DIRECT: Dual Interpretable Recommendation with Multi-aspect Word Attribution

XUANSHENG WU, University of Georgia, USA
HANQIN WAN, University of Georgia, USA
QIAOYU TAN, Texas A&M University, USA
WENLIN YAO, Tencent AI Lab, Bellevue, USA
NINGHAO LIU, University of Georgia, USA

Recommending products to users with intuitive explanations helps improve the system in transparency, persuasiveness, and satisfaction. Existing interpretation techniques include post-hoc methods and interpretable modeling. The former category could quantitatively analyze input contribution to model prediction but has limited interpretation faithfulness, while the latter could explain model internal mechanisms but may not directly attribute model predictions to input features. In this study, we propose a novel Dual Interpretable Recommendation model called DIRECT, which integrates ideas of the two interpretation categories to inherit their advantages and avoid limitations. Specifically, DIRECT makes use of item descriptions as explainable evidence for recommendation. First, similar to the post-hoc interpretation, DIRECT could attribute the prediction of a user preference score to textual words of the item descriptions. The attribution of each word is related to its sentiment polarity and word importance, where a word is important if it corresponds to an item aspect that the user is interested in. Second, to improve the interpretability of embedding space, we propose to extract high-level concepts from embeddings, where each concept corresponds to an item aspect. To learn discriminative concepts, we employ a concept-bottleneck layer, and maximize the coding rate reduction on word-aspect embeddings by leveraging a word-word affinity graph extracted from a pre-trained language model. In this way, DIRECT simultaneously achieves faithful attribution and usable interpretation of embedding space. We also show that DIRECT achieves linear inference time complexity regarding the length of item reviews. We conduct experiments including ablation studies on five real-world datasets. Quantitative analysis, visualizations, and case studies verify the interpretability of DIRECT. Our code is available at: https://github.com/JacksonWuxs/DIRECT.

CCS Concepts: • **Information systems** → **Personalization**; **Collaborative search**.

Additional Key Words and Phrases: Recommendation Systems, Explainable AI, Language Models.

1 INTRODUCTION

Recommender systems help users to access items matching their preferences. While deep learning significantly improves recommendation accuracy, increasing the transparency of recommender systems to users also becomes a new trend recently [10]. Interpretable recommender systems [60, 66] attempt to generate both *accurate predictions* and *intuitive explanations*. However, the two goals often sit on opposite sides of a seesaw. While deep learning

Authors' addresses: Xuansheng Wu, xuansheng.wu@uga.edu, University of Georgia, Athens, Georgia, USA; Hanqin Wan, wanhanqin8@gmail.com, University of Georgia, Athens, Georgia, USA; Qiaoyu Tan, qytan@tamu.edu, Texas A&M University, College Station, Texas, USA; Wenlin Yao, wenlin.yao.cs@gmail.com, Tencent AI Lab, Bellevue, Bellevue, Washington, USA; Ninghao Liu, ninghao.liu@uga.edu, University of Georgia, Athens, Georgia, USA.

ACM 2157-6912/2024/5-ART
https://doi.org/10.1145/3663483

models represent features in uninterpretable high-dimensional spaces to pursue better accuracy, they also sacrifice system transparency.

There are two main categories of techniques for creating interpretable recommender systems. The first category is the post-hoc explanation to understand how predictions are made after a model is trained. Typical techniques include gradient-based [45, 47, 59], path-based [43, 62], and perturbation-based methods [32, 52]. However, it has been pointed out that post-hoc methods may not always faithfully explain the exact inference mechanism [21, 66]. The second category is to build inherently interpretable models. Typical techniques are attention models [8, 11, 41, 53, 54] and disentangled representation learning [13, 26, 33, 56]. Attention weights shed light on how information propagates over user-item interaction graphs, while disentangled factors unravel the global distribution of representations. The attention weights or disentangled factors could help understand certain aspects of the model inference process, but these intermediate information is not directly associated with the output, such as prediction scores. This is different from the post-hoc interpretation like Shapley Values [32] or Counterfactual Analysis [47] that directly decompose output and quantitatively attribute it back to input features. To satisfy both performance and transparency, recent studies [24, 70] suggest that let neural networks map inputs into a human understandable latent space, then apply a linear transformation from this space to the target label set, known as Generalized Additive Model (GAM) [17]. However, GAM-based approaches often require the involvement of experts to define the latent space, which can limit the learning capabilities of deep learning models.

Considering the natural readability of textual user-item reviews, we put reviews forward as a latent space to achieve interpretable recommendations, where this latent space is easily understood by humans and carries rich semantic information for predicting user preferences. There are several challenges in building interpretable recommendation models with review information. First, how to design the model that preserves the advantages of both post-hoc interpretation and inherently interpretable modeling? Second, how to simultaneously utilize the capacity of advanced language models and protect recommendation interpretability? Third, since many users do not write reviews, how to overcome the sparsity issue?

To address the challenges, we propose a novel review-based interpretable recommendation model for user-item rating prediction. First, we design the rating function as the summation of attribution scores of review words. This allows both quantitative and intuitive attribution to the input reviews as post-hoc interpretations. Second, we employ a concept bottleneck layer [22, 24] to map review words into interpretable features before the output, where each feature corresponds to an aspect of items. Different from existing concept bottleneck models that require domain experts to design the features, our model automatically discovers these features from data. The above designs consider the attributable model predictions and human-understandable model mechanisms, so we name our model DIRECT (Dual Interpretable RECommendaTion). Third, we extract a word-word affinity graph from pre-trained language models, and then design an end-to-end solution, by leveraging the implicit community structure in the graph, to guarantee that non-trivial aspects are discovered from word representations. This allows us to effectively utilize pre-trained language models without harming model interpretability. Fourth, we jointly model user reviews and shopping history to merge the two information modalities. This allows our model to express user interests with reviews written by other customers. Furthermore, we introduce how to reduce time complexity for online model inference. The contributions of this work are summarized below.

- We propose a novel review-based interpretable recommender system called DIRECT. It inherits the advantages of both attribution-based interpretation and interpretable modeling mechanism to achieve a transparent decision-making process.
- We propose a novel objective that encourages the model to learn discriminative aspects.
- Experiments on real-world datasets validate the effectiveness of DIRECT. Visualization and case studies demonstrate the interpretability of our model.

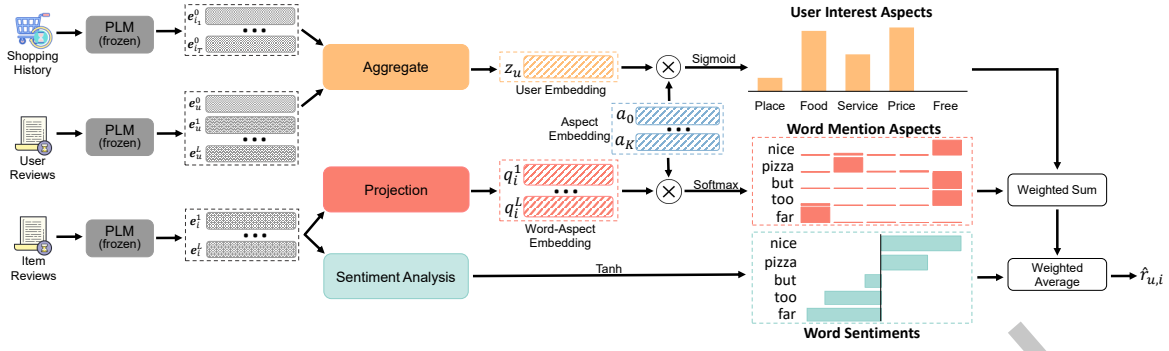


Fig. 1. The overall framework of the proposed interpretable review-based recommendation system.

2 PROBLEM STATEMENT

Notations. In this work, we use boldface lowercase letters (e.g., $\boldsymbol{x}$) to denote vectors, boldface uppercase letters (e.g., $\boldsymbol{A}$) to denote matrices, and calligraphic capital letters (e.g., $\mathcal{D}$) to denote sets. Specifically, we use $\mathcal{U}$ and $\mathcal{I}$ to denote the user set and item set, respectively. The interactions between users and items are stored in a rating matrix $\boldsymbol{R} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$, where each element $r_{u,i}$ indicates the rating score of user $u \in \mathcal{U}$ to item $i \in \mathcal{I}$. In review-based recommendation systems, we also assume that an M-word review is available for some rating actions, denoted by $\mathcal{T}_{u,i} = [w^1_{u,i}, ..., w^m_{u,i}, ..., w^M_{u,i}]$. $w^m_{u,i} \in \mathcal{V}$ indicates the m-th word of the review posed by user u to item i, and $\mathcal{V}$ is a pre-defined vocabulary set. Besides the rating-level review, we also represent each user u (or item i) with a summarized document $\mathcal{D}_u = [w^1_u, ..., w^l_u, ..., w^L_u]$ (or $\mathcal{D}_i = [w^1_i, ..., w^l_i, ..., w^L_i]$), where L is the document length. In practice, each document $\mathcal{D}_u$ (or $\mathcal{D}_i$) is obtained by concatenating all the observed reviews that are commented by the user (or commented on the item). These settings have been widely adopted in existing review-based recommender systems [9, 69].

