

Review Polarity-wise Recommender

Han Liu, Yangyang Guo, Jianhua Yin, *Member, IEEE*, Zan Gao, and Liqiang Nie, *Senior Member, IEEE*

Abstract—The *de facto* review-involved recommender systems, utilizing review information to enhance recommendation, have received increasing interest over the past years. Thereinto, one advanced branch is to extract salient aspects from textual reviews (i.e., the item attributes that users express) and combine them with the matrix factorization technique. However, existing approaches all ignore the fact that semantically different reviews often include opposite aspect information. In particular, positive reviews usually express aspects that users prefer, while the negative ones describe aspects that users dislike. As a result, it may mislead the recommender systems into making incorrect decisions pertaining to user preference modeling. Towards this end, in this paper, we present a Review Polarity-wise Recommender model, dubbed as RPR, to discriminately treat reviews with different polarities. To be specific, in this model, positive and negative reviews are separately gathered and utilized to model the user-preferred and user-rejected aspects, respectively. Besides, in order to overcome the imbalance of semantically different reviews, we further develop an aspect-aware importance weighting strategy to align the aspect importance for these two kinds of reviews. Extensive experiments conducted on eight benchmark datasets have demonstrated the superiority of our model as compared to several state-of-the-art review-involved baselines. Moreover, our method can provide certain explanations to the real-world rating prediction scenarios.

Index Terms—Review-involved Recommendation, Aspect-aware Recommendation, Review Polarity, Review Imbalance Problem.

I. INTRODUCTION

NOWADAYS, posting reviews on e-commerce platforms has become ubiquitous among online shoppers to share their purchasing experiences. These textual reviews usually contain rich semantic information about user preferences and item attributes, thereby playing an increasingly important role in recommender systems [1]. One typical benefit is that reviews enable the machine to effectively exploit more side information, and receive superior performance as compared with canonical matrix factorization-based methods [2], as the latter methods utilize only the sparse rating matrix.

Previous studies on review-involved recommendation mostly adopt a standard scheme: the user and item documents are firstly constructed by merging the associated reviews (i.e., reviews of the user and reviews for the item), wherein each textual token is vectorized via word embedding methods [3], [4]. The two types of documents are respectively processed via convolutional neural networks to generate the user and item representations, followed by a matching function (e.g.,

Han Liu, Yangyang Guo, Jianhua Yin, and Liqiang Nie are with School of Computer Science and Technology, Shandong University, Qingdao 266200, China. E-mail: hanliu.sdu@gmail.com, guoyang.eric@gmail.com, jhyin@sdu.edu.cn, nieliqiang@gmail.com.

Zan Gao is with Shandong Artificial Intelligence Institute, China. E-mail: zangaonsh4522@gmail.com.

Liqiang Nie is the corresponding author.

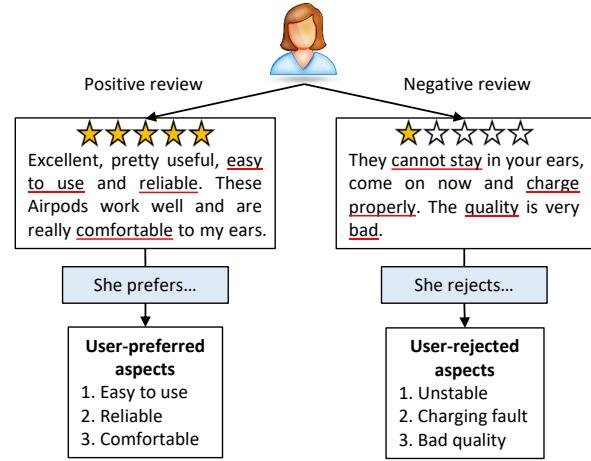


Fig. 1. An example of the opposite aspect information expressed in semantically different reviews.

dot product and Factorization Machines [5]) to predict the final rating score. Based on this scheme, methods like DeepCoNN [6], TransNets [7], D-Attn [8], and MPCN [9] have achieved some improvements over other baselines. Distinct from these approaches, recent efforts have been dedicated to the review aspect modeling [10]–[13]. Foremost, the aspect is defined as follows:

- **Aspect** - It is embodied with high-level semantics, representing the attributes of items that users comment on in their reviews [12]. For example, in the review “Excellent, pretty useful, *easy to use* and *reliable*. These Airpods work well and are *comfortable* to my ears”, the user mentioned the aspects *easy to use*, *reliable*, and *comfortable* of item *Airpods* (as shown in Fig. 1).

In general, these high-level aspect features are firstly extracted through well-developed tools, such as topic modeling [12], which are then integrated with the matrix factorization backbones [2].

Despite their notable progress, one issue hurts the performance of the existing review-involved methods is that the review polarities are not explicitly discriminated, i.e., all reviews are taken as positive feedback. In fact, users tend to convey their sentiments in reviews, i.e., higher rating scores often go with positive reviews, while negative ones meet lower scores frequently [14]–[16]. Moreover, reviews with different polarities usually contain opposite aspect information. Fig. 1 shows two opposite polarities of reviews for *bluetooth headsets*. The left one implies a positive review, describing aspects of an item that the user prefers, e.g., *easy to use*. On the contrary, the right one is a negative review that reveals unsatisfactory aspects of an item the user dislikes, e.g., *unstable*. As illustrated in Fig. 2, negative reviews are quite common in existing

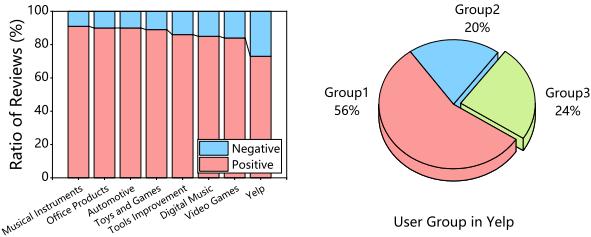


Fig. 2. Imbalance illustration of the two-polarity reviews. The left subfigure shows the ratios of the two kinds of reviews in eight datasets. The right one shows various groups of users in Yelp. Group1 and Group2 respectively denote the numbers of users with positive and negative review ratios over 90%, and Group3 implies the remaining users.

datasets^{1,2}. Simply integrating these two kinds of reviews as positive will mislead the models to recommend items similar to the ones with negative reviews in the future. It thus results in sub-optimal performance and deteriorates the user experiences and faithfulness to the platform. In the light of this, towards making the recommendation more personalized and convincing, we aim to distinguish the user-preferred aspects from the user-rejected ones via explicitly performing polarity discrimination. Nevertheless, discriminately treating positive and negative reviews is non-trivial in recommendation, due to the following facts: 1) It is difficult to effectively exploit the semantic information of each review word for aspect modeling. 2) How to accurately model the user preferences on both user-preferred and user-rejected aspects poses another challenge for us. And 3) there exists severe imbalance regarding different review polarities in current e-commerce datasets, as shown in Fig. 2. For instance, some users tend to post much more positive reviews than negative ones. It therefore exerts adverse impact on the extraction of aspect information, degrading the recommendation performance.

In order to tackle the aforementioned issues, we present a Review Polarity-wise Recommender model, RPR for short, as shown in Fig. 3, to perceive the review polarity towards review-involved recommendation. In particular, for user u and item i , we first leverage their latent factor embeddings to estimate two relevance score vectors for both the user-preferred and user-rejected aspects, whereby each element expresses the preference degree on the corresponding aspect³. Meanwhile, traditional topic modeling has shown certain limitations in leveraging the abundant semantic information. We thus turn to a TextCNN in parallel on the review sentences. This module is expected to extract aspect importance as well as provide some explicit interpretations to the aspect modeling. Finally, the overall rating $r_{u,i}$ is estimated by subtracting the inner product of the score and importance vectors on the user-rejected aspects from the ones on the user-preferred aspects. Based on this, the positive and negative user preferences are seamlessly integrated to implement our RPR.

