A Context-Aware User-Item Representation Learning for Item Recommendation

LIBING WU, CONG QUAN, CHENLIANG LI, and QIAN WANG, Wuhan University, China
BOLONG ZHENG, Huazhong University of Science and Technology, China
XIANGYANG LUO, State Key Lab of Mathematical Engineering and Advanced Computing, China

Both reviews and user-item interactions (i.e., rating scores) have been widely adopted for user rating prediction. However, these existing techniques mainly extract the latent representations for users and items in an independent and static manner. That is, a single static feature vector is derived to encode user preference without considering the particular characteristics of each candidate item. We argue that this static encoding scheme is incapable of fully capturing users' preferences, because users usually exhibit different preferences when interacting with different items. In this article, we propose a novel **c**ontext-**a**ware user-item **r**epresentation **l**earning model for rating prediction, named CARL. CARL derives a joint representation for a given user-item pair based on their individual latent features and latent feature interactions. Then, CARL adopts Factorization Machines to further model higher order feature interactions on the basis of the user-item pair for rating prediction. Specifically, two separate learning components are devised in CARL to exploit review data and interaction data, respectively: *review-based feature learning* and *interaction-based feature learning*. In the review-based learning component, with convolution operations and attention mechanism, the pair-based relevant features for the given user-item pair are extracted by jointly considering their corresponding reviews. However, these features are only reivew-driven and may not be comprehensive. Hence, an interaction-based learning component further extracts complementary features from interaction data alone, also on the basis of user-item pairs. The final rating score is then derived with a dynamic linear fusion mechanism. Experiments on seven real-world datasets show that CARL achieves significantly better rating prediction accuracy than existing state-of-the-art alternatives. Also, with the attention mechanism, we show that the pair-based relevant information (i.e., context-aware information) in reviews can be highlighted to interpret the rating prediction for different user-item pairs.

CCS Concepts: • **Information systems → Recommender systems**; *Web searching and information discovery*; • **Computing methodologies → Neural networks**;

Additional Key Words and Phrases: Rating prediction, neural networks, recommendation systems

This research was supported by National Natural Science Foundation of China (No. 61872278, No. 61502344, No. 61472287, No. 61772377), Natural Scientific Research Program of Hubei Province (No. 2017CFB502, No. 2017CFA007), Natural Scientific Research Program of Wuhan University (No. 2042017kf0225, No. 2042016kf0190), Academic Team Building Plan for Young Scholars from Wuhan University (No. Whu2016012).

Authors' addresses: L. Wu, and C. Quan, Wuhan University, School of Computer Science, Wuhan, China; emails: {wu, quancong}@whu.edu.cn; C. Li (corresponding author), and Q. Wang, Wuhan University, Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan, China; emails: {cllee, qianwang}@whu.edu.cn; B. Zheng, Huazhong University of Science and Technology, School of Computer Science, China; email: zblchris@gmail.com; X. Luo, State Key Lab of Mathematical Engineering and Advanced Computing, China; email: xiangyangluo@126.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2019 Association for Computing Machinery.
1046-8188/2019/01-ART22 $15.00
https://doi.org/10.1145/3298988

ACM Reference format:
Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo. 2019. A Context-Aware User-Item Representation Learning for Item Recommendation. *ACM Trans. Inf. Syst.* 37, 2, Article 22 (January 2019), 29 pages.
https://doi.org/10.1145/3298988

1 INTRODUCTION

Many social media websites and ecommerce systems allow users to write reviews to express their personal opinions on the consumed items, along with a rating score indicating their preferences. The rich information covered in the reviews can reveal the characteristics of the items and also the preferences of each individual user. Exploiting reviews has proved to be effective for better rating prediction performance. Moreover, it is beneficial in alleviating the data sparsity and cold-start issues for recommender systems. Many existing works utilize both review data and user-item interaction data to further enhance recommendation performance [2, 24, 34, 40].

Prior works resort to Latent Dirichlet Allocation (LDA) [3] or Non-negative Matrix Factorization (NMF) [21] to derive latent features over the reviews [2, 24, 34, 40]. These techniques achieve better recommendation performance than conventional latent models solely based on the user-item interactions (i.e., rating scores). However, one intrinsic limitation within these techniques is the Bag-of-Words (BOW) representation for review processing. That is, the semantic contextual information encoded in local word context is ignored.