Problem Definition. The goal of this work is to build an interpretable model f to predict the preference score $\hat{r}_{u,i} = f(u, i, \mathcal{D}_u, \mathcal{D}_i)$ of user u on item i. In real-world scenarios, the review $\mathcal{T}_{u,i}$ is not available for a target user-item pair (u, i), so we always delete the current review from the user and item document during training.

3 PROPOSED METHOD

We now introduce the proposed interpretable recommendation model with user reviews. First, in Section 3.1, we formally define the "interpretability" considered in this work. Then, we describe the architecture design of our multi-aspect recommendation model in Section 3.2~ 3.3. After that, in Section 3.4, we propose a training loss to guarantee the distinction among different aspects. We then introduce the overall objective to train our model in Section 3.5. Finally, we discuss the time complexity of model inference in Section 3.6.

3.1 Interpretability of Recommender Systems

We consider two levels of interpretability in designing the proposed recommender system. First, similar to the post-hoc methods, we want prediction scores to be *attributable*, where several requirements are as below.

- **Attributable prediction:** Let $\boldsymbol{x}$ denote the input, x_i denote the i-th feature, and $f(\boldsymbol{x})$ be the prediction for $\boldsymbol{x}$. The prediction is attributable if we could design an interpretation method $intp()$, where $intp(f, \boldsymbol{x}, x_i)$ returns the attribution score of x_i for $f(\boldsymbol{x})$. We think x_i is more important than x_j if $|intp(f, \boldsymbol{x}, x_i)| > |intp(f, \boldsymbol{x}, x_j)|$. Commonly used attribution methods include raw gradient interpretation [45] and attention scores [8, 11, 53, 54].
- **Measurable attribution:** An attribution is measurable if:

$$f(\boldsymbol{x}) \approx \sum\nolimits_i intp(f, \boldsymbol{x}, x_i). \tag{1}$$

The measurability further requires attributions to compose the prediction value. It thus makes interpretation a quantitative analytic tool. Commonly used measurable attribution methods include Integrated Gradients [47] and Generalized Additive Models [3]. Interpretation methods in the previous category, such as attention scores, do not have this property.

- **Comprehensible attribution:** An attribution is comprehensible if it is easy for humans to understand the meaning of each x_i. For example, each pixel in an image is hardly comprehensible, while objects are more comprehensible [27]. In recommender systems, we regard words in user reviews as comprehensible.

The second level of interpretability refers to a more inherently understandable model mechanism. State-of-the-art neural networks are typically designed in a way that maps input (e.g., nodes, texts) through complicated interactions to the target (e.g., labels). Such a prediction scheme, however, does not match the human cognition habits which rely on high-level *concepts or aspects*. To bridge the gap, we introduce the idea of Concept Bottleneck [24] for building interpretable recommendation models. Specifically, the model consists of two parts. The first part $f_1 : \mathcal{X} \to \mathcal{K}$ maps input to the concept space. The second part $f_2 : \mathcal{K} \to \mathcal{Y}$ makes the final predictions based on the concepts. Thus, given an input x, its concept activation is denoted as $f_1(\boldsymbol{x}) \in \mathbb{R}^K$, where K is the number of concepts. Then, a prediction is made as $\hat{y} = f_2(f_1(\boldsymbol{x}))$. In traditional concept bottleneck models [22, 24, 70], the concepts are provided by domain experts, where the models are trained to fit both concept labels and the prediction label y. However, in our problem, the concepts are not pre-defined and are discovered from data. It is also worth noting that, although some neural recommendation models use linear functions or Factorization Machines [39] as their scoring functions, they do not fully match our definition of interpretability. For example, DeepCoNN [69] uses a dot production as its scoring function upon user embeddings and item embeddings. However, the embeddings are generated by a TextCNN model [7], where the latent space dimensions are not interpretable.

3.2 Model Architecture

We introduce the architecture of our model in this section. Figure 1 presents the overall framework of our method. The general idea is to predict user preference for an item by attributing the preference score to each of the words in item reviews. The score of each word is the product of two factors: (1) the sentiment of the word, and (2) the degree of user interest in the item's aspect described by the word. Specifically, let $w_i^l \in \mathcal{D}_i$ denote the l-th word in the reviews of the i-th item. The preference of user u for item i is predicted as:

$$\hat{r}_{u,i} = \frac{1}{L} \sum_{l=1}^{L} sentiment(w_i^l) \times gate_u(w_i^l) + b_i + b_u + b_g, \tag{2}$$

where b_i, b_u and b_g are trainable item bias, user bias and global bias terms, respectively, and $L = |\mathcal{D}_i|$. Here we design a sentiment analysis module $sentiment : \mathbb{R}^{h_1} \to [-1, 1]$ indicating the sentiment of word w_i^l, and an aspect-interest gate function $gate_u : \mathbb{R}^{h_1} \to [0, 1]$ computing the probability that the word w_i^l describes an item aspect that user u interested in. Meanwhile, a word representation module serves as one of the foundations of our review-based system. The details of each module are introduced below.

3.2.1 Word Representation Module. To obtain high-quality word embeddings with rich semantic information to support other functionalities, we collect contextual word embedding by using pre-trained language models (PLM). This module takes reviews as input and returns an h_1-dimensional embedding $\boldsymbol{e}$ for each of the word tokens as output. Formally,

$$[\boldsymbol{e}^0, \boldsymbol{e}^1, \boldsymbol{e}^2, ..., \boldsymbol{e}^L] = \text{PLM}([w^0, w^1, w^2, ..., w^L]), \tag{3}$$

where $\boldsymbol{e}^l$ denotes the contextual embedding of word w^l. We insert a special token $w^0 = [CLS]$ to each input word sequence. The contextual embedding of the special token $[CLS]$ is used as the representation of the whole

sequence. If the input is a user review document $\mathcal{D}_u$ or item review document $\mathcal{D}_i$, then $\boldsymbol{e}^0$ could be treated as the overall representation of the user or item from the perspective of their review information.

3.2.2 Sentiment Analysis Module. The sentiment analysis module tries to predict the sentiment polarity of each word in item reviews. Considering that the sentiment analysis over words is a relatively simple task with high-quality pre-trained contextual word embeddings, we implement it as a light-weight multi-layer perception (MLP) model that maps the word embedding to a scalar:

$$sentiment(w_i^l) = \tanh(\text{MLP}_1(\boldsymbol{e}_i^l)), \tag{4}$$

where $\boldsymbol{e}_i^l$ denotes the embedding of word w_i^l, and $\text{MLP}_1 : \mathbb{R}^{h_1} \to \mathbb{R}$. Here, the tanh function returns a value between -1 and 1, where a greater value indicates a stronger positive signal. Since the final estimated score is a weighted addition to these sentiment scores, the estimation errors between the ground truth and predicted ratings directly guide the learning of the sentiment analysis module.

3.2.3 Aspect-Guided Interest Gate. A user usually evaluates a product from several aspects. A review segment is useful to the user if it describes the aspects that the user is interested in. Otherwise, the content of that segment will be ignored by the user. For example, a review of a restaurant is "Their pizzas are full of flavor and have a crispy crust, but it is far away, by the way.", where the word "pizza" showing the Food aspect of the restaurant which is positive, and the word "far away" reflecting the Location aspect is negative. This review will only be noticed by the users who care about these aspects of a restaurant. For example, a gourmet will focus on the sentiment score of the pizza in this review and ignore the comments on the restaurant location, while a student without a car may not choose this restaurant due to the negative comment on the location.

Following this intuition, we design the Aspect-Guided Interest Gate (AGIG) to quantify the degree to a word $w_i^l \in \mathcal{D}_i$ falling in the aspects of user u's interest. We assume all items within share K aspects that users might concern about, such as Price, Service, Location, and Food in restaurant recommendation. The AGIG module plays the same role as the Concept Bottleneck layer introduced before, where each concept corresponds to an aspect of items. Formally, we assign each aspect k a h_2-dimensional vector $\boldsymbol{a}_k \in \mathbb{R}^{h_2}$, and form the total K aspects in an aspect matrix $\boldsymbol{A} \in \mathbb{R}^{K \times h_2}$. Suppose we have generated a user embedding $\boldsymbol{z}_u \in \mathbb{R}^{h_3}$ for user u (see details in Section 3.3), then we define the AGIG as:

$$gate_u(\boldsymbol{e}_i^l) = \sum_{k=1}^{K} P(\boldsymbol{a}_k|\boldsymbol{e}_i^l) \times P(\boldsymbol{a}_k|\boldsymbol{z}_u), \tag{5}$$

where

$$0 \le P(\boldsymbol{a}_k|\boldsymbol{z}_u) \le 1, \ \sum_{k=1}^{K} P(\boldsymbol{a}_k|\boldsymbol{e}_i^l) = 1. \tag{6}$$

Here $P(\boldsymbol{a}_k|\boldsymbol{e}_i^l)$ is the probability that word w_i^l mentions the aspect $\boldsymbol{a}_k$ with its contextual word embedding $\boldsymbol{e}_i^l$, and $P(\boldsymbol{a}_k|\boldsymbol{z}_u)$ is the probability that user u is interested in aspect $\boldsymbol{a}_k$. The former probability is user-independent, while the latter is user-specific. For a word to contribute to the recommendation, it must express sufficient information on the aspects that the user cares about.