Besides, to overcome the imbalance between positive and

negative reviews, we introduce an aspect-aware importance weighting module to capture the mapping relationship between the user-preferred and user-rejected aspect importance. The assumption is that if users focus on certain aspects they prefer, they will consistently pay roughly equal attention to the corresponding user-rejected ones, and vice versa. In view of this, according to the correspondence between user-preferred and user-rejected aspects, RPR constructs a user-rejected aspect importance offset from the user-preferred aspect importance, which is further added to the original user-rejected one. In this way, the two types of reviews mutually enhance each other for obtaining the aspect importance.

Overall, the main contributions of this paper are summarized in three-fold:

- We propose a novel recommendation method to extract the semantic information of user-preferred and user-rejected aspects from positive and negative reviews, respectively. To the best of our knowledge, this work is among the first efforts to treat reviews with different polarities discriminately in review-involved recommendation.
- We devise an aspect-aware importance weighting component to construct the semantic mappings between user-preferred and user-rejected aspects. This design has been proven to be quite effective in solving the problem of data imbalance between positive and negative reviews.
- We conduct comprehensive experiments on eight benchmark datasets to evaluate the effectiveness of the proposed model. Extensive results demonstrate the state-of-the-art performance of RPR. As a side contribution, we have released the codes, data, and parameters to facilitate researchers in this field⁴.

The rest of this paper is organized as follows. Section 2 briefly reviews representative literature from two directions that are highly relevant to our work. Section 3 outlines RPR and its architecture, and describes how to optimize RPR. In Section 4 and 5, we experimentally evaluate RPR and analyze the evaluation results, respectively. We summarize the contributions and figure out the potential future research directions in Section 6.

II. RELATED WORK

A. Review-involved Recommendation

Recently, exploiting reviews to enhance the recommendation performance and interpretability has been extensively studied in literature [8], [17]. Along this line, existing methods can be broadly classified into two categories. The first category mainly focuses on the user and item modeling from their separately corresponding documents. To be more specific, the user and item documents are firstly constructed by concatenating the user-posted and item-received reviews, respectively. And the word embedding techniques are then adopted to embed the textual document into a semantic matrix. Normally, a Convolutional Neural Network (CNN) [18] is utilized to extract the user and item representations from the matrices. Thereafter, a matching function (e.g., dot product, Factorization Machines)

¹<http://jmcauley.ucsd.edu/data/amazon/>.

²<https://www.yelp.com/dataset/challenge>.

³In RPR model, the aspects are implicit, and the number of aspects are fixed for all users.

⁴<https://github.com/hanliu95/RPR>.

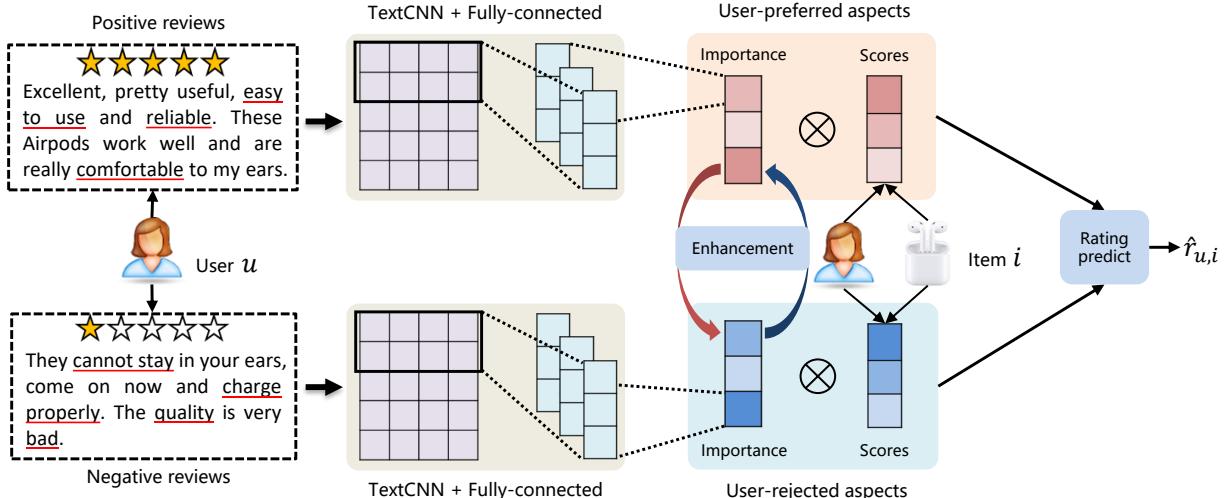


Fig. 3. The architecture of our proposed RPR model. From left to right, 1) two parallel convolutional networks are adopted to extract the importance of user-preferred and user-rejected aspects for user u from u 's positive and negative reviews, respectively. Moreover, to overcome the imbalance problem between positive and negative reviews, an aspect-aware importance weighting module is appended to align the importance between the two polarities of reviews. 2) The user-preferred and user-rejected aspect scores of user u towards item i are separately learned. 3) The inner product of the score and importance vectors on the user-preferred aspects is calculated as the positive score of u towards i , and then the negative score is similarly computed on the user-rejected aspects. RPR finally employs the subtraction of the negative score from the positive one to estimate the final rating.

can be employed to model the user-item interactions. Methods like DeepCoNN [6], TransNets [7], D-Attn [8], and MPCN [9] all adopt the above scheme, obtaining superior performance compared with the prior matrix factorization ones.

The second category aims to effectively learn the aspects from reviews for recommendation, namely, the aspect-aware recommender systems [13], [19], [20]. These methods can be further summarized into the following two groups. The first group tries to extract aspects based on the existing NLP tools for sentiment analysis [20]–[22]. The obtained aspect representation is then incorporated into a matrix factorization framework for more accurate recommendation. For example, EFM [20] and MTER [11] resort to a phrase-level NLP tool for the aspect-level sentiment extraction. The second group devises specific internal components, e.g., topic modeling, to automatically learn explainable aspect representations of users and items from reviews [12], [13], [23]. In particular, these internal components are leveraged to achieve the aspect-aware extraction of the semantic information in reviews. In a nutshell, compared with the first category, the aspect-aware methods are capable of extracting high-level semantic features from reviews, delivering improved results as well as interpretability.

B. Deep Learning in Recommendation

A prevailing trend in recent years has been leveraging deep learning for recommendation. It is now extensively recognized that deep learning techniques are capable of modeling the complex and non-linear user-item interactions [24], [25]. Generally speaking, the success of deep learning in recommendation mainly comes from two aspects: representation learning and matching function modeling [26]–[28]. Regarding representation learning, the entity embeddings, i.e., users and items, can be greatly enhanced with advanced tools in deep learning. For example, CNNs are used to enrich the item representation

learning from both texts and images [6], and RNNs show considerable advantage in session-based recommendation [29]. For the other aspect of matching function modeling, deep learning methods often utilize multi-level neural networks as the interaction function to effectively aggregate the low-level signals. Among these methods, Multi-Layer Perceptron (MLP) is distinguished with its strong capability to learn both the second-order and higher-order feature interactions [30].

Notably, the attention mechanism [31]–[33] has been extensively integrated into recommendation, which demonstrates promising results and great potentials in existing studies [34], [35]. For example, ACF [36] introduces a hybrid item-and component-level attention model. Meanwhile, NAIS [37] presents a neural attentive item similarity model for item-based collaborative filtering, enabling itself to identify the more important historical items in a user profile for rating prediction. In addition, AFM [38] learns the weights of feature interactions in factorization machines via the neural attention networks. A³NCF [39] introduces a topic model-based attention method, where the attention module is used to capture the attentive user preferences on each aspect of the target item.

III. PROPOSED METHOD

A. Preliminaries

In this subsection, we first briefly present the general framework for the aspect-aware model which exploits reviews to predict user-item interaction ratings, and point out its limitation caused by ignoring the review polarity. We then recapitulate how our RPR model can overcome the issue step-by-step.