Recently, neural network-based models have been proposed to derive latent features from the reviews for rating prediction [5, 17, 45]. In these works, the Convolutional Neural Network (CNN) architecture is employed to extract the user or item latent features from the corresponding user and item reviews,[1] respectively. By using dense word embedding representations and a local window to capture the contextual information, the CNN-based techniques facilitate a better semantic understanding of reviews, leading to significant improvement over the existing BOW-based methods for rating prediction. Nevertheless, people find that it is difficult to understand the features extracted by neural networks and this therefore limits the interpretability of recommender systems. To address this problem, Seo and Huang [30] first incorporate an attention mechanism to render a more interpretable model for rating prediction.

However, all existing BOW-based methods, CNN-based methods, or attention-based methods derive user and item latent feature vectors from the reviews in an independent and static manner. A single static latent vector is assigned for a user to estimate her rating score on a candidate item whose latent vector is also static. That is, user and item latent vectors are learned during the training phase without considering the interaction between the user and the item. It is intuitive that not all words in the reviews written by a user are relevant to her rating on a particular item. For example, the critical defects mentioned by a user in her comment about an inferior-quality product would be useless to guess her preference on many other high-quality products. Identifying the relevant semantic information by jointly considering both the reviews of the user and item could be a new avenue to improve prediction accuracy. Inspired by this idea, we take a closer look at conventional Collaborative Filtering (CF) techniques—mainly factor learning techniques (e.g., matrix factorization). It is surprising that similar observations are made. Existing CF methods mainly derive static latent feature vectors for each user and item by matching their past interaction data. Then the rating prediction is calculated by considering only the linear combination of the latent featuers through a Dot-Product (DP) operation. It is expected that learning a joint

[1] A user review refers to all reviews written by the user. Similarly, an item review refers to all reviews written for it.

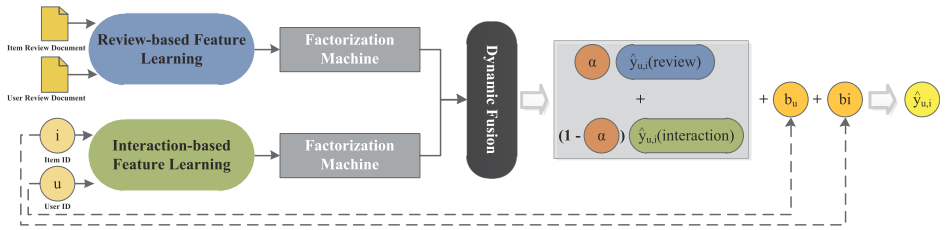


Fig. 1. The architecture of CARL.

and pair-dependent representation for a user-item pair based on both the individual characteristics and their interactions would benefit the rating prediction of them. Here, we term this feature learning for a user-item pair as *context-aware* feature learning. We argue that context-aware information extracted by modeling the latent feature interactions between users and items is more expressive and discriminative to understand users' complex rating behaviors. Moreover, it is easy to further facilitate the modeling of the higher order feature interactions over the context-aware latent features for a user-item pair, and this leads to better prediction performance.

To this end, in this article, we propose a **c**ontext-**a**ware user-item **r**epresentation **l**earning model for rating prediction by integrating information from both textual reviews and rating scores, named CARL. Figure 1 illustrates the overall network architecture for CARL. CARL consists of two independent feature-learning components: *review-based feature learning* and *interaction-based feature learning*. In review-based feature learning, CARL utilizes the convolution operations and an attention mechanism to highlight the pair-based relevant semantic information by jointly considering the reviews written by a given user, and the reviews written for a given item. Then, an abstracting layer is employed to derive the refined latent feature vectors for the user and the item, respectively. In other words, the latent feature vector for a given user is context-aware when facing different items. The representation of the user-item pair is then constructed by further considering the latent feature interactions. Note that the semantic information contained in reviews could only serve as a partial reflection of the user rating behavior. Existing interaction-based latent models have delivered promising rating prediction performance. We believe that the interaction data alone could provide complementary knowledge to complete the picture. Therefore, in interaction-based feature learning, CARL constructs another joint representation for a user-item pair based on another set of feature vectors and their interactions. Then, CARL feeds the latent representation learned by each component into a Factorization Machine (FM) [27] to further model higher order feature interactions for rating prediction. Finally, a novel dynamic weighted linear fusion is devised to perform the final rating prediction.