- We estimate the distribution $P(\boldsymbol{a}_k|\boldsymbol{e}_i^l)$ of a word $w_i^l \in \mathcal{D}_i$ on different aspects $\{\boldsymbol{a}_k\}_{k=1}^K$ as below:

$$P(\boldsymbol{a}_k|\boldsymbol{e}_i^l) = \frac{\exp\left(\boldsymbol{q}_i^l \cdot \boldsymbol{a}_k^\top\right)}{\sum_{k'=1}^{K} \exp\left(\boldsymbol{q}_i^l \cdot \boldsymbol{a}_{k'}^\top\right)}, \tag{7}$$

  where $\boldsymbol{q}_i^l = \text{MLP}_2(\boldsymbol{e}_i^l)$, and $\text{MLP}_2 : \mathbb{R}^{h_1} \to \mathbb{R}^{h_2}$ bridges between the word embedding space and the aspect embedding space. Here $\boldsymbol{q}_i^l \in \mathbb{R}^{h_2}$ is called word-aspect embedding.

- We compute the distribution $P(\boldsymbol{a}_k|\boldsymbol{z}_u)$ of user interests on different aspects $\{\boldsymbol{a}_k\}_{k=1}^K$ as below:

$$P(\boldsymbol{a}_k|\boldsymbol{z}_u) = \sigma(\text{MLP}_3(\boldsymbol{z}_u) \cdot \boldsymbol{a}_k^\top), \tag{8}$$

  where $\text{MLP}_3 : \mathbb{R}^{h_3} \rightarrow \mathbb{R}^{h_2}$ aligns the user embedding space with the aspect embedding space, σ is the Sigmoid activation function.

In the above, Equation (7) measures the correlation between the word-aspect embeddings $\{\boldsymbol{q}_i^l\}$ and the aspect embeddings $\{\boldsymbol{a}_k\}_{k=1}^K$. This is similar to K-means clustering in the gradient-descent form [1, 40] if we could optimize each centroid to minimize its distances to the nearby data points. Since one of the main requests of DIRECT for better recommendation quality is to correctly predict the aspects reflected by each word, the distance between the words and their closest aspect has a chance to be minimized. However, different from traditional K-means algorithms where the input data samples are fixed, we aim to identify topical aspects from word-aspect embeddings that are trainable. Directly optimizing both $\{\boldsymbol{q}_i^l\}$ and $\{\boldsymbol{a}_k\}_{k=1}^K$ via gradient descent could cause model collapse. To avoid this problem, in Section 3.4, we discuss further constraints on the aspect distribution towards producing diverse aspects.

3.3 Learning User Representations

Effective user representations $\{\boldsymbol{z}_u\}$ is the key to personalized recommendation in our system. Modeling user-item interactions is one of the popular directions to generating user embeddings [25, 55, 63]. However, user interests hiding in their historical interactions are implicit. A straightforward way is generating user embedding from their reviews [28, 65, 69], where user preferences are explicit. But, another issue is raised: many reviews are biased, sparse, and incomprehensive because most users only write reviews when they feel the items are particularly bad or beyond expectations.

To fill the gaps, we utilize user shopping histories to be combined with the review information. The user shopping history is denoted as $\mathcal{I}_u$, where $\mathcal{I}_u = \{i_1, ..., i_t, ..., i_T\}, i_t \in \mathcal{I}$, is a T-size item set storing the items purchased by the user u. We then design a fusion network to generate the final user representations.

3.3.1 Representing Users with Sequences. Both user reviews and shopping history are processed as two sequences of h_1-dimensional embeddings. We use a self-attention module to aggregate the two sequences into two vectors $\boldsymbol{z}_{u,d}$ and $\boldsymbol{z}_{u,h}$, and then use the fusion network to merge them into $\boldsymbol{z}_u$. First, given the user review document $\mathcal{D}_u$ and its contextual word embeddings $[\boldsymbol{e}_u^1, ..., \boldsymbol{e}_u^l, ..., \boldsymbol{e}_u^L]$ from the pre-trained language model, we aggregate them into a single embedding as $\boldsymbol{z}_{u,d} \in \mathbb{R}^{h_1}$:

$$\boldsymbol{z}_{u,d} = \text{AGGR}([\boldsymbol{e}_u^1, ..., \boldsymbol{e}_u^L], \boldsymbol{e}_u^0) = \sum_{l=1}^{L} \alpha_l \boldsymbol{e}_u^l, \tag{9}$$

where $\boldsymbol{e}_u^0$ acts as the query for self-attention,

$$\alpha_l = \frac{\exp(\tilde{\boldsymbol{e}}_l)}{\sum_{l'=1}^{L} \exp(\tilde{\boldsymbol{e}}_{l'})}, \quad \tilde{\boldsymbol{e}}_l = \lambda_0 \cdot \tanh(\boldsymbol{e}_u^0 \cdot FC(\boldsymbol{e}_u^l)^\top). \tag{10}$$

Then, given the user shopping history $\mathcal{I}_u$ and history embedding sequence $[\boldsymbol{e}_{i_1}^0, ..., \boldsymbol{e}_{i_t}^0, ..., \boldsymbol{e}_{i_T}^0]$, we adopt the similar aggregation function to generate a single user history embedding $\boldsymbol{z}_{u,h} \in \mathbb{R}^{h_1}$, where $\boldsymbol{z}_{u,h} = \text{AGGR}([\boldsymbol{e}_{i_1}^0, ..., \boldsymbol{e}_{i_T}^0], \bar{\boldsymbol{e}}_u^0)$, and $\bar{\boldsymbol{e}}_u^0 = \frac{1}{T}\sum_{t=1}^{T} e_{i_t}^0$ is the mean.

3.3.2 Embedding Fusion Network. As discussed earlier, the user document embedding $z_{u,d}$ keeps explicit but biased user preferences, while the history embedding $z_{u,h}$ contains more general but implicit information. We design a fusion network to filter information coming from each embedding guided by the other resource to generate the final user embedding $\boldsymbol{z}_u$:

$$\begin{aligned} \boldsymbol{z}_u &= [\boldsymbol{z}_{u,h} \odot \boldsymbol{s}_h; \boldsymbol{z}_{u,d} \odot \boldsymbol{s}_d], \\ \boldsymbol{s}_h &= \sigma(\mathrm{MLP}_4(\boldsymbol{z}_{u,d})), \quad \boldsymbol{s}_d = \sigma(\mathrm{MLP}_5(\boldsymbol{z}_{u,h})), \end{aligned} \tag{11}$$

where $\odot$ stands for element-wise multiplication, $\mathrm{MLP}_4 : \mathbb{R}^{h_1} \to \mathbb{R}^{h_1}$, and $\mathrm{MLP}_5 : \mathbb{R}^{h_1} \to \mathbb{R}^{h_1}$. If user reviews are not available, we can simply let $\boldsymbol{z}_u = \boldsymbol{z}_{u,h}$. Here, s_h and s_d are gates to filter out redundant information. Inspired by SE-Net [20], each gate first represents its input embedding into a lower-dimensional hidden space and then maps them back to the original space with a gated activation function (e.g., the sigmoid function). With this design, each gate could identify the essential information from its inputs and use them as guidance to filter out the original information.

3.4 Learning Discriminative Aspect Representations

Learning diverse and comprehensive aspects is crucial for the interpretability of our recommendation model, which is also a non-trivial task. There are two categories of methods to learn aspect-based embeddings. The first category relies on a two-step procedure [13, 30], i.e., it conducts clustering to find the aspects and then learns embeddings. While this approach could produce human-understandable clusters, it is difficult to guarantee the quality of clustering results used for embedding. The second category jointly conducts aspect discovery and embedding learning in an end-to-end manner [33, 36, 37], which aims to learn word embeddings and interpretable aspects simultaneously. However, the model could suffer from *mode collapse* [23, 33], since there is no explicit constraint to control the diversity of embedding distribution. Specific to our paper, the mode collapse is caused by jointly learning aspect and word-aspect embeddings during model training, which differs from the K-means clustering [1] assuming the input entries (word-aspect embeddings here) are fixed. This could lead to a trivial solution that maps every aspect and word-aspect embedding to the same point, while the objective of K-means is minimized, i.e., the distances between words and their closest aspects are zero. To tackle the challenges, we propose a new end-to-end approach with an explicit objective to learn discriminative representations of words in different aspects. Specifically, we leverage the idea of maximizing coding rate reduction (MCR^2) of representations [64], which encourages the words sharing similar semantic concepts to be represented closer and pushes the representations of semantically different words further. Therefore, this constraint can be regarded as our prior on the word-aspect space to prevent the model from collapsing into trivial solutions.

3.4.1 Maximization of Coding Rate Reduction. In information theory, the coding rate [34] is defined as the minimum number of binary bits needed to encode a set of data instances with a prescribed precision $\epsilon > 0$. Intuitively, the coding rate of a dataset is large if its instances are scattered in a broad spatial region. In supervised learning, if a dataset contains multiple classes, where each class is cohesive but instances in different classes are uncorrelated, then we can reduce the coding rate of the whole dataset by coding each subset separately and summing them up. Thus, to learn discriminative word representations distributed over multiple aspects, we want to maximize the coding rate reduction.