Aspect-aware recommender systems assume that a user-item rating $r_{u,i}$ depends on user u 's score towards item i on each aspect a (i.e., aspect score $s_{u,i,a}$) and the importance of each aspect to u (i.e., aspect importance $\rho_{u,a}$). It is worth noting

that the aspects are implicitly defined throughout this paper following [12], [23]. In general, the overall rating $r_{u,i}$ can be predicted by,

$$\hat{r}_{u,i} = \rho_u^\top \mathbf{s}_{u,i}, \quad (1)$$

where the aspect importance $\rho_u \in \mathbb{R}^{|\mathcal{A}|}$ is estimated based upon user reviews (\mathcal{A} denotes the set of aspects, e.g., *{easy to use, reliable, comfortable}* for *Airpods*), and the aspect score $\mathbf{s}_{u,i} \in \mathbb{R}^{|\mathcal{A}|}$ is computed through matrix factorization relying on user-item interactions. However, the expression capability of these models is largely limited, since they treat all aspects as user-preferred and ignore the fact that reviews can contain negative opinions. This problem may lead to sub-optimal model performance, degrading the faithfulness of these methods.

To overcome the above issue, in this paper, we aim to predict $r_{u,i}$ via differently handling the two opposite polarities of reviews. One straightforward solution is to divide the aspects into user-preferred and user-rejected aspects from positive and negative reviews, respectively. The objective is then formulated as follows:

$$\hat{r}_{u,i} = \rho_u^p \top \mathbf{s}_{u,i}^p - \rho_u^r \top \mathbf{s}_{u,i}^r, \quad (2)$$

where the subtraction of the two scores considers both user-preferred and user-rejected aspects during the rating prediction of u towards i . Vector $\mathbf{s}_{u,i}^p$ ($\mathbf{s}_{u,i}^r$) consists of the estimated scores of u towards i on the user-preferred (user-rejected) aspects. ρ_u^p (ρ_u^r) denotes the importance vector of the user-preferred (user-rejected) aspects for u , extracted from u 's positive (negative) reviews.

Moreover, in order to tackle the imbalance between positive and negative reviews, we intuitively assume that there exists a latent correlation of importance between the two kinds of aspects. We model the correlation to generate the aspect importance offsets μ_u^p and μ_u^r to enhance ρ_u^p and ρ_u^r , respectively. Ultimately, the predictive model of RPR is given as:

$$\hat{r}_{u,i} = (\rho_u^p + \mu_u^p) \top \mathbf{s}_{u,i}^p - (\rho_u^r + \mu_u^r) \top \mathbf{s}_{u,i}^r. \quad (3)$$

In the following, we will elaborate the proposed RPR model in three-fold: aspect score estimation, aspect importance extraction, and aspect importance offset learning.

B. Aspect Score Estimation

Following mainstream recommender models [30], we map users and items into a latent factor space and represent user u and item i by latent factor vectors $\mathbf{p}_u \in \mathbb{R}^f$ and $\mathbf{q}_i \in \mathbb{R}^f$, respectively. According to [40], the interaction between u and i on each latent factor is characterized by $\mathbf{p}_u \odot \mathbf{q}_i$, where \odot represents the element-wise product between two vectors. Similar to [12], to leverage the latent factor-level interactions for aspect score prediction, we introduce two indicator matrices $\mathbf{M} \in \mathbb{R}^{f \times |\mathcal{P}|}$ and $\mathbf{V} \in \mathbb{R}^{f \times |\mathcal{R}|}$ (where $|\mathcal{P}|$ and $|\mathcal{R}|$ are the numbers of user-preferred and user-rejected aspects, respectively), to associate the latent factors to different user-preferred and user-rejected aspects, respectively. The weight vector \mathbf{m}_x , which is the x -th column of \mathbf{M} , indicates which latent factor-level interactions are related to the score of the x -th user-preferred aspect. Similarly, the weight vector \mathbf{v}_y

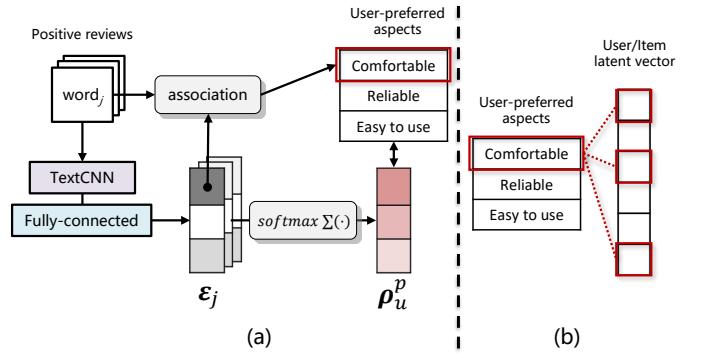


Fig. 4. Aspect importance modeling based upon the review words and correlation between aspects and latent vectors. (a) Aspect importance modeling. Each review word is embedded with TextCNN and Fully-connected layers to produce semantic embedding ϵ . The user-preferred aspect importance ρ_u^p is then estimated via the summation of all the word semantic embeddings followed by a $softmax$ function. We then associate the largest element $\epsilon_{j,1}$ of ϵ_j to its corresponding aspect, i.e., the first aspect is expressed with the j -th word *comfortable*. And (b) an exemplar correlation between aspect and latent vectors. The three elements circled with red boxes in the latent vector are related to the first aspect *comfortable*.

indicates which interactions are related to the score of the y -th user-rejected aspect. Towards this end, we estimate the scores of the user-preferred and user-rejected aspects via the following formula,

$$\begin{cases} \mathbf{s}_{u,i}^p = \mathbf{M}^\top (\mathbf{p}_u \odot \mathbf{q}_i) \\ \mathbf{s}_{u,i}^r = \mathbf{V}^\top (\mathbf{p}_u \odot \mathbf{q}_i) \end{cases}, \quad (4)$$

where $\mathbf{s}_{u,i}^p$ and $\mathbf{s}_{u,i}^r$ represent user's preference scores on the user-preferred and user-rejected aspects of items, respectively.

C. Aspect Importance Extraction

We define that the user u 's positive document D_u^{pos} is constructed by collecting all the positive reviews posted by u , and the user's negative document D_u^{neg} is built in a similar manner. It is widely accepted that users tend to comment on aspects with opposite attitudes pertaining to different polarities of reviews. In addition, users would individually care more about certain aspects than others. For example, fashion enthusiasts often focus on the user-preferred aspect *"fashionable style"* of *"clothing"* items, as the review words like *"fashion sense"* frequently appear in their reviews. In the following, we mainly detail the formulation on how to extract the user-preferred aspect importance by word-wise extraction of D_u^{pos} , while the user-rejected one can be obtained in a similar way.

Firstly, we use the pre-trained word embeddings to initialize the word representations in the positive document, where $\mathbf{e}_j \in \mathbb{R}^d$ is the embedding vector for the j -th word, and d denotes the embedding dimension. We then adopt a CNN model to extract the contextual information of each word [41], [42]. The convolution layer can be regarded as a tensor \mathbf{K} , which consists of N neurons. Each neuron uses a filter with size $d \times c$ spanning $\epsilon = \frac{c-1}{2}$ words on both sides of each word. The latent contextual feature vector for the j -th word is formulated as follows,

$$\mathbf{c}_j = ReLU(\mathbf{W}_c([\mathbf{e}_{j-\epsilon} \dots \mathbf{e}_j \dots \mathbf{e}_{j+\epsilon}] * \mathbf{K}) + \mathbf{b}_c), \quad (5)$$

where $\mathbf{c}_j \in \mathbb{R}^N$ is the latent contextual feature vector for the j -th word, $\mathbf{W}_c \in \mathbb{R}^{N \times N}$ and $\mathbf{b}_c \in \mathbb{R}^N$ denote the weight matrix and bias vector, respectively.

