items.

Table 3: Comparisons of adopted methods in terms of RMSE with $f = K = 5$.

Datasets	BMF	HFT	CTR	RMR	RBLT	TransNet	ALFM	Improvement(%)					
	(a)	(b)	(c)	(d)	(e)	(f)	(g)	g vs. a	g vs. b	g vs. c	g vs. d	g vs. e	g vs. f
Instant Video	1.162	0.999	1.014	1.039	0.978	0.996	**0.967**	16.79	3.19	4.63*	6.94*	1.12**	2.88
Automotive	1.032	0.968	1.016	0.997	0.924	0.918	**0.885**	14.26*	8.58**	12.86*	11.19*	4.24**	3.56*
Baby	1.359	1.112	1.144	1.178	1.122	1.110	**1.076**	20.83**	3.24	5.98*	8.66**	4.11	3.05*
Beauty	1.342	1.132	1.171	1.190	1.117	1.123	**1.082**	19.39**	4.47	7.65**	9.12**	3.18**	3.65**
Phones	1.432	1.216	1.271	1.289	1.220	1.207	**1.167**	18.47**	3.98*	8.18*	9.4**	4.33	3.27**
Clothing	1.073	1.103	1.142	1.145	1.073	1.064	**1.032**	3.8**	6.47**	9.65	9.9**	3.86**	2.96*
Digital Music	1.093	**0.918**	0.921	0.960	**0.918**	1.061	0.920	15.82	-0.15	0.13*	4.49**	-0.15**	4.13**
Grocery	1.192	1.016	1.045	1.061	1.012	1.022	**0.982**	17.66**	3.36**	6.07	7.46**	3.01**	3.94*
Health	1.263	1.073	1.105	1.135	1.070	1.114	**1.042**	17.48*	2.83	5.65*	8.20	2.56**	6.46**
Home & Kitchen	1.297	1.083	1.123	1.149	1.086	1.123	**1.049**	19.16**	3.15**	6.62	8.7**	3.41**	6.61**
Musical Instruments	1.004	0.972	0.979	0.983	0.946	0.901	**0.893**	11.08	8.17**	8.83**	9.2**	5.61	0.95
Office Products	1.025	0.879	0.898	0.934	0.872	0.898	**0.848**	17.29**	3.55**	5.61*	9.26**	2.77**	5.67**
Patio	1.180	1.041	1.062	1.077	1.032	1.046	**1.001**	15.19**	3.84*	5.7*	7.07*	2.96	4.33**
Pet Supplies	1.367	1.137	1.177	1.200	1.139	1.149	**1.099**	19.64**	3.41*	6.67*	8.41	3.54**	4.38**
Sports & Outdoors	1.130	0.970	0.998	1.019	0.964	0.990	**0.933**	17.42**	3.8*	6.47	8.4*	3.2**	5.77**
Tools & Home	1.168	1.013	1.047	1.090	1.011	1.041	**0.974**	16.63**	3.90	6.98	10.68**	3.7**	6.51**
Toys & Games	1.072	0.926	0.948	0.974	0.923	0.951	**0.902**	15.81**	2.59*	4.82**	7.39**	2.3**	5.11*
Video Games	1.321	1.096	1.115	1.150	1.094	1.123	**1.070**	19.02*	2.43	4.03**	6.97*	2.24**	4.77*
Yelp 2017	1.415	1.174	1.233	1.266	1.202	1.190	**1.155**	18.35*	1.60**	6.33**	8.74*	3.88**	2.92*
Average	1.207	1.044	1.074	1.097	1.037	1.049	**1.004**	14.56**	2.84*	7.16**	8.31*	3.37**	4.26**

The improvements with * are significant with $p - value < 0.05$, and the improvements with ** are significant with $p - value < 0.01$ with a two-tailed paired t-test.