3.4.2 Unsupervised MCR^2. In recommender systems, we treat each item aspect as a class and aim at learning discriminative word representations between different aspects. However, in our problem, the labels are not available to assign words to aspects. To overcome this, we build a word-word affinity graph with an adjacency matrix $\boldsymbol{G}$, where $\boldsymbol{G}_{i,j}$ denotes the semantic similarity between word i and word j, and leverage the *group information* implicitly contained in $\boldsymbol{G}$ [16]. The words sharing similar semantic meanings will form a group in the graph, and words with different meanings fall into different groups. Each group plays the role of a class. Formally, given an item document $\mathcal{D}_i$ with L distinct words, the adjacency matrix $\boldsymbol{G} = [\boldsymbol{g}^1, ..., \boldsymbol{g}^l, ..., \boldsymbol{g}^L] \in \mathbb{R}^{L \times L}$, where $\boldsymbol{g}^l \in \mathbb{R}^L$. In addition, the word-aspect embeddings $\{\boldsymbol{q}_i^l\}_{l=1}^L$ form the matrix $\boldsymbol{Q} \in \mathbb{R}^{L \times h_2}$. The objective of learning

discriminate word-aspect representations $\boldsymbol{Q}$ is thus formulated as:

$$\Omega_d = R(\boldsymbol{Q}, \epsilon) - R^c(\boldsymbol{Q}, \epsilon|\boldsymbol{G}). \tag{12}$$

where $R(\boldsymbol{Q}, \epsilon)$ is the coding rate of the entire representations, and $R^c(\boldsymbol{Q}, \epsilon|\boldsymbol{G})$ denotes the summation of the coding rates over groups. Since the word-aspect matrix $\boldsymbol{Q}$ and the aspect embeddings $\boldsymbol{A}$ share the same latent space, optimizing Ω_d indirectly controls the distribution of aspect embeddings. Specifically,

$$\begin{aligned} R(\boldsymbol{Q}, \epsilon) &= \frac{1}{2\beta} \log\det(\boldsymbol{I} + \frac{\beta h_2}{L\epsilon^2}\boldsymbol{Q}^\top \boldsymbol{Q}), \\ R^c(\boldsymbol{Q}, \epsilon|\boldsymbol{G}) &= \sum_{l=1}^{L} \frac{\mathrm{tr}(\boldsymbol{G}^l)}{2L} \log\det(\boldsymbol{I} + \frac{h_2}{\mathrm{tr}(\boldsymbol{G}^l)\epsilon^2}\boldsymbol{Q}^\top \boldsymbol{G}^l \boldsymbol{Q}), \end{aligned} \tag{13}$$

where $\boldsymbol{I} \in \mathbb{R}^{h_2 \times h_2}$ is an identity matrix, $\boldsymbol{G}^l = \mathrm{diag}(g^l) \in \mathbb{R}^{L \times L}$ diagonalizes the word similarity vector, $\beta \in \mathbb{R}$ is a hyper-parameter to control the compactness of grouped word representations. Note that Ω_d trivially increases with the norm of $\boldsymbol{Q}$, so we need to normalize its columns into unit vectors. Intuitively, maximizing Ω_d equals to maximize $R(\boldsymbol{Q}, \epsilon)$ and minimize $R^c(\boldsymbol{Q}, \epsilon|\boldsymbol{G})$. The former encourages $\{\boldsymbol{q}_i^l\}_{l=1}^L$ to be mutually independent, and the latter encourages $\{\boldsymbol{q}_i^l\}$ within the same group to be correlated. Using Ω_d as a regularization term tends to separate word representations between different groups in $\boldsymbol{G}$, and squeeze word representations within the same group.

In this work, we use the cosine similarity between pre-trained word embeddings $\boldsymbol{v}^{l_1}, \boldsymbol{v}^{l_2}$ to measure the semantic similarity of two words w^{l_1} and w^{l_2}. The pre-trained embeddings are obtained from the first layer of *PLM*. We let $\boldsymbol{G}_{l_1,l_2} = \boldsymbol{G}_{l_2,l_1} = 1$ if $\cos(\boldsymbol{v}^{l_1}, \boldsymbol{v}^{l_2}) = \frac{\langle \boldsymbol{v}^{l_1}, \boldsymbol{v}^{l_2} \rangle}{\|\boldsymbol{v}^{l_1}\|_2 \|\boldsymbol{v}_2^{l_2}\|_2}$ is greater than a threshold T. Otherwise, we let $\boldsymbol{G}_{l_1,l_2} = \boldsymbol{G}_{l_2,l_1} = 0$ to build a sparse adjacency matrix.

3.4.3 Residual Aspect. Forcing every word to reflect an item aspect violates the truth that many words are not related to item aspects. Taking the review "My parents love this restaurant so much!" as an example, parents and love are nontrivial words but they are not related to any aspect of restaurants. To tackle this problem, we add an additional aspect called *residual aspect*, denoted as $\boldsymbol{a}_0 \in \mathbb{R}^{h_2}$, to the aspect embedding matrix $\boldsymbol{A}$, so the matrix finally has the shape of $K' \times h_2$, where $K' = K + 1$. Moreover, we let the user interest probability $P(\boldsymbol{a}_0|\boldsymbol{z}_u) = 0$ for the residual aspect to minimize the influence of residual-aspect words on the recommendation.

3.5 Objective Function

In this section, we introduce the overall objective function (including several terms) and training method in our model.

3.5.1 Prediction Loss. The major objective of our model is predicting user preference scores. We introduce prediction loss $\mathcal{L}_p$ to measure the differences between prediction scores and user rating scores. We follow the previous studies [9, 44] to measure the accuracy of the proposed model by using Mean Square Error (MSE):

$$\mathcal{L}_p = \frac{1}{|\mathcal{X}|} \sum_{(u,i) \in \mathcal{X}} (r_{u,i} - \hat{r}_{u,i})^2. \tag{14}$$

3.5.2 Contrastive Loss. Inspired by the idea of contrastive learning in graphs [5, 18], we consider the user/item history reviews $\mathcal{D}_u/\mathcal{D}_i$ and the current review $\mathcal{T}_{u,i}$ as the two views of the same user interests and the same item aspects in different moments. Suppose that user interests and the item aspects will not change in a short time period, then the preference scores estimated according to the history and the current reviews should be similar.

Table 1. Statistics of datasets.

Dataset	#Users	#Items	#Reviews	Density
Toys	19,412	11,924	167,597	0.1448%
Games	24,303	10,672	231,780	0.1787%
Clothing	39,387	23,033	278,677	0.0614%
Yelp2019	19,936	14,587	84,370	0.0580%
CDs	75,258	64,443	1,097,592	0.0453%

Thus, given the recommender f, we set up a contrastive loss to help model training:

$$\mathcal{L}_c = \frac{1}{|\mathcal{X}|} \sum_{(u,i)\in\mathcal{X}} (\hat{r}_{u,i} - \hat{r}'_{u,i})^2, \tag{15}$$

$$\begin{aligned} \hat{r}_{u,i} &= f(u, i, \mathcal{D}_u, \mathcal{D}_i), \\ \hat{r}'_{u,i} &= f(u, i, \mathcal{T}_{u,i}, \mathcal{T}_{u,i}). \end{aligned} \tag{16}$$

Following previous studies [6], we drop the gradients coming from calculating $\hat{r}'_{u,i}$ to prevent the collapsing issue (i.e., constantly predicting the same result regardless of inputs).

3.5.3 Training Loss. The final objective function for training is:

$$\mathcal{L} = \mathcal{L}_p + \gamma_1 \mathcal{L}_c + \gamma_2 \Omega_d, \tag{17}$$

where γ_1, γ_2 are hyper-parameters to balance the losses. Here Ω_d denotes the objective of maximizing coding rate reduction for word representation learning, as introduced in the previous subsection.

3.6 Analysis of Inference Complexity

In this part, we analyze the time complexity of model inference, where we assume one-layer architectures for all the MLPs in our model. Given a review document with the length of L, a P-layer Transformer-based PLM requires $O(P \cdot h_1 \cdot L^2)$ time to generate word embeddings. Then, the MLP_1 takes $O(L \cdot h_1)$ time to process L words; the MLP_2 and MLP_3 take $O(L \cdot h_1 \cdot h_2)$ and $O(h_3 \cdot h_2)$ time, respectively; the MLP_4 and MLP_5 both take $O(h_1^2)$ time; the user review AGGR function takes $O(L \cdot h_1)$ time. After that, given a T-length shopping history, the AGGR function takes $O(T \cdot h_1)$ time. Finally, mapping L words to K' aspects takes $O(L \cdot K' \cdot h_2)$ time, and computing the prediction score based on K' aspects takes $O(K'L)$ time. In total, since $h_3 = 2h_1$, the time complexity is $O(P \cdot h_1 \cdot L^2 + L \cdot h_1 \cdot h_2 + h_1^2 + L \cdot K' \cdot h_2)$.

The above computation is costly for online systems. Thus, to reduce online computation, we propose to cache some intermediate quantities, including word-aspect mentions (i.e., $P(\boldsymbol{a}_k|\boldsymbol{e}_i^l)$), user-aspect affiliations (i.e., $P(\boldsymbol{a}_k|\boldsymbol{z}_u)$), and word sentiments. Under this setting, the time complexity of our model is reduced to $O(L \cdot K')$. In practice, K' is small (e.g., the optimal $K' \approx 5$ as shown in Section 4.4), and L could be reduced by pre-processing reviews to select useful words. Empirically, the inference time of our model is comparable to that of Matrix Factorization [25].