Secondly, we focus on how to extract the user-preferred aspect importance based on the word contextual feature vectors. In fact, some salient words are supposed to contribute more to the aspect importance modeling of users. For example, the word “cost-effective” in reviews implies that the user probably puts more emphasis on aspects like “good price” and “effectiveness”. In addition, it is natural that if an aspect-specific word is frequently mentioned in D_u^{pos} , the user u will attach more importance to the aspects related to this word. For example, “delicious” and “yummy” would be repeatedly written by a user who pays much attention to the “good taste” aspect. Inspired by this, we develop a fine-grained semantic extraction network for the aspects. Specifically, we resort to the fully-connected layers to automatically discriminate the importance contribution of each review word,

$$\begin{cases} \varepsilon_j = \text{ReLU}(\mathbf{W}_\rho \mathbf{c}_j + \mathbf{b}_\rho) \\ \rho_u^p = \text{softmax}(\sum_{j=1}^{l_{pos}} \varepsilon_j), \end{cases} \quad (6)$$

where the semantic embedding $\varepsilon_j \in \mathbb{R}^{|\mathcal{P}|}$ is projected from the j -th word contextual features through a matrix $\mathbf{W}_\rho \in \mathbb{R}^{|\mathcal{P}| \times N}$, bias $\mathbf{b}_\rho \in \mathbb{R}^{|\mathcal{P}|}$, and the ReLU activation function. l_{pos} represents the number of words in the user’s positive document, and $\rho_u^p \in \mathbb{R}^{|\mathcal{P}|}$ is the user-preferred aspect importance, reflecting the emphasis degree on each user-preferred aspect attached by user u .

From the above formulation, it can be seen that each element in the word semantic embedding contributes distinctively to the aspect importance modeling. In the light of this, we adopt a loose hypothesis that each aspect associates closely with certain review words. We hence assume that one word can express one aspect (i.e., one element in the aspect vector) if the corresponding element from the word semantic embedding is the largest, since the dimensions of word semantic embedding and aspects are the same⁵. For example, as shown in Fig. 4, the j -th word, whose first element $\varepsilon_{j,1}$ is the largest in the semantic embedding ε_j , can be associated with the first aspect.

Based on the same procedure, we can also extract the user’s importance vector for user-rejected aspects ρ_u^r with a similar module from her/his negative document D_u^{neg} .

D. Aspect Importance Offset Learning

Though we have obtained the expected scores and the corresponding importance distribution for the user-preferred and user-rejected aspects, it is still insufficient to correctly predict the overall rating. The reason is that in a practical scenario, it is common that the objective user u provides much more reviews from one polarity than the other, referring to Fig. 2. For a better understanding, we make an extreme assumption that if user u only posts positive reviews, the learned user-rejected aspect importance vector ρ_u^r would approximately

⁵We can also use k words to express an aspect with sorting the corresponding element values.

approach zero in RPR. Furthermore, if we leverage the user-rejected aspect importance vector to predict the ratings toward the items in aspects the user distastes, the predicted rating would be unfavorable.

In order to solve the imbalance in reviews with different polarities, we propose to construct the relations between the user-preferred and user-rejected aspect importance. In particular, we intuitively assume that if a user attaches more importance to a user-preferred aspect, she/he will correspondingly pose roughly equal importance to the related user-rejected aspects. For example, a user pays attention to the user-preferred aspect of “elegant environment” and has chosen a restaurant, she/he would similarly care about the user-rejected aspects like “obsolete decoration” and “poor sanitary situation”. Based on this assumption, we leverage an aspect-aware importance weighting module to construct the mapping relationship between the user-preferred and user-rejected aspect importance.

In order to construct the relationships between two opposite polarities of aspects, we utilize the common *user/item latent space* as a bridge. Given that the aspect is associated with specific latent factors as shown in Eqn.(4), the related aspects from the other polarity are expected to be more similar in the latent space. As weight vectors \mathbf{m}_x and \mathbf{v}_y are responsible for the latent aspect modeling (as mentioned in Section 3.2), RPR takes these indicator vectors as inputs to the aspect-aware importance weighting module for measuring the similarity among aspects. Specifically, the following function is employed to obtain the attention weight of the user-rejected aspect y to each user-preferred aspect,

$$\begin{cases} \phi'_{y,x} = \mathbf{h}_a^\top \text{ReLU}(\mathbf{W}_a(\mathbf{v}_y \odot \mathbf{m}_x) + \mathbf{b}_a) \\ \phi_{y,x} = \frac{\exp(\phi'_{y,x})}{\sum_{x \in \mathcal{P}} \exp(\phi'_{y,x})} \end{cases}, \quad (7)$$

where $\phi_{y,x}$ denotes the attention weight of the user-rejected aspect y to the x -th user-preferred aspect, and \mathbf{h}_a , \mathbf{W}_a , and \mathbf{b}_a are the parameters of the aspect attention network. For simplicity, we concatenate these attention weights into a matrix $\Phi \in \mathbb{R}^{|\mathcal{P}| \times |\mathcal{R}|}$. The y -th column ϕ_y of matrix Φ denotes the attention weight vector of the user-rejected aspect y , whose x -th element is $\phi_{y,x}$.

We take the attention weights as the mapping relationships between the user-preferred and user-rejected aspect importance. As for the aforementioned imbalance problem, we can employ an offset vector as an addition to the user-rejected aspect importance vector by utilizing the matrix Φ . Thus, the enhanced user-rejected aspect importance vector of user u equals,

$$\begin{cases} \mu_u^r = \Phi^\top \rho_u^p \\ \rho_u^{r+} = \rho_u^r + \mu_u^r \end{cases}, \quad (8)$$

where μ_u^r denotes the offset for user-rejected aspect importance, and ρ_u^p is the user-preferred aspect importance vector that has already been extracted in Eqn.(6). The y -th element of the offset vector μ_u^r equals the inner product $\phi_y^\top \rho_u^p$, and the attention weights of the y -th user-rejected aspect on each user-preferred aspect model a linear relationship, which maps the extracted user-preferred aspect importance to the objective user-rejected aspect importance space. In this way, we can

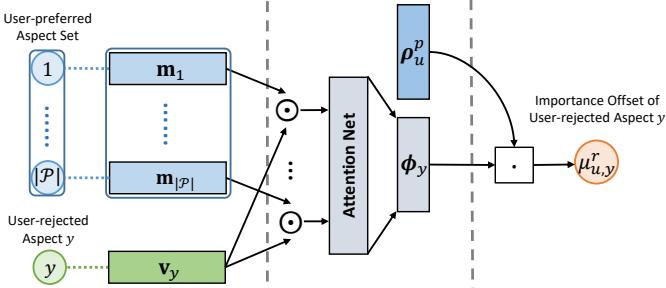


Fig. 5. The generation process of importance offset for the y -th user-rejected aspect. The similarity between v_y and each column of M is computed via the attention net, which is then combined with the initial user-preferred aspect importance vector ρ_u^p .

achieve a more intuitive user-rejected aspect importance vector of a user from her/his positive reviews indirectly, even if the negative reviews are inadequate. Fig. 5 illustrates the generation process of importance offset for the y -th user-rejected aspect. Similarly, we can obtain the enhanced user-preferred aspect importance ρ_u^{p+} , where the process is omitted due to space limitation. Extensive experiments have demonstrated that the offset components can effectively resolve the imbalance problem between the positive and negative reviews, which will be elaborated in Section 5.2.