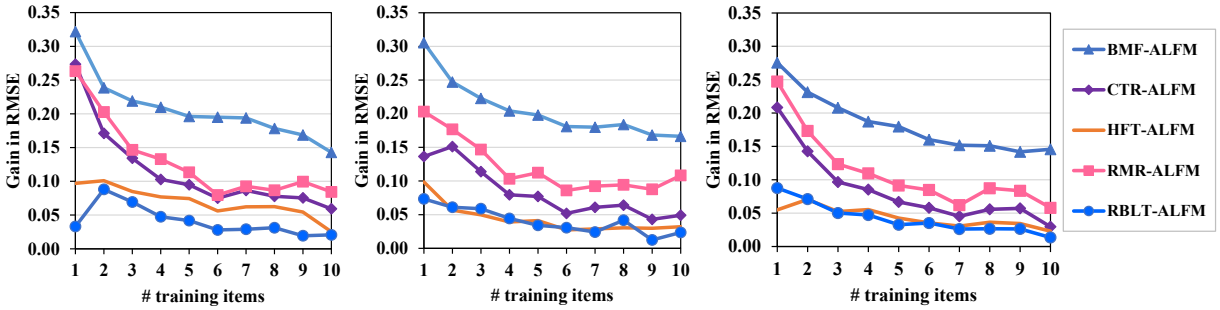


Figure 4: Gain in RMSE for user with limited training data on two individual datasets and overall 19 datasets.

4.5 Cold-Start Setting (RQ3)

As shown in Table 2, the datasets are usually very sparse in practical systems. It is inherently difficult to provide satisfactory recommendation based on limited ratings. In the matrix factorization model, given a few ratings, the penalty function tends to push q_u and p_i towards zero. As a result, such users and items are modeled only with the bias terms [23]. Therefore, matrix factorization easily suffers from the cold-start problem. By integrating reviews in users' and items' latent factor learning, our model could alleviate the problem of cold-start to a great extent, since reviews contain rich information about user preferences and item features.

To demonstrate the capability of our model in dealing with users with very limited ratings, we randomly split the datasets into training, validation, and testing sets in ratio 80:10:10 based on the number of ratings in each set. In this setting, it is not guaranteed that a user has at least 3 ratings in the training set. It is possible that a user has no rating in the training set. For the users without any ratings in the training set, we also removed them in the testing set. Then we evaluate the performance of users who have the number of ratings from 1 to 10 in the training set. In Fig. 4, we show the **Gain in RMSE** (y-axis) grouped by the number of ratings (x-axis) of users in the training set. The value of **Gain in RMSE** is equal to the average RMSE of baselines *minus* that of our model (e.g., "BMF-TALFM"). A positive value indicates that our model achieves

Table 4: Top ten words of each aspect for a user (index 1511) from *Clothing*. Each column is corresponding to an aspect attached with an "interpretation" label.

Value	Comfort	Accessories	Shoes	Clothing
price	size	ring	socks	shirt
color	fit	pretty	foot	back
quality	wear	dress	boots	bra
worth	comfortable	time	comfort	top
cute	bra	beautiful	sandals	feel
comfortable	small	gift	walk	soft
fits	color	earrings	toe	black
ring	fits	compliments	pairs	jeans
dress	perfect	chain	hold	pants
shirt	material	jewelry	strap	tight
material	long	shoes	pockets	material

Table 5: Interpretation for why the "user 1511" rated "item 1" and "item 2" with 5 and 2, respectively, from *Clothing*.

Aspects	Value	Comfort	Accessories	Shoes	Clothing
Importance (1)	0.621	0.042	0.241	0.001	0.095
Matching (1)	0.982	0.596	0.660	0.759	0.638
Polarity (1)	+	-	+	-	+
Importance (2)	0.621	0.042	0.241	0.001	0.094
Matching (2)	0.920	0.303	0.362	1.000	0.638
Polarity (2)	-	-	-	-	+

better prediction. As we can see, our ALFM model substantially improves the prediction accuracy compared with the BMF model. More importantly, our model also outperforms all the other baselines which also utilize reviews. This demonstrates that our model is more effective in exploiting reviews and ratings, because it learns user's preferences and item's features in different aspects and is capable of estimating the aspect weights based on the targeted user's preferences and targeted item's features.