4 EXPERIMENT

We try to answer four research questions through experiments. **Q1:** How effective is DIRECT compared with other SOTA baselines? **Q2:** How does each component contribute to the performance of DIRECT? **Q3:** How will DIRECT react to different numbers of aspects? **Q4:** How effective is DIRECT in learning interest aspects and generating interpretable recommendations?

4.1 Dataset

We evaluate DIRECT on 5 benchmarks including "Toys and Games" (Toys), "Video Games" (Games), "Clothing, Shoes and Jewelry" (Clothing), and "CDs and Vinyl" (CDs) subsets from the Amazon Review Dataset [35] and the popular Yelp dataset [1] based on the year of 2019 (Yelp2019). We use the five-core versions of these datasets, where each user/item has at least five reviews. We divide 70%, 10%, and 20% of each user's reviews to constitute the training, validation, and test sets respectively. The data statistics are summarized in Table 1, in which the Density is defined as $\frac{2\times\#Reviews}{\#Users\times\#Items}$.

4.2 Comparison with Baseline Methods

To answer **Q1**, we compare DIRECT with 13 state-of-the-art recommendation baselines below.

Baseline Methods. To have a rigorous and fair comparison, we include standard matrix factorization methods (BiasMF [68] and NeuMF [19]), language model enhanced methods (DeepCoNN [69], NARRE [4], and DAML [28]), aspect-aware methods (EMF [68], ANR [9], CARP [26], AARM [15] and UARM [46]), and graph-based review systems (SSG [14], RMG [57] and RGCL [44]).

Experimental Settings. For all baseline methods, we use their publicly available source codes for experiments, and tune their hyper-parameters based on the validation set. We train our model for 50 epochs with AdamW [31] optimizer and early stop which is triggered by two times the learning rate decay. The learning rate decay strategy with a decay factor of 0.1 is adopted, where the initial learning rate is $1e-3$. We set $\gamma_1 = 5e^{-3}$, $\gamma_2 = 1e^{-6}$, $K = 5$, and $h_2 = 64$ by default, and the batch size is fixed as 32. The dropout rates for the contextual word embedding and all MLP modules are 0.3 and 0.5, respectively. For the language model, we use the pre-trained BERT-small [49] with embedding size 512 to initialize the contextual word embedding in Equation (3). Following common practice [44], our text preprocessing strategies include: 1) removing the HTML tags, special characters and stopwords, and 2) recovering abbreviation spellings and truncating the maximum length of user/item documents to 512 words. For a fair comparison, we replace the static word embedding table in several baselines (DeepCoNN, NARRE, DAML, ANR, and UARM) with the pre-trained BERT, which has been proved to be effective in our preliminary results.

Results. Table 2 reports the averaged MSE results over 5 random seeds. In general, DIRECT performs very competitively with the best baselines in all scenarios. Specifically, it significantly performs better than both matrix factorization and language model enhanced methods. Compared with aspect-aware baselines, DIRECT outperforms generally outperform them. Moreover, DIRECT even surpasses 2 of 3 graph-based methods and achieves comparable results with the strongest baseline. These results verify the effectiveness of DIRECT in terms of accuracy.

Discussions. We notice that UARM achieves performance comparable with DIRECT, which initially learns the aspect distributions of users and items with contrastive learning, subsequently integrating these aspect distributions with user/item representations for making recommendations. The strong performance of both UARM and DIRECT validates the idea of modeling the aspect distribution for users and items. However, UARM slightly under-performs compared to DIRECT, which can be attributed to the concurrent learning of users, items, and aspects, where the coding rate reduction is introduced to prevent the collapse of training and guarantee the interpretability and distinctiveness of the learned aspects. Meanwhile, although UARM learns aspect distributions of users and items, it treats them as additional features which are mapped to a latent space and concatenated with user/item embeddings. Moreover, UARM does not provide explicit attribution of prediction. Thus, UARM does not provide an interpretable decision-making process compared to DIRECT.

[1] Yelp Open Dataset: https://www.yelp.com/dataset

Table 2. Recommendation performance comparison.

Model	Toys	Clothing	Games	CDs	Yelp2019	A.R.
BiasMF	$1.054_{\pm 0.061}$	$1.497_{\pm 0.054}$	$1.339_{\pm 0.019}$	$1.024_{\pm 0.007}$	$1.339_{\pm 0.012}$	13.8
NeuMF	$0.935_{\pm 0.006}$	$1.324_{\pm 0.004}$	$1.225_{\pm 0.012}$	$0.949_{\pm 0.006}$	$1.174_{\pm 0.004}$	10.4
DeepCoNN	$0.911_{\pm 0.001}$	$1.297_{\pm 0.010}$	$1.216_{\pm 0.013}$	$0.990_{\pm 0.013}$	$1.172_{\pm 0.006}$	10.0
NARRE	$0.952_{\pm 0.028}$	$1.314_{\pm 0.022}$	$1.236_{\pm 0.012}$	$0.999_{\pm 0.013}$	$1.232_{\pm 0.031}$	12.2
DAML	$0.897_{\pm 0.007}$	$1.275_{\pm 0.011}$	$1.204_{\pm 0.014}$	$0.965_{\pm 0.005}$	$1.160_{\pm 0.011}$	8.8
EMF	$0.906_{\pm 0.005}$	$1.201_{\pm 0.004}$	$1.196_{\pm 0.003}$	OOM[1]	$1.322_{\pm 0.007}$	11.0
ANR	$0.824_{\pm 0.009}$	$1.126_{\pm 0.023}$	$1.190_{\pm 0.097}$	$0.918_{\pm 0.002}$	$1.116_{\pm 0.026}$	5.4
CARP	$0.845_{\pm 0.009}$	$1.081_{\pm 0.012}$	$1.195_{\pm 0.019}$	$1.021_{\pm 0.027}$	$1.143_{\pm 0.007}$	6.6
AARM	$0.848_{\pm 0.001}$	$1.150_{\pm 0.008}$	$1.184_{\pm 0.003}$	$0.951_{\pm 0.005}$	$1.128_{\pm 0.008}$	6.8
UARM	$0.810_{\pm 0.001}$	$1.108_{\pm 0.002}$	$1.118_{\pm 0.003}$	$0.886_{\pm 0.002}$	$1.075_{\pm 0.007}$	3.8
SSG	$0.828_{\pm 0.002}$	$1.129_{\pm 0.012}$	$1.144_{\pm 0.005}$	$0.869_{\pm 0.006}$	$1.205_{\pm 0.005}$	6.4
RMG	$0.808_{\pm 0.002}$	$1.111_{\pm 0.010}$	$1.110_{\pm 0.003}$	$0.859_{\pm 0.004}$	$1.187_{\pm 0.004}$	4.4
RGCL	$0.803_{\pm 0.003}$	$1.103_{\pm 0.009}$	$1.109_{\pm 0.006}$	$0.844_{\pm 0.003}$	$1.179_{\pm 0.004}$	3.0
DIRECT	$0.804_{\pm 0.002}$	$1.100_{\pm 0.010}$	$1.115_{\pm 0.001}$	$0.885_{\pm 0.009}$	$1.063_{\pm 0.011}$	2.4

[1] The model raises an out-of-memory error during training on a 24GB memory GPU.

4.3 Ablation Study

To study **Q2**, we conduct experiments to examine the contributions of 1) using user reviews, 2) fusion network for final user embedding, and 3) the contrastive loss in Section 3.5.2 to capture the shared interests between history reviews and the target review. Specifically, we introduce three DIRECT variants: "w/o Review", "w/o Fusion", and "w/o CL". "w/o Review" is obtained by excluding user reviews from DIRECT. "w/o Fusion" is obtained by replacing the fusion network in Section 3.3.2 with concatenation operation. "w/o CL" is obtained by excluding the contrastive loss function from DIRECT. Table 3 summarizes their results on five benchmark datasets.

Table 3. Ablation study of DIRECT.

	Toys	Clothing	Games	CDs	Yelp2019	average
w/o Review	0.8109	1.1132	1.1199	0.8943	1.0718	1.0020
w/o Fusion	0.8091	1.0939	1.1172	0.8867	1.0683	0.9950
w/o CL	0.8077	1.0954	1.1164	0.8847	1.0674	0.9943
DIRECT	0.8044	1.1004	1.1152	0.8854	1.0628	0.9936

From Table 3, we made three observations. First, DIRECT improves w/o Review in all cases. This result verifies our motivation to capture user interests upon their reviews. Second, compared with w/o Fusion, DIRECT performs better on 4 of 5 datasets. It is reasonable because our fusion network can adaptively combine users' posts and shopping behaviors in a learnable fashion. Third, by enforcing the alignment between review documents and target reviews, DIRECT outperforms w/o CL on Toys and Clothing while performing comparably on the others. The observations above validate the effectiveness of the three crucial components of the proposed model.

4.4 Sensitivity Analysis on Aspect Number

Aspect embeddings are the key to achieve word-level explanation in our system. In this section, we analyze the sensitivity of our model on the number of aspects **Q3**. Specifically, we follow the same experimental setups above and search the optimal number of aspects K in the set $\{1, 3, 5, 7, 9, 11\}$.