E. Overall Objective

Up to now, we have obtained the scores and importance of the user-preferred and user-rejected aspects, respectively. The expected rating $\hat{r}_{u,i}$ that user u would give to item i is computed as follows,

$$\hat{r}_{u,i} = \rho_u^{p+T} s_{u,i}^p - \rho_u^{r+T} s_{u,i}^r. \quad (9)$$

In this way, both the user positive and negative preferences can be simultaneously considered and discriminated. The estimation of model parameters is to minimize the rating prediction error on the training dataset. In this way, the objective function is,

$$\begin{aligned} \min_{\mathbf{p}_u, \mathbf{q}_i, \mathbf{M}, \mathbf{V}, \Theta} & \frac{1}{2} \sum_{u,i} (r_{u,i} - \hat{r}_{u,i})^2 + \beta_1 (\|\mathbf{M}\|_1 + \|\mathbf{V}\|_1) \\ & + \frac{\beta_2}{2} \left(\|\mathbf{p}_u\|_2^2 + \|\mathbf{q}_i\|_2^2 + \sum_{\theta \in \Theta} \|\theta\|_2^2 \right). \end{aligned} \quad (10)$$

With the minimization of this objective function, all model parameters can be effectively updated through the gradient decent strategy. The involved parameters include user and item latent vectors \mathbf{p}_u and \mathbf{q}_i , indicator matrices \mathbf{M} and \mathbf{V} from the aspect score estimation module, and the remaining parameters Θ from the aspect importance extraction and the aspect importance offset learning modules. $\|\cdot\|_1$ and $\|\cdot\|_2$ respectively denote the l_1 and l_2 regularization norm, with the corresponding hyper-parameters β_1 and β_2 . Considering the fact that l_1 regularization yields sparse solution of the weights, we thus use l_1 norm to obtain approximately binary matrices \mathbf{M} and \mathbf{V} for better indication ability. The l_2 regularization of \mathbf{p}_u , \mathbf{q}_i , and θ prevents these parameters from uncontrollable values and over-fitting.

Optimization. We leverage the Adam optimization algorithm [43] to learn all the parameters by minimizing the objective function in Eqn.(10). Note that for training stability, the parameter is optimized in the following sequence: the user matrix, the item matrix, the weight matrices, and the remaining parameters.

IV. EXPERIMENTAL SETUP

In this section, we first presented the evaluation datasets, and then introduced our experimental settings. Finally, we listed several baseline methods for comparison.

A. Datasets

We conducted experiments on two publicly available datasets: Amazon¹ and Yelp². The Amazon dataset provides rich review information with rating scores. In our experiments, we applied its seven sub-datasets: *Musical Instruments*, *Office Products*, *Digital Music*, *Tools Improvement*, *Automotive*, *Toys and Games*, and *Video Games*. Yelp is a famous online review platform for business, such as restaurants, bars, and spas. We selected the dataset from the latest version and used a 20-core setting to provide a denser dataset. Each record in our datasets consists of user ID, item ID, rating, and the corresponding review text. For all datasets, we filtered out the empty-review records. The target rating score used in these datasets ranges from 1 to 5. Similar to [44], reviews with rating scores higher than or equal to 3 are split into positive documents, while the ones lower than 3 are regarded negative on all datasets. Table I summarizes the detailed statistics of the evaluated datasets, where “pos/neg ratio” denotes relative proportions between positive and negative reviews of each dataset.

TABLE I
STATISTICS OF THE EVALUATED DATASETS.

Datasets	Users	Items	Ratings	pos/neg ratio
Musical Instruments	1,429	900	10,262	10.11
Office Products	4,905	2,420	53,258	9.10
Digital Music	5,541	3,568	64,706	5.67
Tools Improvement	16,638	10,217	134,345	6.14
Automotive	2,928	1,835	20,473	9.20
Toys and Games	19,412	11,924	167,597	8.09
Video Games	24,303	10,672	231,577	5.25
Yelp	40,500	58,755	2,024,283	2.70

For each dataset, we randomly split its records into two parts: 80% for training and the rest 20% for testing. Moreover, 10% records in the training set are randomly selected as the validation set for hyper-parameter tuning. Note that we slightly adjusted the training and testing sets to ensure that at least one record for each user/item would be included in the training set. The target reviews in the validation and testing sets are excluded since they are unavailable in the practical scenario.

B. Experimental Settings

Evaluation Metrics. To thoroughly evaluate our model and the baselines, we adopted the **MSE** (Mean Squared Error) and the **MAE** (Mean Absolute Error) as the evaluation metrics to measure the rating prediction performance.

Implementation Details. We implemented our model via the development tool Tensorflow⁶. The embedding matrix of the document is initialized via word vectors which have been pre-trained in GloVe⁷ (used 50-d vectors for its efficiency). For the convolutional layer in the proposed model, the number and size of filters are set to 50 and 3, respectively. We utilized the popular approach of Xavier [45] to initialize the weights in our model. And we adopted grid search to tune the hyper-parameters based on the results from the validation set. Moreover, we varied the number of both the user-preferred and user-rejected aspects within the set {1, 2, 3, 4, 5}, the dimension of user and item latent factor vectors amongst {4, 8, 16, 32, 64}, the learning rate amongst {1E-05, 1E-04, 1E-03, 1E-02}, and the size of training mini-batch amongst {100, 200, 500, 1,000}.

C. Baseline Comparison

We compared the performance of our proposed method with a series of state-of-the-art recommendation methods. To summarize, we divided the baselines into three categories. The first category is interaction-based, including: **MF**, **FM** [46], **MLP** [30], and **NeuMF** [30]. The second category is plain review-involved, including: **DeepCoNN** [6], **TransNet** [7], and **MPCN** [9]. The last category is aspect-aware, including: **ALFM** [12] and **CARP** [23].

We adopted the publicly available implementations of FM, NeuMF, DeepCoNN, TransNet, MPCN, ALFM, and CARP in our experiments. When training all these baseline models, we set the maximum training epoch to 50 for a fair comparison. For the interaction-based models, we varied the embedding size within {8, 16, 32, 64}. All word embeddings in review-based baselines are initialized using pre-trained word vectors in GloVe [4] or word2vec [3]⁸. For the convolutional layer in the review-based models, the filter size is set to 3, and we tested various numbers of filters amongst {20, 50, 100, 200} to select the optimal one. Dropout is appended after all fully-connected and convolution layers with a dropout rate of 0.2. For TransNet, we used two transform layers, following the model setting adopted in the original paper [7]. For MPCN, the number of pointers is tuned amongst {1, 3, 5, 8, 10}. For ALFM, we tuned the numbers of aspects and latent factors following the original paper [12]. For CARP, we took the suggestion in the original paper [23], and set the capsule number and the predefined threshold value to 5 and 3, respectively.

V. EXPERIMENTAL RESULTS

In order to validate the effectiveness of our proposed method, we conducted extensive quantitative and qualitative experiments to answer the following questions:

- **Q1.** Can our proposed method outperform both the state-of-the-art review-involved and traditional recommendation baselines?
- **Q2.** How do different components (e.g., the aspect-aware importance weighting component for learning importance offsets) contribute to the overall performance of our proposed model?

- **Q3.** How do the key hyper-parameters (e.g., the number of latent factors) affect our model performance?
- **Q4.** Can our model provide explicit interpretations?

A. Q1: Performance Comparison

The results of our method and other baselines over the experimented datasets are presented in Table II. The key observations can be summarized as follows:

- The recommendation methods from the first category obtain the worst performance on all datasets. And the deep learning-based methods (i.e., NeuMF and MLP) can achieve superior performance compared to the factorization-based ones (i.e., FM and MF), since they can model more complex interactions than the factorization ones.
- The review-involved methods consistently surpass the interaction-based ones, demonstrating that reviews contain valuable side information for accurate recommendation. Moreover, amongst the review-involved baselines, it is obvious that MPCN largely outperforms both DeepCoNN (D-CON) and TransNet (T-NET), which mainly benefits from the fact that MPCN adopts the pointer-based scheme to filter important reviews for recommendation. Nevertheless, TransNet outperforms DeepCoNN, since it takes the review of target user-item pair as the approximation object in the training process.
- The aspect-aware methods surpass the plain review-involved ones in most cases. This indicates that capturing aspect-aware information from reviews is effective. Among these baselines, the recently proposed CARP model yields a better result than ALFM. It is because that CARP introduces the capsule network, which is proposed to model the complex relations among features, while ALFM employs a general aspect-aware topic model.
- Finally, we can observe that RPR substantially outperforms all the baselines on the eight datasets. When comparing with the review-involved baselines regarding the MSE metric, its relative improvement is satisfying with gains up to 38.2% (DeepCoNN), 29.6% (TansNet), and 13.9% (MPCN), respectively. Moreover, it is obvious that RPR consistently exceeds the aspect-aware baselines by a considerable margin. Jointly analyzing Fig. 2 and Table II, we observed that the improvements over the best baseline are more significant in datasets *Digital Music*, *Tools Improvement*, *Video Games*, and *Yelp*, where the negative reviews are more sufficient. This observation indicates that discriminately treating reviews with different polarities would promote the recommendation performance.