We conduct comprehensive experiments over seven real-world datasets with reviews. The experimental results validate that the two feature learning components complement each other and result in better prediction performance when fused. Also, the proposed CARL significantly outperforms the existing state-of-the-art alternatives across seven datasets. In summary, the contributions made in this area are listed as follows:

- We propose a neural network-based context-aware user-item representation learning model for rating prediction. The model derives pair-dependent latent representations on the basis of user-item pairs instead of learning a static user/item latent representation for rating prediction. To the best of our knowledge, CARL is the first work that learns context-aware representations for each user-item pair based on their individual characteristics and their interactions together by exploiting both textual reviews and user-item interaction data.

- With the neural attention mechanism introduced in review-based feature learning, we can identify pair-based relevant information based on semantic interaction between a user review document and an item review document. In this sense, we are the first to extract context-aware information from reviews on the basis of the user-item pair via neural network techniques. The attention mechanism also facilitates the explainability of the recommendation. Moreover, a simple but novel dynamic linear fusion strategy is proposed in CARL to aggregate the evidences from the two components for final rating prediction.
- Through extensive experiments conducted on seven real-world datasets, the results demonstrate that our proposed CARL consistently achieves better rating prediction accuracy than existing state-of-the-art alternatives, including slap-up BOW-based methods, CNN-based methods, and attention-based methods. Further case studies demonstrate that the words highlighted by CARL in reviews are very meaningful and uncover users' specific preferences toward an item of interest, which helps to improve explainability for the recommendation.

The remainder of this article is organized as follow: Related works are reviewed in Section 2. Secion 3 introduces the overall framework of CARL in detail. The experimental settings and results are presented in Section 4. Finally, Section 5 concludes the article and discusses future works.

2 RELATED WORK

Our work is related to the studies of interaction-based CF, text-based rating prediction, deep neural network-based recommendation, and context-based feature learning. Therefore, in this section, we briefly review the relevant literatures in these areas.

2.1 Interaction-Based Collaborative Filtering

The interaction based recommendation methods are mainly based on CF techniques [33], which aim to represent users and items with latent feature vectors. Matrix Factorization (MF) is the most popular technique in this line of literatures. Basic MF models [20, 28] try to learn users' and items' latent features purely by matching the user-item interaction (i.e., binary indicators or user-item rating scores) matrix with a DP operation. The rating prediction is then calculated also by using a DP operation with the derived user/item latent features for a given user-item pair. Plenty of works try to enhance the performance of MF by modeling more information based on user-item interactions. For example, Koren et al. [20] introduce user and item biases into MF. Koren [19] integrates a neighborhood model into MF that assumes that a user's rating on an item is formed not only by the latent characteristics of the user-item pair, but also by the user's rating behaviors on other items. All these methods have been validated to outperform the vanilla MF model in many domains. However, all these MF-based models use the DP operation as their rating predictor. One inherent limitation of the DP operation is that latent features are independent of each other. That is, DP only enables linear combination of latent features without considering higher order feature interaction. It is validated that the performance of existing MF-based methods is hindered by this strong constraint [13].

With the tremendous successes of neural networks in many fields, some researchers turn to using neural networks to learn users' rating behaviors. Salakhutdinov et al. [29] propose a Restricted Boltzmann Machines (RBM)-based model for rating prediction. It applies separate share-parameter RBMs, each of which has visible softmax units for the rated item, to model the interaction history of each user. Hence, when two users have similar rating behaviors, the prediction for them is similar as well. Neighborhood information of users and items is then utilized to extend the RBM model for better prediction performance [11]. The principle underlying these two pioneering works is still

conventional user-based and item-based CF. He et al. [13] present a general deep neural framework for collaborative filtering with implict feedback. The proposed NeuMF takes the static user and item feature vectors as input and calculates the rating score by replacing the DP operation with a neural atchitecture. The empirical study shows that NeuMF achieves superior performance over the existing latent factor learning techniques. NeuMF models latent feature interactions between users and items through a deep MLP architecture with nonlinearity. However, higher order feature interactions still cannot be well captured by NeuMF. In contrast, the proposed CARL derives a context-aware representation for a user-item pair and uses FM to model the higher order latent feature interactions for rating prediction. Both the complex interactions between users and items as well as the higher order latent feature interactions are well captured by CARL to model users' diverse rating behaviors on different items. Moreover, since NeuMF is devised for CF with implicit feedback, its applicability to rating prediction as well as its incorporation with review data is still unknown.