Table 4 reports the results with three random seeds. In general, the best K value is varied from one dataset to another in a small range. For example, the optimal K values for Toys, Clothing, Games, CDs, and Yelp2019 are 3,

Table 4. Sensitivity analysis on hyper-parameter K.

K	Toys	Clothing	Games	CDs	Yelp2019
1	0.8083	1.0867	1.1122	0.8852	1.0774
3	**0.8053**	**1.0859**	1.1114	0.8843	**1.0759**
5	0.8060	1.0863	1.1113	0.8797	1.0769
7	0.8062	1.0860	**1.1112**	**0.8766**	1.0791
9	0.8059	1.0859	1.1120	0.8792	1.0766
11	0.8066	1.0864	1.1127	0.8863	1.0790

3, 7, 7, and 3, respectively. That is, the best K value falls between 3 and 7. This observation echos the findings in ANR [9]. Given that DIRECT performs relatively stable when K is between 3 and 7, we set $K = 5$ for all datasets without further specification.

Table 5. Top frequent words for aspects in Clothing dataset.

Gift	Texture	Environment	LowerBody	Material
year	cold	little	shirt	den
watch	soft	day	pair	synthetic
bag	water	old	socks	summer
ear	dark	house	feet	cotton
daughter	second	watch	sole	rubber
sand	strong	socks	run	fan
day	light	pair	pocket	tin
son	thick	wash	side	accent
small	fast	light	bra	cap
gift	gray	warm	back	composite

Table 6. Top frequent words for aspects in Toys dataset.

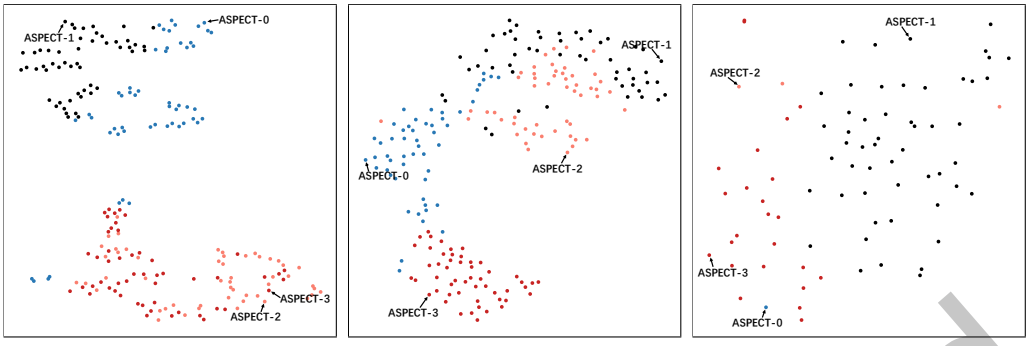
Quailty	Texture	Puzzle	Doll	BoardGame
new	set	piece	doll	game
quality	plastic	make	different	year
collection	hard	game	thing	card
build	train	work	size	car
come	long	time	large	set
beautiful	learn	set	pretty	figure
challenge	big	use	color	pretty
wood	sturdy	together	amazing	look
additional	old	puzzle	heavy	player
grand	young	put	cool	daughter

4.5 Interpretability Analysis

To study **Q4**, we first analyze the performance of our model in learning aspects via visualization and verbalization (section 4.5.1). Then, we quantitatively analyze whether the proposed DIRECT could provide interpretations that reflect the user preferences (section 4.5.2). Finally, we demonstrate the transparent decision making process of DIRECT with some cases (section 4.5.3).

4.5.1 Understanding Learned Aspects.

To check if DIRECT learns discriminative interest aspects, we visualize the words and their aspect associations in Figure 2. Specifically, each word w is represented by a word-aspect embedding vector $\boldsymbol{q}_w$, which is obtained by averaging its word-aspect representations over the entire training set. After we get the word-aspect embeddings, we assign word w to the k-th aspect if $k = \max_k \boldsymbol{q}_w \cdot \tilde{\boldsymbol{a}}_k^\top$, where $\tilde{\boldsymbol{a}}_k$ is the normalized embedding of the k-th



(a) $\gamma_2 = 0.0$ (b) $\gamma_2 = 1e^{-6}$ (c) $\gamma_2 = 1e^{-3}$

Fig. 2. Aspect visualization on Toys dataset with t-SNE [50].

aspect. Since there are tens of thousands words in the vocabulary, we only visualize the top 50 most frequently mentioned words for each aspect.

Figure 2 shows the aspect distributions of our model under three different settings, where different colors denote different aspects. We can observe that, in Figure 2(b), the four aspects (including one residual aspect) are discriminative and linearly separable. However, this character is not solid if we ignore the constraint term Ω_d (Figure 2(a)) or put too much weight on it (Figure 2(c)).

Moreover, we report the top 10 popular words under each learned aspect to examine if they conceptually make sense. In particular, we use the pre-trained PLM to encode words in each user review $\mathcal{T}_{u,i}$, then estimate their aspect associations with Equation (7). Here, we use the checkpoints trained for performance comparison. Table 5 and 6 show the results on the Clothing and Toys datasets. Each column indicates the potential aspect identified by our model. We summarize each aspect in the first line and omit quantifiers, simple sentiment polarity adjectives (e.g., good, bad), and intensity adverbs (e.g., bit, much). As we can see from the tables, our model could effectively cluster words into different aspects that customers may concern about. For example, our model figures out five crucial factors, i.e., Gift, Texture, Environment, LowerBody, and Material, for the clothing domain. Furthermore, the top popular words in each aspect are also closely related. Taking the "Gift" aspect as an example, words like daughter and son are popular roles of the gift receiver in real-world life. In summary, DIRECT can not only identify informative semantic aspects for different domain products but also assign words to their most appropriate aspects automatically.

Table 7. Quantitative analysis to explanation quality.

		Toys	Clothing	Games	CDs	Yelp2019
Baseline-BoW	MSE ↓	$0.936_{\pm 0.013}$	$1.240_{\pm 0.026}$	$1.315_{\pm 0.034}$	$1.027_{\pm 0.006}$	$1.244_{\pm 0.032}$
	Top-K ↑	$0.546_{\pm 0.304}$	$0.517_{\pm 0.320}$	$0.583_{\pm 0.331}$	$0.580_{\pm 0.346}$	$0.500_{\pm 0.270}$
	Last-K ↓	$0.391_{\pm 0.276}$	$0.412_{\pm 0.324}$	$0.357_{\pm 0.330}$	$0.378_{\pm 0.346}$	$0.379_{\pm 0.269}$
	Diff ↑	39.5%	25.5%	63.3%	53.4%	31.9%
DIRECT	MSE ↓	$0.804_{\pm 0.002}$	$1.100_{\pm 0.010}$	$1.115_{\pm 0.001}$	$0.885_{\pm 0.009}$	$1.063_{\pm 0.011}$
	Top-K ↑	$0.526_{\pm 0.282}$	$0.492_{\pm 0.289}$	$0.593_{\pm 0.293}$	$0.525_{\pm 0.321}$	$0.452_{\pm 0.241}$
	Last-K ↓	$0.372_{\pm 0.293}$	$0.399_{\pm 0.313}$	$0.303_{\pm 0.301}$	$0.412_{\pm 0.339}$	$0.382_{\pm 0.273}$
	Diff ↑	41.4%	23.3%	95.7%	27.4%	18.3%

4.5.2 Quantitative Analysis. We quantitatively assess whether our system provides explanations that reflect user preference. In particular, given a user-item pair, we treat the target review written by the user to the item as the

ground truth of the user's preference. We measure the similarity between the DIRECT generated explanations and the target reviews. In this experiment, we consider the top-K sentences from the item document with the greatest maximum interest scores, according to Eq. (5), as the explanations generated by DIRECT, where we set $K = 3$. The semantic similarities between the sentences of explanations and the user reviews are estimated by a fine-tuned semantic similarity estimator based on RoBERTa [38]. This fine-tuned model will return a value between 0 to 1, indicating a stronger semantic similarity between the explanation and the user target review if the value is closer to 1. We further normalized and computed the average scores across the explanation sentences. To ensure the item document covers the user interests, we ignore those user-item pairs with less than ten sentences in the item document. Meanwhile, to guarantee item properties are clearly expressed, we ignore those verbose reviews with over five sentences from a user. For comparison, we also report the similarities between the target review and the last-K sentences. In our experimental design, a recommender demonstrating a higher interpretability could receive a greater average semantic similarity between the top-K sentences and the target review, while the similarity score between the last-K sentences and the target review should be lower. We also calculate the growth percentage of the average similarity of Top-K compared to Last-K, denoted as "Diff". In addition to DIRECT, we implement an inherently interpretable baseline for our analysis of interpretability. This baseline first constructs item feature vectors by counting the frequencies of keywords from item reviews. To perform personalized recommendations, it further estimates which item keywords may interest the given user. The final user rating is predicted with a linear function over selected keyword embeddings. Since this baseline is built on the bag-of-word assumption, a fully transparent and human-understandable decision-making process, it can be considered an oracle in our interpretability analysis experiment. We denote this baseline as BoW.