B. Q2: Ablation Study

We conducted detailed ablation studies to validate how each component contributes to the overall performance of our model. In particular, we compared our proposed model with the following variants:

- **Base:** The base refers to our complete model with the optimal setting.

⁶<http://www.tensorflow.org>.

⁷<https://nlp.stanford.edu/projects/glove/>.

⁸<https://code.google.com/archive/p/word2vec/>.

TABLE II

PERFORMANCE COMPARISON ON EIGHT DATASETS. THE BEST PERFORMANCE IS HIGHLIGHTED IN BOLDFACE. Δ_{CA} DENOTES THE RELATIVE IMPROVEMENT (%) OF RPR OVER THE BEST BASELINE CARP. P-VALUE REFLECTS THE T-TEST RESULT OF RPR COMPARED WITH CARP.

Datasets	Musical Instruments		Office Products		Digital Music		Tools Improvement	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
MF	1.970	1.404	1.144	1.070	1.956	1.399	1.560	1.249
FM	1.167	1.080	1.098	1.048	1.395	1.181	1.320	1.149
MLP	1.082	1.040	1.120	1.058	1.356	1.164	1.351	1.162
NeuMF	1.187	1.089	1.078	1.038	1.334	1.155	1.314	1.146
D-CON	1.286	1.135	0.975	0.825	1.330	1.153	1.248	1.063
T-NET	1.130	0.872	0.951	0.789	1.325	1.052	1.124	0.923
MPCN	0.923	0.860	0.879	0.738	1.291	0.936	1.097	0.912
ALFM	0.891	0.735	0.870	0.758	1.280	0.976	1.065	0.870
CARP	0.879	0.714	0.827	0.686	1.236	0.945	1.069	0.847
RPR	0.795	0.652	0.814	0.641	1.141	0.836	0.987	0.784
Δ_{CA}	9.6	8.7	1.6	6.6	7.7	11.5	7.7	7.4
p-value	9.8E-4	5.6E-4	1.5E-3	4.8E-4	1.5E-3	1.5E-3	9.7E-4	4.5E-4
Datasets	Automotive		Toys and Games		Video Games		Yelp	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
MF	2.080	1.442	1.801	1.342	1.625	1.275	1.738	1.418
FM	1.599	1.265	1.197	1.094	1.656	1.287	1.726	1.414
MLP	1.073	1.036	1.285	1.134	1.576	1.255	1.733	1.416
NeuMF	1.023	1.011	1.210	1.101	1.568	1.252	1.682	1.397
D-CON	1.045	0.931	1.056	0.909	1.307	1.154	1.487	1.342
T-NET	0.963	0.887	1.032	0.782	1.278	1.029	1.466	1.201
MPCN	0.942	0.771	1.036	0.767	1.267	1.026	1.445	1.159
ALFM	0.924	0.765	0.980	0.752	1.245	0.997	1.417	1.073
CARP	0.913	0.712	0.943	0.735	1.238	0.978	1.406	1.079
RPR	0.896	0.703	0.915	0.727	1.142	0.939	1.303	0.986
Δ_{CA}	1.9	1.3	3.0	1.1	7.8	4.0	7.3	8.6
p-value	2.6E-3	3.5E-3	1.1E-3	1.4E-3	1.6E-3	9.5E-4	1.0E-3	7.8E-4

TABLE III

PERFORMANCE COMPARISON OF THE MODEL VARIANTS ON THE *Automotive* AND *Video Games* DATASETS.

Setup	Automotive	Video Games
Base	0.896	1.142
Coarse-grained Model	0.952	1.319
W/o Review Polarity	0.949	1.325
Uniform Aspect Importance	0.964	1.330
W/o Importance Offset	0.947	1.384

- Coarse-grained Model:** Instead of extracting the word-wise aspect importance distribution in Eqn.(5), this variant introduces the max-pooling layer to the convolutional layer in the base.
- W/o Review Polarity:** We removed the components of distinguishing the positive and negative reviews of the user, and followed the previous review-involved methods to collect all the reviews as the user document.
- Uniform Aspect Importance:** We replaced ρ_u^{p+} and ρ_u^{r+} in Eqn.(9) with the uniform importance distributions, i.e., all user-preferred and user-rejected aspects are assumed to be equally important.
- W/o Importance Offset:** We removed the aspect-aware importance weighting component from the complete model. It is worth noting that the imbalance problem is not specifically tackled in this variant.

Table III shows the MSE results of the above variants on the *Automotive* and *Video Games* datasets. First, the word-wise extraction of aspect importance outperforms the document-wise one, which verifies that the fine-grained semantic extraction boosts the performance of review-involved recommendation. Second, we can observe that ignoring the polarities of reviews will deteriorate the performance, it is thus necessary to distinguish the positive and negative reviews for review-involved recommendation. Additionally, it is reasonable for a

user to attach different importance to different aspects of an item. Thus, uniforming the aspect importance would limit the modeling capacity of user preferences. Finally, the result of the last variant reflects that the aspect-aware importance weighting module in our model can effectively reduce the impact of the review imbalance.

C. Q3: Effectiveness of Key Hyper-parameters

In this subsection, we analyzed the effectiveness of the key hyper-parameters in our method for the overall performance. We primarily focused on the number of aspects and latent factors (i.e., the dimension of embeddings \mathbf{p}_u and \mathbf{q}_i). Fig. 6 shows the performance variations with changing the number of aspects and latent factors on three datasets.

Number of Aspects. We set the number of latent factors to 32 for better studying the effect of aspects. With the number of aspects changing from 1 to 5 as illustrated in Fig. 6, we observed that the optimal number of aspects varies with different datasets, which is probably because that users comment different aspects in reviews for different categories of items. Additionally, promising performance can be obtained when the number of user-preferred/user-rejected aspects is from 2 to 4.

Number of Latent Factors. To study the effect of latent factors, we fixed the number of user-preferred/user-rejected aspects to 3. From the Fig. 6, it can be seen that the MSE decreases with increasing the latent factors, since the rating prediction is still based on matrix factorization in our model. Therefore, increasing latent factors can better represent the user/item, contributing to a more accurate rating prediction.

To visualize the joint effects of aspects and factors, we presented 3D figures by varying the number of user-preferred/user-rejected aspects in {1, 2, 3, 4, 5}, and the num-

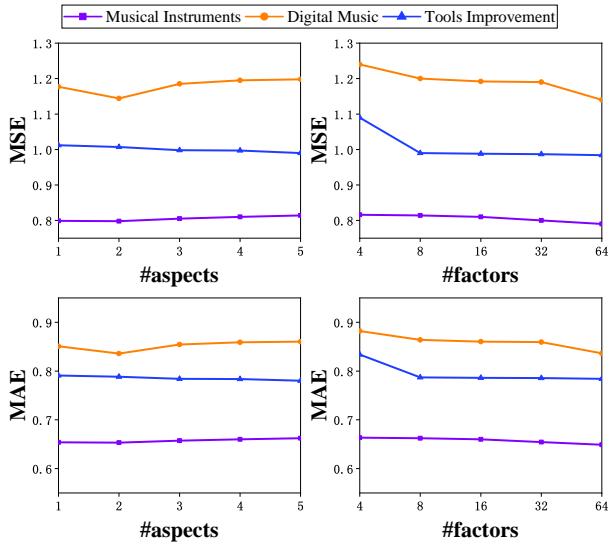


Fig. 6. Effect of the number of aspects and factors in our model.