2.2 Rating Prediction from Text

Although CF methods [9, 20, 29] based on user-item interactions are prevalent in the past decades, they have two obvious limitations. First, the prediction accuracy of most CF methods drops significantly when the data are sparse. Second, they are incapable of handling new users and items (i.e., cold-start issue). Textual information (e.g., users' reviews, item description or labels) is the most popular auxiliary information available in many recommender systems. Consequently, exploiting textual information to address these inherent limitations has become a hot research topic.

Some researchers [2, 22, 24, 34, 40] propose to employ topic modeling techniques to learn latent topic factors from text. HFT [24] and CTR [40] employ an LDA-like technique to exploit latent topics from review text. RBLT [34] employs similar techniques to uncover topic features from rating-boost review text as latent factors. The authors assume that more recommendable features would be contained in a higher rated review. Thus, they construct the rating-boost review text of an original review by repeating the review r times, where r is the rating score associated with it. In this sense, topic models like LDA can easily extract these preferred features as latent topic factors. These latent topic fators are later integrated into an MF framework to derive item characteristics. RMR [22] uses a similar technique to derive topic factors from text but using Gaussian mixtures to model ratings instead of an MF framework. TopicMF [2] uses the MF technique to jointly model the interaction data and topics from review text. In their transform function, these authors use a linear combination of the latent factors of users and items to transform the latent topic in the reviews. CDL [41] tightly couples SADE [37] over the text information and PMF [28] for the implicit rating matrix. The deep neural structure enables CDL to learn interpretable latent factors from the text. These methods outperform models that solely rely on user-item interaction data. However, these methods all belong to the category of BOW models which ignore word order and local context information. Hence, much concrete information in the form of phrases and sentences is lost through this coarse-grained text processing strategy.

To tackle this limitation, several methods endowing the MF framework with a neural treatment [17, 44] are proposed. CMLE [44] leverages an embedding-based model to integrate a word embedding model with a standard MF model to accommodate contextual information for words in the reviews. A CNN-based neural network model (named ConvMF) is proposed by Kim et al. [17]. ConvMF utilizes a CNN network to obtain better latent semantic representations from textual reviews by considering word order and local context. Zheng et al. [45] use a parallel CNN model (named DeepCoNN) to separately derive the latent features of users and items based on their reviews. Then they concatenate the latent features of the corresponding user and item and feed the resultant vector into an FM for rating prediction. TransNets [5] extends DeepCoNN by

adding an additional layer (target network) to learn the representation of a target user-target item review at training time and then using the learned represention to regularize the output of the source network. The source network therefore can mimic the latent representation of a target review, yet it is not available at test time. TransNets gains improvement in rating prediction against DeepCoNN. These recent methods perform better than the existing BOW-based methods and are proved to be effective in alleviating cold-start and data-sparsity issues. Despite these significant improvements on recommendation performance, these works solely derive the latent feature vectors of users and items in a static and separate manner which neglects the diverse and complex interactions between users and items.

2.3 Attention-Based Deep Recommender System

As discussed earlier, neural network-based techniques have been widely applied to recommender systems. However, it is difficult to output meaningful patterns to help interpret the recommendation decisions due to their well-known black-box property. Recently, several works have been proposed to discriminate the importance of each latent feature or factor to enhance recommendation accuracy, drawing on the attention mechanism recently proposed in the neural networks area [1]. With an attention mechanism, we can identify the important words from textual auxiliary information and therefore provide a way to offer semantic interpretation for recommendations. D-attn [30] combines local and global attention on review text and produces weighted text. The weighted text is later passed to a CNN model to derive better learned representations of users and items. Chen et al. [6] propose an attention-based deep learning model (NARR) to compute the usefulness of reviews for recommendation. NARR can provide a review-level explanation for rating prediction. TARMF [23] employs an attention-based GRU network to interpret the semantic meaning associated with each latent factor in the latent vectors gained by an MF model. It outperforms ConvMF on different publicly available rating prediction datasets. In multimedia recommendation, in order to better characterize users' preference, ACF [7] employs two attentive modules which learn to select the informative components of multiple items and representative items from users' purchased records, respectively. ACF incorporates the attentive modules into classic CF models with implicit feedback [14, 26]. Other than applying attention to derive the importance of item components, DAMD [42] leverages an attention model to adaptively incorporate multiple prediction models based on their suitability for article recommendation. Though these existing attention-based recommender systems improve recommendation performance and also the interpretability of recommender systems, they also neglect the diverse and complex interactions between users and items. Here, we utilize an attention mechanism to identify the pair-based relevant semantic information by jointly considering both the user and item reviews. Differing from all the text-based and attention-based methods mentioned earlier, the proposed CARL extracts the latent features for a user-item pair based on both their individual characteristics and their interactions. The incorporation of FM further models the higher order latent feature interactions for better rating prediction. Our experimental results show that CARL achieves much better rating prediction performance than existing state-of-the-art alterantives.