4.6 Interpretability (RQ4)

In our ALFM model, a user's preference on an item is decomposed into user's preference on different aspects and the importance of those aspects. An aspect is represented as a distribution of latent topics discovered based on reviews. A user's attitude/sentiment on an aspect of the targeted item is decided by the latent factors (learned from ratings) associating with the aspect. Based on the topic distribution of an aspect ($\boldsymbol{\theta_{u,a_s}}$) and the word distribution of topics ($\boldsymbol{\phi_w}$), we can semantically represent an aspect by the top words in this aspect. Specifically, the probability of a word w in an aspect a_s of a user u can be computed as $\sum_{k=1}^{K} \theta_{u,a_s,k}\phi_{k,w}$. The top 10 aspect words (#aspect = 5) of "user 1511" from *Clothing* dataset discovered by our model are shown in Table 4. Notice that in order to obtain a better visualization of each aspect, we removed the "background" words that belong to more than 3 aspects. As shown in Table 4, the five aspects can be semantically interpreted to "value"[7], "comfort", "accessories", "shoes", and "clothing". Next, we illustrate the interpretability of our ALFM model on high or low ratings by examples from the same dataset. Table 5 shows the aspect importance (i.e., $\rho_{u,i,a}$ in Eq. 4) of the "user 1511" , the aspect matching scores (i.e., $s_{u,i,a}$ in Eq. 2) as well as sentiment polarity (obtained by Eq. 5) on the five aspects with respect to "item 1" and "item 2" in *Clothing* dataset. From the results, we can see that "user 1511" pays more attention to "Value" and "Accessories" aspects. On the "Value" aspect, both "item 1" and "item 2" highly match her preference, however, she has a positive sentiment on "item 1" while a negative sentiment on "item 2".[8] For the "Accessories" aspect, "item 1" has a higher matching score than "item 2"; and more importantly, the sentiment is positive on "item 1" while negative on "item 2". As a result, "user 1511" rated "item 1" with 5 while rated "item 2" with 2. From the examples, we can see that our model could provide explanations for the recommendations in depth with *aspect semantics*, *aspect matching score*, as well as *aspect ratings* (which shows sentiment polarity).

5 CONCLUSIONS

In this paper, we proposed an aspect-aware latent factor model for rating prediction by effectively combining reviews and ratings. Our model correlates the latent topics learned from review text and the latent factors learned from ratings based on the same set of aspects, which are discovered from textual reviews. Accordingly, our model does not have the constraint of one-to-one mapping between latent factors and latent topics as previous models (e.g., HFT, RMR, RBLT, etc.), and thus could achieve better user preference and item feature modeling. Besides, our model is able to estimate aspect ratings and assign weights to different aspects. The aspect weight is dependent on each user-item pair, since it is estimated based on user's personal preferences on the corresponding aspect towards an item. Experimental results on 19 real-world datasets show that our model greatly improves the rating prediction accuracy compared to the state-of-the-art methods, especially for users who have few ratings. With the extracted aspects from textual reviews, estimated aspect weights, and aspect ratings, our model could provide interpretation for recommendation results in great detail.

ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centre in Singapore Funding Initiative. The authors would like to thank Rose Catherine Kanjirathinkal (from CMU)'s great help on fine-tuning the results of TransNet on all datasets.

[7] "Value" means value for money

[8] As a reminder, the aspect matching is based on the reviews. It is possible that both item 1 and item 2 contains comments on aspect "value". However, "item 1" has a high value while "item 2" has a low value.

REFERENCES

[1] R. Arun, V. Suresh, CE V. Madhavan, and MN N. Murthy. 2010. On finding the natural number of topics with latent dirichlet allocation: Some observations. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 391–402.

[2] Y. Bao, H. Fang, and J. Zhang. 2014. TopicMF: Simultaneously exploiting ratings and reviews for recommendation. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 2–8.

[3] R. M Bell and Y. Koren. 2007. Lessons from the Netflix prize challenge. *ACM SIGKDD Explorations Newsletter* 9, 2 (2007), 75–79.

[4] David M Blei. 2012. Probabilistic topic models. *Commun. ACM* 55, 4 (2012), 77–84.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3 (2003), 993–1022.

[6] R. Catherine and W. Cohen. 2017. TransNets: Learning to Transform for Recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems*. ACM, 288–296.

[7] Z. Cheng and J. Shen. 2016. On effective location-aware music recommendation. *ACM Trans. Inf. Syst.* 34, 2 (2016), 13:1–13:32.