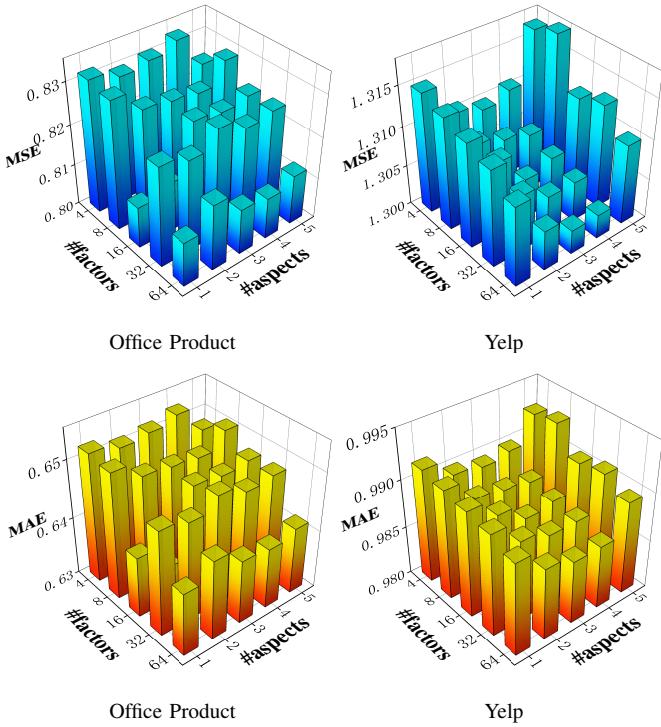


Fig. 7. Confounding effect of the number of aspects and factors.

ber of factors in $\{4, 8, 16, 32, 64\}$ and illustrated the results of *Office Products* and *Yelp* datasets. From Fig. 7, it can be recognized that the optimal numbers of aspects and factors are different across datasets. In general, more latent factors usually lead to better performance, while the optimal number of user-preferred/user-rejected aspects might depend on the review details of different datasets.

D. Q4: Interpretability

In our RPR model, a user's preference on an item depends on the scores and the importance of both user-preferred and user-rejected aspects to the user. The importance on user-preferred/user-rejected aspects is computed by the aspect-specific semantic embeddings in the user's positive/negative

reviews. Since we assume one element corresponds to one aspect, accordingly, review words which hold the largest corresponding elements in their semantic embeddings are adopted as the semantic explanation for this aspect. Table IV records the positive and negative reviews of a randomly selected user from the *Musical Instruments* dataset. In our experiments, we set the number of user-preferred and user-rejected aspects to 2, and filtered their corresponding review words. Table V shows the top 10 aspect words (we removed stop words for better illustration). Based on this experiment, the two user-preferred aspects can be semantically interpreted as "Performance" and "Fine Strings", and the two user-rejected aspects can be interpreted as "Poor Quality" and "Crack Sensitive".

Next, we aimed to study the interpretability of our proposed model on high and low ratings [11], [47]. From the same dataset, we chose "item 1" and "item 2", which are rated with 5 and 1 by the selected user, respectively. We firstly obtained the selected user's aspect importance on the user-preferred and user-rejected aspects (i.e., ρ_u^{p+} and ρ_u^{r+}), and then computed the aspect scores on the user-preferred and user-rejected aspects (i.e., $s_{u,i}^p$ and $s_{u,i}^r$) of "item 1" and "item 2", respectively. As shown in Table VI, we could observe that the selected user pays more attention on the user-preferred aspect "Fine Strings" and the user-rejected aspect "Poor Quality". On the user-preferred aspects, we could see that "item 1" is a better match to the user's preference compared with "item 2", because the user-preferred aspect scores of "item 1" are higher. On the user-rejected aspects, we could observe that the scores of "item 1" are both negative values. This probably means that "item 1" has none of these user-rejected aspects. The scores of "item 2" on user-rejected aspects are positive values, indicating that "item 2" possesses these user-rejected aspects in the user's opinion. This example illustrates that our proposed model is capable of providing the interpretability for recommendation from the view of the user-preferred and user-rejected aspects.

VI. CONCLUSION AND FUTURE WORK

In this paper, we present a review polarity-wise recommender, dubbed as RPR, which treats reviews with different polarities discriminately. Specifically, RPR simultaneously learns the scores and importance of the user-preferred and user-rejected aspects to a user. The final rating is then estimated via the mathematical difference of the positive and negative scores, which are the weighted sum of the relevance scores with the corresponding importance. To overcome the problem of imbalanced review polarity, RPR implements an aspect-aware importance weighting module to effectively learn the mapping relations for one aspect importance based on the other. In addition to its remarkable performance over eight datasets, RPR is also capable of providing explicit explanation for the recommendation results.

In future, we will apply pairwise learners to strengthen RPR and validate its effectiveness via extensive experiments. Moreover, we are particularly interested in discriminating the user-preferred and user-rejected aspects of multi-media items [48], which contain abundant aspect information to

TABLE IV
POSITIVE AND NEGATIVE REVIEWS OF A RANDOMLY SELECTED USER FROM *Musical Instruments*.

Positive Reviews	Negative Reviews
...My guitar player had a different lock system, and his \$1000 Les Paul fell to the stage, completely knocking it out of tune. Mine stayed locked perfectly...	It's a decent unit, but stopped working completely after about 6 months. Tried every thing I could, but it's gone. I would not recommend this pedal.
These strings are nice and easy to fly over. No buildup or residue on your fingers either. I really like this stuff.	The plastic piece that screws the wire into the end was made of very thin plastic, and cracked in two within a week of light use.
Nothing sounds as bright as these strings. I've tried many, and these are the best by far. I know, cause I play guitar really god-like.	It stopped working after 2 gigs. I'm not sure why, but it is very frustrating. I guess you get what you pay for here.
I often use this to record my band's gigs. Now if they would make one that will hold a PAR 38 so I can turn my extra microphone stands into single can lighting racks!	I expected more from this manufacturer, but I guess the quality is not the same as it used to be.

TABLE V
TOP 10 WORDS FOR EACH USER-PREFERRED AND USER-REJECTED ASPECT OF THE SELECTED USER FROM *Musical Instruments*. THE “ASPECT LABELS” ARE ATTACHED BASED ON THE INTERPRETATION OF THE ASPECT.

User-preferred Aspects	User-rejected Aspects
Performance	Fine Strings
record	tried
system	buildup
gigs	fingers
stayed	extra
tune	perfectly
band	sounds
guitar	strings
stuff	easy
stage	bright
knocking	guitar
microphone	lock
Poor Quality	
Crack Sensitive	

TABLE VI
INTERPRETATION ON WHY THE USER RATED “ITEM 1” AND “ITEM 2” WITH 5 AND 1, RESPECTIVELY. THE EXAMPLES ARE OBTAINED BASED ON *Musical Instruments*.

Aspects	Importance	Score (1)	Score (2)
Performance	0.992	0.71	0.681
Fine Strings	1.873	0.83	0.752
Poor Quality	1.996	-0.67	0.185
Crack Sensitive	0.967	-0.898	0.723

reflect users’ preferences. Another interesting direction is to extend RPR to solve the long-tail recommendation problem by extracting semantic information from the attribute descriptions of less frequent items, which requires urgent solution for e-commerce platforms.