2.4 Context-Based Feature Learning

Attention-based neural networks have been shown to be effective in improving performance in many tasks [1, 38, 39, 43]. Many recent works have utilized the attention mechanism to learn context-aware latent representations from textual information. For example, several works [10, 38, 39, 43] propose to utilize an attention layer to learn context-based representation for question and answer matching, and thus targeted answers can be formed for different questions. CANE [36] uses a similar structure to learn adaptive network embeddings and yields better performance

than the static embedding-based methods. Similar to these techniques, CARL is devised to jointly infer the context-aware latent features based on both the review data and user-item rating scores. To the best of our knowledge, we are the first to introduce context-aware representation learning on the basis of user-item pairs for rating prediction.

3 THE PROPOSED MODEL

In this section, we present CARL, a **c**ontext-**a**ware user-item **r**epresentation **l**earning model for item recommendation. As demonstrated in Figure 1, CARL is devised to estimate the personalized rating score for a new user-item pair by exploiting two heterogenous information sources: *item reviews* written by users and *user-item interaction matrix*. Hence, CARL consists of two independent feature learning components: *review-based feature learning* and *interaction-based feature learning*. In the following sections, we first detail the rating prediction framework for CARL, followed by a description about the two learning components.

3.1 Rating Prediction Framework

The DP operation is often used by existing works for rating prediction [28, 44, 45]. However, DP holds a strong constraint in that the latent dimensions are independent of each other. That is, each dimension in a latent user vector could only interact with the corresponding dimension in the latent item vector. This independence constraint is incapable of learning complex user-item rating behaviors through higher order feature interactions. Since the proposed CARL derives a context-aware representation for each user-item pair, it is desirable to model higher order latent feature interactions to better understand the rating behaviors. Hence, in this work, we choose an FM [27] to calculate the rating score. Specifically, given a latent feature vector learned for a user-item pair, denoted as $\mathbf{z}_{u,i}$, FM calculates the corresponding rating score as follows:

$$\hat{y}_{u,i}(\mathbf{z}_{u,i}) = m_0 + \mathbf{m}^{\mathrm{T}}\mathbf{z}_{u,i} + \frac{1}{2}\mathbf{z}_{u,i}^{\mathrm{T}}\mathbf{M}\mathbf{z}_{u,i} \tag{1}$$

$$\mathbf{M}_{j,k} = \mathbf{v}_j^{\mathrm{T}}\mathbf{v}_k, j \neq k \tag{2}$$

where m_0 is the global bias, $\mathbf{m}$ is the coefficent vector for latent feature vector $\mathbf{z}_{u,i}$, $\mathbf{M}$ is the weight matrix for second-order interactions and its diagonal elements are 0 (i.e., $\mathbf{M}_{j,j} = 0$), and $\mathbf{v}_j, \mathbf{v}_k \in \mathbb{R}^v$ are the v-dimensional latent vectors associated with dimension j and k of $\mathbf{z}_{u,i}$. From Equation (1), we can see that both first- (i.e., $\mathbf{m}$) and second-order (i.e., $\mathbf{M}$) feature interactions are utilized for rating prediction. FM is also applied for rating prediction in other studies [19, 45]. We believe capturing higher order feature interactions is important for modeling complex user-item rating behaviors. We have also tried using other rating prediction formulas like Linear Regression (LR) and Multiple-Layer Perception (MLP) [13]. However, our results show that FM yields better prediction accuracy than LR and MLP (ref. Section 4.3).

In reality, rating behaviors contain multiple inherent tendencies, known as *bias*. For example, some users tend to rate a higher score for all items. And some items are likely to recieve higher ratings from all users [20]. Smola [31] proved that incorporating user and item biases can accommodate rating variations well and hence improve the performance of rating prediction. Following these works, we modify the rating predictor by adding both user and item biases as follows:

$$\hat{y}_{u,i} = \hat{y}_{u,i}(\mathbf{z}_{u,i}) + b_u + b_i, \tag{3}$$

where b_u and b_i are the corresponding bias for user u and item i, respectively. $\hat{y}_{u,i}$ is the predicted rating. We take the square loss as the objective function for parameter optimization.