Table 7 reports the results derived from 5,000 randomly sampled user-item pairs from each dataset. Analysis of these results reveals that the explanations generated by DIRECT exhibit a greater similarity to target reviews compared to those un-selected sentences. This pattern is consistent across all five datasets, demonstrating the efficacy of DIRECT in accurately capturing user interests from item documents, aligning closely with its design objectives. When comparing the interpretability of DIRECT and baseline BoW, we observe that the Diff score of DIRECT is comparable with the ideal interpretable baseline, emphasizing the strong interpretability of DIRECT. However, it is crucial to recognize that BoW achieves this transparent design while sacrificing its recommendation quality. Specifically, we also observe that BoW's MSE is significantly greater than DIRECT's. Putting these together, we conclude that DIRECT simultaneously improves the performance and transparency of recommender systems.

4.5.3 Case Study. We provide case studies to show whether DIRECT improves the transparency of recommendation systems via interpretable features. To this end, we trace the activated user aspects, popular words of activated aspects, and the word sentiments predicted by our model.

Table 8 displays two good recommendation samples (Case 1 and Case 2) and a "bad" one (Case 3). For each case, i.e., user-item predication, we not only report its ground-truth information such as the user/item IDs, rating score r, and target review, but also summarize all related explainable features extracted by DIRECT, including the aspect distribution, activated frequent words and their sentiment polarity, the predicted rating score $\hat{r}$, the predicted preference $pref$ and bias $bias$ scores. To better visualize these cases, we omit contexts that are irrelevant to the target review and highlight the top 20 segments ranked by their activation scores (defined in Equation (5)) among the entire item document. We render the highlighted segments with different colors to emphasize their sentiment polarities (i.e., positive and negative), and underline some segments of the target review to indicate the potential concerns of the anchor user.

Good case. Case 1 and Case 2 list the recommendations of two users on the same item. Since the two cases report the prediction results on the same item, we can observe that our model highlights several common parses in the item document, such as neutral runner, runner, and mid foot striker. However, there still have some parses being only activated by the second user. For example, our model also activates Heavier runners and road training

Table 8. Case study of three user-item pairs coming from the Amazon datasets.

Case 1: userID=A3KHRW6ZC2EQIL, itemID=B006H30KAE (ASICS Men's GEL-Nimbus 14 Running Shoe)	
Prediction:	$r = 5.0$, $\hat{r} = 4.89$, $pref = 0.41$, $bias = 4.48$
Interest Aspect:	$Aspect_1 = 0.5622$, $Aspect_2 = 0.5594$, $Aspect_3 = 0.5676$, $Aspect_4 = 0.5585$, $Aspect_5 = 0.5567$
Item Document:	... Similar to the New Balance 1080 and better than the Brooks Ravena. I am a 192 pound, 51 year old runner. I am a neutral runner and mid foot striker. ... Gel Nimbus may be it, especially as a road training and long distance racing shoe. Heavier runners will really like the plush and cushioned ...
Target Review:	My wife hated the color of the white/blue Nimbus 13s I had ... I'm a neutral shoe guy and I have had multiple heel spur surgeries. ...
Case 2: userID=AOMEH9W6LHC4S, itemID=B006H30KAE (ASICS Men's GEL-Nimbus 14 Running Shoe)	
Prediction:	$r = 5.0$, $\hat{r} = 4.64$, $pref = 0.32$, $bias = 4.32$
Interest Aspect:	$Aspect_1 = 0.4850$, $Aspect_2 = 0.3980$, $Aspect_3 = 0.3982$, $Aspect_4 = 0.4692$, $Aspect_5 = 0.4155$
Item Document:	... Similar to the New Balance 1080 and better than the Brooks Ravena. I am a 192 pound, 51 year old runner. I am a neutral runner and mid foot striker. ... Gel Nimbus may be it, especially as a road training and long distance racing shoe. Heavier runners will really like the plush and cushioned ...
Target Review:	... but I'm quite confident in the fit of ASICs. ... It's neutral (the wrong shoe if you over-pronate) with good lateral stiffness. ...
Case 3: userID=A2DXFI46OKWC8G, itemID=630508985X (Blue Oyster Cult - Live 1976)	
Prediction:	$r = 5.0$, $\hat{r} = 4.06$, $pref = -0.05$, $bias = 4.10$
Interest Aspect:	$Aspect_1 = 0.3172$, $Aspect_2 = 0.9756$, $Aspect_3 = 0.1492$, $Aspect_4 = 0.4661$, $Aspect_5 = 0.9886$
Item Document:	... Bad picture, bad sound, bad performance. Not entirely true. I found the performance to be very good/typical and the picture pretty watchable. I sure wish the sound was better though! ... I do feel a little sorry for people who pay $60-$70 for this disc. I was lucky enough to get it for around $20. ...
Target Review:	... The sound on this isn't bad but its not the greatest so ... its Blue Öyster Cult back in the day, not New Blue Öyster Cult nowadays playing old songs! ...

and long distance racing shoes in the second case. These results seem unreasonable at the first glance, as the two users have the same item target. However, when we trace back, we find that the second user bought another shoe earlier and posted some comments—"These might be the perfect shoes for some runners or race-walkers (perhaps those with slender builds)". By jointly considering the two posts, the reason behind our model in activating Heavier runners is clear and reasonable, since the second user might be a heavier runner. The difference between the two users in the activated parses shed light on the effectiveness of our model in capturing users' personal interests and making interpretable recommendations via extracting human-understandable review words.

"Bad" case. It's impossible for a recommendation model to make correct predictions all the time, our model could fail either. We report an failure case —Case 3 in Table 8. For this case, our model gives a negative preference score (i.e., −0.05) to the item since it believes this user dislikes the sound quality based on the negative sentiments of those highlighted parses such as Bad picture, bad sound, and bad performance. In fact, this prediction is not totally wrong as the user also admits that the sound of this product should be improved based on the target review. However, it might ignore some implicit facts such as the user is a big fan of the band, which makes he/she can tolerate the sound quality to some extent. This kind of conjecture is reasonable since the user gives 5 stars to the product. This running case indicates the limitation of DIRECT in capturing users' fine-grained interests. In the future, we will explore more advanced aspect learning strategy to fill in the gap.

5 RELATED WORK

Recent studies on review-based neural recommendation mainly focus on two topics: 1) improving the accuracy of predicting user preferences and 2) enhancing the interpretability of recommenders.

Review-based Neural Recommender Systems. The earliest successful attempt at the review-based neural recommender is DeepCoNN [69], which uses a Dual-TextCNN [7] architecture to gather user/item embeddings from their reviews. TransNet [2] extents DeepCoNN by forcing the document representations to be similar to the representation of the target user-item review. Inspired by the great success of Transformer [51], MPCN [48], D-Attn [42] and NARRE [4] apply the self-attention mechanism to user and item reviews respectively. To aggregate the information between user and item reviews, DAML [28], CARL [58], AHN [12] exploit the attention mechanism across the two resources. CARP [26] develops a confidence matrix to only keep the embedding of high confidence reviews. AENAR [65] first measures the difference between the current review embedding and a global review embedding, then treats the difference as a gate to filter the review embedding. To better capture the interactions between users and items, RMG [57], SSG [14], and RGCL [44] consider the user/item preference prediction as an edge classification problem and aligned the graph learning methods to aggregate users and items embedding. Recently, researchers [29, 61, 67] directly use pre-trained language models to process reviews or other textual user/item resources for recommendations by leveraging their strong in-context learning ability.

Explainable Review-based Recommender Systems. D-Attn [42] designs a local attention module and a global attention module to find out essential words of reviews. Similarly, CAML [8] first designs a Multi-Pointer Co-Attention Selector to collect a user embedding, a item embedding, and a concept embedding. Then, it uses these embeddings to make recommendations and generate textual explanations. AHR [11] designs an asymmetric attention method to find out important words from the reviews. The user-side attention mechanism extracts words related to the target item. In contrast, the item-side attention mechanism extracts words that most reflect the current item. ANR [9] and CARP [26] are the only two aspect-based end-to-end learning methods in this path. ANR [9] is the first model that applied aspect detection process within the training process. It first represents the item and the user with several aspect embedding and importance scores. The final scoring function is the summation of the similarity of the aspect embeddings weighted by the importance score. CARP [26] predicts specific numbers of aspects by giving the user and item reviews. Next, it combines pairs of aspects from the user and the item and finally uses a capsule network to obtain positive and negative scores.

6 CONCLUSION

We propose a novel self-interpretable review-based recommender system named DIRECT in this study. DIRECT predicts user preferences by averaging the sentiment polarities of words weighted by the word importance. DIRECT assigns more weights to words that express the user's interested aspects. We also leverage the idea of Maximizing Coding Rate Reduction (MCR^2) to encourage the learned aspects to be more discriminate, diverse, and explainable. Under the online system setup, by caching intermediate information such as word-aspect affiliations, DIRECT could achieve linear time complexity with respect to document length. Experimental results on real-world datasets show that DIRECT outperforms traditional baseline methods and is comparable to state-of-the-art methods. Quantiative analysis, visualization and case studies verify the interpretability of DIRECT.

The future works include: (1) exploring more effective user representation learning methods to further improve model performance; (2) developing more effective graph construction methods to describe word-word relationships for generating better aspect embeddings; (3) introducing expert knowledge to construct more controllable representations.

ACKNOWLEDGMENTS

The work is, in part, supported by NSF (#IIS-2223768). The views and conclusions in this paper are those of the authors and should not be interpreted as representing any funding agencies.