[8] Z. Cheng, J. Shen, L. Nie, T.-S. Chua, and M. Kankanhalli. 2017. Exploring user-specific information in music retrieval. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 655–664.

[9] Z. Cheng, J. Shen, L. Zhu, M. Kankanhalli, and L. Nie. 2017. Exploiting music play sequence for music recommendation. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 3654–3660.

[10] E. Christakopoulou and G. Karypis. 2016. Local item-item models for top-n recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 67–74.

[11] P. Covington, J. Adams, and E. Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 191–198.

[12] Q. Diao, M. Qiu, C.-Y. Wu, A. J Smola, J. Jiang, and C. Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 193–202.

[13] D. M. Endres and J. E Schindelin. 2003. A new metric for probability distributions. *IEEE Trans. Inf. Theory* 49, 7 (2003), 1858–1860.

[14] G. Ganu, N. Elhadad, and A. Marian. 2009. Beyond the Stars: Improving Rating Predictions using Review Text Content. In *Proceedings of the 12th International Workshop on the Web and Databases*, Vol. 9. Citeseer, 1–6.

[15] T. L Griffiths and M. Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences* 101, Suppl 1 (2004), 5228–5235.

[16] R. He and J. McAuley. 2016. VBPR: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. 144–150.

[17] X. He, T. Chen, M.-Y. Kan, and X. Chen. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1661–1670.

[18] X. He and T.-S. Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 355–364.

[19] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[20] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 549–558.

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

[22] F. Li, S. Wang, S. Liu, and M. Zhang. 2014. SUIT: A supervised user-item based topic model for sentiment analysis. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 1636–1642.

[23] G. Ling, M. Lyu, and I. King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 105–112.

[24] H. Ma, D. Zhou, C. Liu, M. Lyu, and I. King. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 287–296.

[25] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. 2010. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research* 11, Jan (2010), 19–60.

[26] J. McAuley and J. Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 165–172.

[27] N. Pappas and A. Popescu-Belis. 2013. Sentiment analysis of user comments for one-class collaborative filtering over ted talks. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 773–776.

[28] Š. Pero and T. Horváth. 2013. Opinion-driven matrix factorization for rating prediction. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 1–13.

[29] L. Qiu, S. Gao, W. Cheng, and J. Guo. 2016. Aspect-based latent factor model by integrating ratings and reviews for recommender system. *Knowledge-Based Systems* 110 (2016), 233–243.

[30] Y. Shi, M. Larson, and A. Hanjalic. 2013. Mining contextual movie similarity with matrix factorization for context-aware recommendation. *ACM Trans. Intell. Syst. Technol.* 4, 1 (2013), 16.

[31] Y. Tan, M. Zhang, Y. Liu, and S. Ma. 2016. Rating-boosted latent topics: Understanding users and items with ratings and reviews. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 2640–2646.

[32] C. Wang and D. 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.

[33] H. Wang, Y. Lu, and C. Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 618–626.

[34] X. Wang, X. He, L. Nie, and T.-S. Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 185–194.

[35] X. Wang, X. He, L. Nie, and T.-S. Chua. 2018. TEM: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 27th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee.

[36] Y. Wu and M. Ester. 2015. FLAME: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, 199–208.

[37] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T.-S. Chua. 2016. Discrete collaborative filtering. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 325–334.

[38] H. Zhang, Z.-J. Zha, Y. Yang, S. Yan, Y. Gao, and T.-S. Chua. 2014. Attribute-augmented semantic hierarchy: towards a unified framework for content-based image retrieval. *ACM Transactions on Multimedia Computing, Communications, and Applications* 11, 1s (2014), 21:1–21:21.

[39] W. Zhang and J. Wang. 2016. Integrating topic and latent factors for scalable personalized review-based rating prediction. *IEEE Trans. Knowledge Data Eng.* 28, 11 (2016), 3013–3027.

[40] W. Zhang, Q. Yuan, J. Han, and J. Wang. 2016. Collaborative multi-level embedding learning from reviews for rating prediction. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 2986–2992.

[41] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1449–1458.

[42] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma. 2015. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 1661–1670.

[43] L. Zheng, V. Noroozi, and P. S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 425–434.