$$J_{sqr} = \sum_{(u,i)\in O} (y_{u,i} - \hat{y}_{u,i})^2 + \lambda_\Theta \|\Theta\|^2, \tag{4}$$

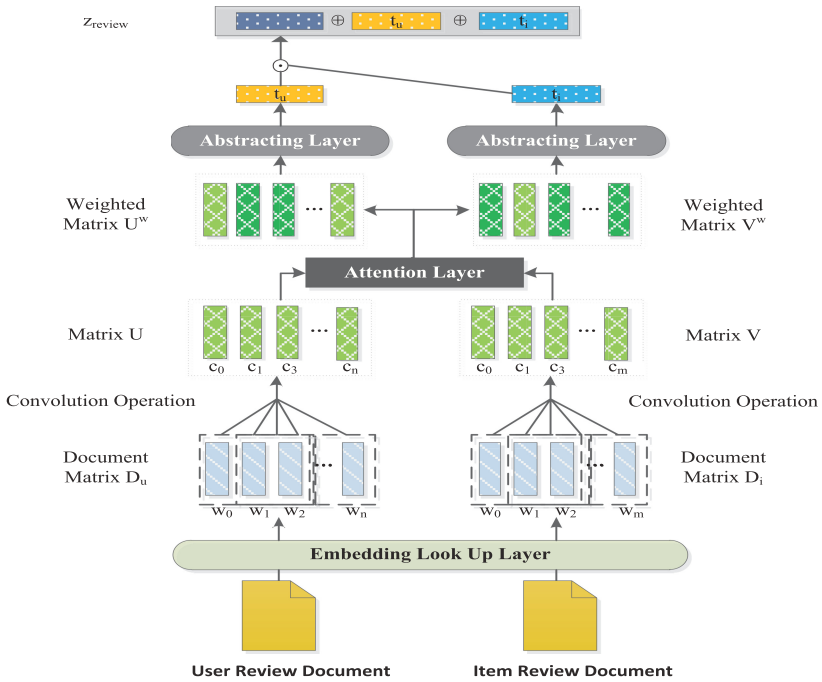


Fig. 2. The network architecture of the review-based feature learning.

where O denotes the set of observed user-item rating pairs, $y_{u,i}$ is the observed rating score for user u on item i, and Θ denotes all the parameters. The second term of Equation (4) is used as regularization to prevent the model from overfitting.

3.2 Review-Based Feature Learning

Since the reviews made by a user reflect her preference, we take all the reviews written by the same user to form a single document as a user review document. Similarly, we merge all the reviews made by the users for an item as the item review document. It is expected that the semantic information covered in two kinds of review documents are quite different. While a user review document would contain more personal preference, an item review document is mainly comprised of different aspects focused on by all the relevant users. The task of review-based feature learning in CARL is to infer a customized latent feature vector for each user-item pair by jointly considering their review documents.[2] The convolution operation has been successfully adopted in many natural language processing and information retrieval tasks, such as document representation learning [8, 15, 16, 18]. Specifically, we use the convolution operation to extract different aspects covered by the review documents. Then we utilize an attentive layer to highlight the pair-based relevant aspects by considering the interaction between the review documents of user and item. Finally, an abstracting layer is utilized to derive the final latent feature vector for the user-item pair. Figure 2 demonstrates the network architecture for the review-based feature learning.

Convolution Layer. Given a review document $D = (w_1, w_2, \ldots, w_n)$, an embedding look-up layer first projects each word to its corresponding embedding $\mathbf{w}_i \in \mathbb{R}^{1\times t}$. Then a document matrix $\mathbf{D} \in \mathbb{R}^{nt\times 1}$ is formed by concatenating these embeddings in the order of their appearances in the

[2]When both user and item review documents are applicable, we use *review document* instead for simplicity.

document:

$$\mathbf{D} = [\cdots \mathbf{w}_{j-1}, \mathbf{w}_j, \mathbf{w}_{j+1} \cdots]^{\mathrm{T}}$$

where $\mathbf{w}_j$ is word embedding of the word at the jth position in document D. That is, the order of words is preserved in matrix $\mathbf{D}$ and in turn enables the convolution layer to extract more accurate semantic information compared to BOW techniques [8]. Specifically, a convolution filter f_j over a sliding window of size s is used to extract the contextual feature c_h^j from the local context. To be specific:

$$c_h^j = \sigma\left(\mathbf{D}_{h-\frac{s-1}{2}:h+\frac{s-1}{2}}\right), \tag{5}$$