REFERENCES

[1] Leon Bottou and Yoshua Bengio. 1994. Convergence properties of the k-means algorithms. Advances in neural information processing systems 7 (1994).

[2] Rose Catherine and William Cohen. 2017. Transnets: Learning to transform for recommendation. In RecSys.

[3] Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. 2021. How interpretable and trustworthy are gams?. In KDD.

[4] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In WWW.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In ICML. PMLR.

[6] Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In CVPR.

[7] Yahui Chen. 2014. Convolutional neural network for sentence classification. Master's thesis.

[8] Zhongxia Chen, Xiting Wang, Xing Xie, Tong Wu, Guoqing Bu, Yining Wang, and Enhong Chen. 2019. Co-Attentive Multi-Task Learning for Explainable Recommendation. In IJCAI.

[9] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based neural recommender. In CIKM.

[10] J Shane Culpepper, Fernando Diaz, and Mark D Smucker. 2018. Research frontiers in information retrieval: Report from the third strategic workshop on information retrieval in lorne (swirl 2018). In SIGIR Forum, Vol. 52. ACM New York, NY, USA, 34–90.

[11] Xin Dong, Jingchao Ni, Wei Cheng, Zhengzhang Chen, Bo Zong, Dongjin Song, Yanchi Liu, Haifeng Chen, and Gerard De Melo. 2020. Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation. In AAAI.

[12] Xin Dong, Jingchao Ni, Wei Cheng, Zhengzhang Chen, Bo Zong, Dongjin Song, Yanchi Liu, Haifeng Chen, and Gerard De Melo. 2020. Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation. In AAAI.

[13] Alessandro Epasto and Bryan Perozzi. 2019. Is a single embedding enough? learning node representations that capture multiple social contexts. In WWW.

[14] Jingyue Gao, Yang Lin, Yasha Wang, Xiting Wang, Zhao Yang, Yuanduo He, and Xu Chu. 2020. Set-sequence-graph: A multi-view approach towards exploiting reviews for recommendation. In CIKM.

[15] Xinyu Guan, Zhiyong Cheng, Xiangnan He, Yongfeng Zhang, Zhibo Zhu, Qinke Peng, and Tat-Seng Chua. 2019. Attentive aspect modeling for review-aware recommendation. ACM Transactions on Information Systems (TOIS) 37, 3 (2019), 1–27.

[16] Xiaotian Han, Zhimeng Jiang, Ninghao Liu, Qingquan Song, Jundong Li, and Xia Hu. 2022. Geometric Graph Representation Learning via Maximizing Rate Reduction. In WWW.

[17] Trevor J Hastie. 2017. Generalized additive models. In Statistical models in S. Routledge, 249–307.

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In CVPR.

[19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In WWW.

[20] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. 7132–7141.

[21] Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. NAACL (2019).

[22] Dmitry Kazhdan, Botty Dimanov, Mateja Jamnik, Pietro Liò, and Adrian Weller. 2020. Now you see me (CME): concept-based model extraction. arXiv preprint arXiv:2010.13233 (2020).

[23] Yunji Kim and Jung-Woo Ha. 2021. Contrastive Fine-grained Class Clustering via Generative Adversarial Networks. In International Conference on Learning Representations.

[24] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept bottleneck models. In ICML. PMLR.

[25] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. Computer (2009).

[26] Chenliang Li, Cong Quan, Li Peng, Yunwei Qi, Yuming Deng, and Libing Wu. 2019. A capsule network for recommendation and explaining what you like and dislike. In SIGIR.

[27] Jiahui Li, Kun Kuang, Lin Li, Long Chen, Songyang Zhang, Jian Shao, and Jun Xiao. 2021. Instance-wise or Class-wise? A Tale of Neighbor Shapley for Concept-based Explanation. In ACMMM.

[28] Donghua Liu, Jing Li, Bo Du, Jun Chang, and Rong Gao. 2019. Daml: Dual attention mutual learning between ratings and reviews for item recommendation. In KDD.

[29] Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. arXiv preprint arXiv:2304.10149 (2023).

[30] Ninghao Liu, Qiaoyu Tan, Yuening Li, Hongxia Yang, Jingren Zhou, and Xia Hu. 2019. Is a single vector enough? exploring node polysemy for network embedding. In KDD.

[31] Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. ICLR (2019).

[32] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. NeurIPS (2017).
[33] Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. 2019. Learning disentangled representations for recommendation. NeurIPS (2019).
[34] Yi Ma, Harm Derksen, Wei Hong, and John Wright. 2007. Segmentation of multivariate mixed data via lossy data coding and compression. IEEE Trans. Pattern Anal. Mach. (2007).
[35] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In KDD.
[36] Deng Pan, Xiangrui Li, Xin Li, and Dongxiao Zhu. 2021. Explainable recommendation via interpretable feature mapping and evaluation of explainability. In IJCAI.
[37] Chanyoung Park, Carl Yang, Qi Zhu, Donghyun Kim, Hwanjo Yu, and Jiawei Han. 2020. Unsupervised differentiable multi-aspect network embedding. In KDD.
[38] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). 3982–3992.
[39] Steffen Rendle. 2010. Factorization machines. In ICDM. IEEE.
[40] David Sculley. 2010. Web-scale k-means clustering. In Proceedings of the 19th international conference on World wide web. 1177–1178.
[41] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In RecSys.
[42] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In RecSys.
[43] Yucheng Shi, Qiaoyu Tan, Xuansheng Wu, Shaochen Zhong, Kaixiong Zhou, and Ninghao Liu. 2024. Retrieval-Enhanced Knowledge Editing for Multi-Hop Question Answering in Language Models. arXiv preprint arXiv:2403.19631 (2024).
[44] Jie Shuai, Kun Zhang, Le Wu, Peijie Sun, Richang Hong, Meng Wang, and Yong Li. 2022. A Review-aware Graph Contrastive Learning Framework for Recommendation. (2022).
[45] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034 (2013).
[46] Peijie Sun, Le Wu, Kun Zhang, Yu Su, and Meng Wang. 2021. An unsupervised aspect-aware recommendation model with explanation text generation. ACM Transactions on Information Systems (TOIS) 40, 3 (2021), 1–29.
[47] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In ICML. PMLR.
[48] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. Multi-pointer co-attention networks for recommendation. In KDD.
[49] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. arXiv preprint arXiv:1908.08962 (2019).
[50] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. JMLR (2008).
[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. (2017).
[52] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. Harv. JL & Tech. (2017).
[53] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In CIKM.
[54] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In KDD.
[55] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In SIGIR.
[56] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled graph collaborative filtering. In Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. 1001–1010.
[57] Chuhan Wu, Fangzhao Wu, Tao Qi, Suyu Ge, Yongfeng Huang, and Xing Xie. 2019. Reviews meet graphs: enhancing user and item representations for recommendation with hierarchical attentive graph neural network. In EMNLP-IJCNLP.
[58] Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo. 2019. A context-aware user-item representation learning for item recommendation. TOIS (2019).
[59] Xuansheng Wu, Wenlin Yao, Jianshu Chen, Xiaoman Pan, Xiaoyang Wang, Ninghao Liu, and Dong Yu. 2023. From language modeling to instruction following: Understanding the behavior shift in llms after instruction tuning. arXiv preprint arXiv:2310.00492 (2023).
[60] Xuansheng Wu, Haiyan Zhao, Yaochen Zhu, Yucheng Shi, Fan Yang, Tianming Liu, Xiaoming Zhai, Wenlin Yao, Jundong Li, Mengnan Du, et al. 2024. Usable xai: 10 strategies towards exploiting explainability in the llm era. arXiv preprint arXiv:2403.08946 (2024).
[61] Xuansheng Wu, Huachi Zhou, Yucheng Shi, Wenlin Yao, Xiao Huang, and Ninghao Liu. 2024. Could Small Language Models Serve as Recommenders? Towards Data-centric Cold-start Recommendations. In The Web Conference (WWW).
[62] Fan Yang, Ninghao Liu, Suhang Wang, and Xia Hu. 2018. Towards interpretation of recommender systems with sorted explanation paths. In ICDM. IEEE.

[63] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In KDD.
[64] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. 2020. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. NeurIPS (2020).
[65] Tianwei Zhang, Chuanhou Sun, Zhiyong Cheng, and Xiangjun Dong. 2022. AENAR: An aspect-aware explainable neural attentional recommender model for rating predication. Expert Syst. Appl. (2022).
[66] Yongfeng Zhang, Xu Chen, et al. 2020. Explainable recommendation: A survey and new perspectives. Foundations and Trends® in Information Retrieval (2020).
[67] Yuhui Zhang, Hao Ding, Zeren Shui, Yifei Ma, James Zou, Anoop Deoras, and Hao Wang. 2021. Language models as recommender systems: Evaluations and limitations. In I (Still) Can't Believe It's Not Better! NeurIPS 2021 Workshop.
[68] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. 83–92.
[69] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In WSDM.
[70] Honglei Zhuang, Xuanhui Wang, Michael Bendersky, Alexander Grushetsky, Yonghui Wu, Petr Mitrichev, Ethan Sterling, Nathan Bell, Walker Ravina, and Hai Qian. 2021. Interpretable ranking with generalized additive models. In WSDM.