REFERENCES

- [1] Z. Cheng, X. Chang, L. Zhu, R. C. Kanjirathinkal, and M. Kankanhalli, “Mmalfm: Explainable recommendation by leveraging reviews and images,” *ACM TOIS*, vol. 37, no. 2, pp. 1–28, 2019.
- [2] K. Liu, X. Li, Z. Zhu, L. Brand, and H. Wang, “Factor-bounded nonnegative matrix factorization,” *ACM TKDD*, vol. 15, no. 6, pp. 1–18, 2021.
- [3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *NIPS*, 2013, pp. 3111–3119.
- [4] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in *EMNLP*. ACL, 2014, pp. 1532–1543.
- [5] Y. Guo, Z. Cheng, J. Jing, Y. Lin, L. Nie, and M. Wang, “Enhancing factorization machines with generalized metric learning,” *IEEE TKDE*, 2020.
- [6] L. Zheng, V. Noroozi, and P. S. Yu, “Joint deep modeling of users and items using reviews for recommendation,” in *WSDM*. ACM, 2017, pp. 425–434.
- [7] R. Catherine and W. Cohen, “Transnets: Learning to transform for recommendation,” in *RecSys*. ACM, 2017, pp. 288–296.
- [8] S. Seo, J. Huang, H. Yang, and Y. Liu, “Interpretable convolutional neural networks with dual local and global attention for review rating prediction,” in *RecSys*. ACM, 2017, pp. 297–305.
- [9] Y. Tay, A. T. Luu, and S. C. Hui, “Multi-pointer co-attention networks for recommendation,” in *KDD*. ACM, 2018, pp. 2309–2318.
- [10] K. Bauman, B. Liu, and A. Tuzhilin, “Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews,” in *KDD*. ACM, 2017, pp. 717–725.
- [11] N. Wang, H. Wang, Y. Jia, and Y. Yin, “Explainable recommendation via multi-task learning in opinionated text data,” in *SIGIR*. ACM, 2018, pp. 165–174.
- [12] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, “Aspect-aware latent factor model: Rating prediction with ratings and reviews,” in *WWW*. ACM, 2018, pp. 639–648.
- [13] J. Y. Chin, K. Zhao, S. Joty, and G. Cong, “Anr: Aspect-based neural recommender,” in *CIKM*. ACM, 2018, pp. 147–156.
- [14] T. Chen, H. Yin, G. Ye, Z. Huang, Y. Wang, and M. Wang, “Try this instead: Personalized and interpretable substitute recommendation,” in *SIGIR*. ACM, 2020, pp. 891–900.
- [15] Y. Lyu, H. Yin, J. Liu, M. Liu, H. Liu, and S. Deng, “Reliable recommendation with review-level explanations,” in *ICDE*. IEEE, 2021, pp. 1548–1558.
- [16] H. Wang, Y. Fu, Q. Wang, H. Yin, C. Du, and H. Xiong, “A location-sentiment-aware recommender system for both home-town and out-of-town users,” in *KDD*. ACM, 2017, pp. 1135–1143.
- [17] S. Purushotham, Y. Liu, and C.-C. J. Kuo, “Collaborative topic regression with social matrix factorization for recommendation systems,” in *ICML*. icml.cc, 2012.
- [18] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, “Convolutional matrix factorization for document context-aware recommendation,” in *RecSys*. ACM, 2016, pp. 233–240.
- [19] X. Chen, Z. Qin, Y. Zhang, and T. Xu, “Learning to rank features for recommendation over multiple categories,” in *SIGIR*. ACM, 2016, pp. 305–314.
- [20] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, “Explicit factor models for explainable recommendation based on phrase-level sentiment analysis,” in *SIGIR*. ACM, 2014, pp. 83–92.
- [21] D. W. Otter, J. R. Medina, and J. K. Kalita, “A survey of the usages of deep learning for natural language processing,” *IEEE TNNLS*, vol. 32, no. 2, pp. 604–624, 2020.
- [22] X. He, T. Chen, M.-Y. Kan, and X. Chen, “Trirank: Review-aware explainable recommendation by modeling aspects,” in *CIKM*. ACM, 2015, pp. 1661–1670.
- [23] C. Li, C. Quan, L. Peng, Y. Qi, Y. Deng, and L. Wu, “A capsule network for recommendation and explaining what you like and dislike,” in *SIGIR*. ACM, 2019, pp. 275–284.
- [24] X. Luo, M. Zhou, S. Li, Z. You, Y. Xia, and Q. Zhu, “A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method,” *IEEE TNNLS*, vol. 27, no. 3, pp. 579–592, 2015.
- [25] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu, “On deep learning for trust-aware recommendations in social networks,” *IEEE TNNLS*, vol. 28, no. 5, pp. 1164–1177, 2016.
- [26] H. Wang, N. Wang, and D.-Y. Yeung, “Collaborative deep learning for recommender systems,” in *KDD*. ACM, 2015, pp. 1235–1244.
- [27] Y. Wei, X. Wang, L. Nie, X. He, and T.-S. Chua, “Graph-refined convolutional network for multimedia recommendation with implicit feedback,” in *MM*. ACM, 2020, pp. 3541–3549.

- [28] Y. Wei, X. Wang, Q. Li, L. Nie, Y. Li, X. Li, and T.-S. Chua, “Contrastive learning for cold-start recommendation,” *arXiv preprint arXiv:2107.05315*, 2021.
- [29] J. Han, L. Zheng, Y. Xu, B. Zhang, F. Zhuang, S. Y. Philip, and W. Zuo, “Adaptive deep modeling of users and items using side information for recommendation,” *IEEE TNNLS*, vol. 31, no. 3, pp. 737–748, 2019.
- [30] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *WWW*. ACM, 2017, pp. 173–182.
- [31] G. Zhu, L. Zhang, L. Yang, L. Mei, S. A. A. Shah, M. Bennamoun, and P. Shen, “Redundancy and attention in convolutional lstm for gesture recognition,” *IEEE TNNLS*, vol. 31, no. 4, pp. 1323–1335, 2019.
- [32] X. Xu, T. Wang, Y. Yang, L. Zuo, F. Shen, and H. T. Shen, “Cross-modal attention with semantic consistence for image–text matching,” *IEEE TNNLS*, vol. 31, no. 12, pp. 5412–5425, 2020.
- [33] Y. Guo, Z. Cheng, L. Nie, Y. Wang, J. Ma, and M. Kankanhalli, “Attentive long short-term preference modeling for personalized product search,” *ACM TOIS*, vol. 37, no. 2, pp. 1–27, 2019.
- [34] Z. Huang, X. Xu, H. Zhu, and M. Zhou, “An efficient group recommendation model with multiattention-based neural networks,” *IEEE TNNLS*, vol. 31, no. 11, pp. 4461–4474, 2020.
- [35] X. Guan, Z. Cheng, X. He, Y. Zhang, Z. Zhu, Q. Peng, and T.-S. Chua, “Attentive aspect modeling for review-aware recommendation,” *ACM TOIS*, vol. 37, no. 3, pp. 1–27, 2019.
- [36] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, “Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention,” in *SIGIR*. ACM, 2017, pp. 335–344.
- [37] X. He, Z. He, J. Song, Z. Liu, Y.-G. Jiang, and T.-S. Chua, “Nais: Neural attentive item similarity model for recommendation,” *IEEE TKDE*, vol. 30, no. 12, pp. 2354–2366, 2018.
- [38] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, “Attentional factorization machines: Learning the weight of feature interactions via attention networks,” in *IJCAI*. ijcai.org, 2017, pp. 3119–3125.
- [39] Z. Cheng, Y. Ding, X. He, L. Zhu, X. Song, and M. Kankanhalli, “A3ncf: an adaptive aspect attention model for rating prediction,” in *IJCAI*. ijcai.org, 2018, pp. 3748–3754.
- [40] X. He and T.-S. Chua, “Neural factorization machines for sparse predictive analytics,” in *SIGIR*. ACM, 2017, pp. 355–364.
- [41] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, “Very deep convolutional networks for text classification,” in *EACL*. ACL, 2017, pp. 1107–1116.
- [42] Y. Kim, “Convolutional neural networks for sentence classification,” in *EMNLP*. ACL, 2014, pp. 1746–1751.
- [43] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [44] W. Wang, F. Feng, X. He, L. Nie, and T.-S. Chua, “Denoising implicit feedback for recommendation,” in *WSDM*. ACM, 2021, pp. 373–381.
- [45] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *AISTATS*. JMLR.org, 2010, pp. 249–256.
- [46] S. Rendle, “Factorization machines with libfm,” *ACM TIST*, vol. 3, no. 3, pp. 1–22, 2012.
- [47] X. Wang, X. He, F. Feng, L. Nie, and T.-S. Chua, “Tem: Tree-enhanced embedding model for explainable recommendation,” in *WWW*. ACM, 2018, pp. 1543–1552.
- [48] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T.-S. Chua, “Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video,” in *MM*. ACM, 2019, pp. 1437–1445.