where σ is the nonlinear activation function, $\mathbf{W}_c^j$ is the convolution weight vector for filter j, and $\mathbf{D}_{h-\frac{s-1}{2}:h+\frac{s-1}{2}}$ is the slice of matrix $\mathbf{D}$ within the sliding window centering at hth position: $\mathbf{D}_{h-\frac{s-1}{2}:h+\frac{s-1}{2}} = [\mathbf{w}_{h-s}, \ldots, \mathbf{w}_{h+s}]$. Without explicit specification, we opt for Rectified Linear Unit (ReLU) as the activation function, ($ReLU(x) = \max(x, 0)$). Here, we pad $\frac{s-1}{2}$ zero vectors at the begining and end of document matrix $\mathbf{D}$ to produce n contextual features, where n is the length of document D. We use multiple convolution filters with different convolution weight vectors to extract the contextual features for each word with its local context (i.e., s consecutive words). In this work, we use two different $\mathbf{W}_c^*$ for processing the user review document and item review document, respectively.

Attentive Layer. As stated earlier, we assume that a user review document could contain more personal and different preferences on different items. Likewise, an item review document might consist of different aspects focused on by different users. In other words, not all information contained in the review documents could be useful for inferring the rating score for a specific user-item pair. To effectively capture useful information on the basis of given user-item pair, we employ an attention layer to jointly process the review documents of the corresponding user-item pair.

After applying the convolution operation for a review document, we can form a contextual feature vector $\mathbf{c}_h$ for the word at hth position in the document:

$$\mathbf{c}_h = [c_h^1, \ldots, c_h^f], \tag{6}$$

where c_h^f is the contextual feature calculated by convolution filter f for the hth word by using Equation (5). Vector $\mathbf{c}_h$ is thus the contextual feature vector of the hth word after the convolution layer. In this way, we can form two contextual matrices for the user and item review documents, respectively:

$$\mathbf{U} = [\mathbf{c}_1^u; \ldots; \mathbf{c}_n^u] \tag{7}$$

$$\mathbf{V} = [\mathbf{c}_1^i; \ldots; \mathbf{c}_m^i], \tag{8}$$

where $\mathbf{c}_j^u$ and $\mathbf{c}_k^i$ are the contextual feature vectors based on Equation (6) for the jth word and the kth word in the user and item review documents, respectively, n and m are the lengths of the user and item review documents, respectively.

Inspired by the work of others [10, 36], we utilize an attentive matrix, $\mathbf{T} \in \mathbb{R}^{f \times f}$, to derive the importance of each contextual feature vector for both $\mathbf{U}$ and $\mathbf{V}$. In detail, we project matrix $\mathbf{U}$ and $\mathbf{V}$ into the same latent space and calculate the pair-wise relatedness between each pair of contextual feature vectors $\mathbf{c}_j^u$ and $\mathbf{c}_k^i$ as follows:

$$R_{j,k} = \tanh(\mathbf{c}_j^{u\mathrm{T}} \mathbf{T} \mathbf{c}_k^i), \tag{9}$$

where $R_{j,k}$ is the relatedness between $\mathbf{c}_j^u$ and $\mathbf{c}_k^i$, and tanh is the hyperbolic tangent function.

Based on Equation (9), a row $\mathbf{R}_{j,*}$ contains the relatedness scores between the contextual feature vector $\mathbf{c}_j^u$ in $\mathbf{U}$ and all the contextual feature vectors in $\mathbf{V}$. Similarly, a column $\mathbf{R}_{*,k}$ contains the

relatedness scores between the contextual feature vector $\mathbf{c}_k^i$ in $\mathbf{V}$ and all the contextual feature vectors in $\mathbf{U}$. A mean-pooling operation is then applied to each row/column of $\mathbf{R}$ as follows:

$$g_j^u = \mathbf{mean}(R_{j,1}, \ldots, R_{j,m}) \tag{10}$$

$$g_k^i = \mathbf{mean}(R_{1,k}, \ldots, R_{n,k}). \tag{11}$$

Based on the mean relatedness calculated in Equations (10) and (11), we can highlight the importance of each contextual feature vector in $\mathbf{U}$ and $\mathbf{V}$, respectively:

$$a_j^u = \frac{\exp(g_j^u)}{\sum_h^n \exp(g_h^u)} \tag{12}$$

$$a_k^i = \frac{\exp(g_k^i)}{\sum_h^m \exp(g_h^i),} \tag{13}$$

where a_j^u and a_k^i are the attentive weights of $\mathbf{U}_{j,*}$ and $\mathbf{V}_{k,*}$ respectively. At last, we obtain the pair-based attentive weight vectors $\mathbf{a}^u$ and $\mathbf{a}^v$ for $\mathbf{U}$ and $\mathbf{V}$ via the attentive layer:

$$\mathbf{a}^u = [a_1^u, \ldots, a_n^u] \tag{14}$$

$$\mathbf{a}^i = [a_1^i, \ldots, a_m^i]. \tag{15}$$

Here, $\mathbf{a}^u$ and $\mathbf{a}^i$ can be regarded as the learned distribution of the degree of importance to the words in a user review document and an item review document, respectively. Note that, $\mathbf{a}^*$ is computed under the consideration of the interaction between $\mathbf{U}$ and $\mathbf{V}$. Namely, the value in $\mathbf{a}^*$, indicating the importance of corresponding words in the user/item review document, depends on the user-item pairs.

Abstracting Layer. We obtain a weighted $\mathbf{U}$ and $\mathbf{V}$ based on the attentive vectors $\mathbf{a}^u$ and $\mathbf{a}^i$ as follows:

$$\mathbf{U}^w = \mathrm{diag}(\mathbf{a}^u)\mathbf{U} \tag{16}$$

$$\mathbf{V}^w = \mathrm{diag}(\mathbf{a}^v)\mathbf{V}, \tag{17}$$

where $\mathrm{diag}(\mathbf{a}^*)$ is the diagonal matrix whose diagonal are elements in vector $\mathbf{a}^*$. Recall that the attentive weights are calculated based on both the user review document and the item review document; thus, a large weight indicates that the corresponding contextual feature vector is more relevant to the given user-item pair. In this sense, we can consider these highly weighted contextual vectors as relevant aspects covered in the corresponding review documents for the user-item pair only.

At this step, we can simply sum up the weighted contextual feature vectors to represent the user/item under consideration, following the work of others [10, 36]. However, a simple weighted average could introduce too much noisy information since irrelevant aspects covered in both user and item review documents could account for a major proportion. Here, we choose to stack further neural transformations to extract higher level semantic features based on $\mathbf{U}^w$ and $\mathbf{V}^w$, respectively. To accommodate the noise and irrelevant aspects extracted earlier, we employ a mean-pooling CNN network to further abstract higher-level features h_a^u as follows:

$$\begin{aligned} h_h^j &= \sigma\left(\mathbf{W}_a^j \mathbf{U}_{h:h+s-1}^w\right) \\ h_j &= \mathbf{mean}\left(h_1^j, \ldots, h_n^j\right) \\ \mathbf{h}^u &= [h_1, \ldots, h_f], \end{aligned} \tag{18}$$

where $\mathbf{W}_a^j$ is the convolution weight vector for filter j, and a sliding window of size s is used. A similar process is applied to extract higher level feature vector $\mathbf{h}^i$ from $\mathbf{V}^w$. There is one straightforward merit for applying a CNN network over $\mathbf{U}^w$ and $\mathbf{V}^w$. Note that all the contextual feature vectors in $\mathbf{U}$ and $\mathbf{V}$ are extracted based on the convolution operation with a local context window. A further convolution operation inside the CNN network could cover a larger context for latent feature extraction, but with relevance-weighted information (i.e., less noisy information). In detail, with a window size of s, Equation (18) actually takes $2 \cdot s - 1$ words into consideration. And these words are weighted based on their relevance to the user-item pair, leading to more precise higher level semantic information extraction. The intuition behind our mean-pooling setting is that users could express their opinions on various aspects for an item in their reviews. For example, a movie fan considers not only the cast but also the director and the genre to make a rating. By using mean-pooling strategy, more relevant latent features would be extracted in the abstraction layer. Instead, a max-pooling strategy may ignore some important features due to its downsampling property, such that only the most important feature is retained. Our experimental results also validate the superiority of using the mean-pooling strategy (ref. Section 4.4).

Note that we use different $\mathbf{W}_a^*$ for the users and items, respectively, in the CNN network. The purpose is to allow the two independent CNN models to project both $\mathbf{U}^w$ and $\mathbf{V}^w$ into the same latent space. Finally, we stack one shared MLP layer to further extract higher-level features:

$$\mathbf{t}_u = \sigma(\mathbf{W}_1 \mathbf{h}^u + \mathbf{b}_1) \tag{19}$$

$$\mathbf{t}_i = \sigma(\mathbf{W}_1 \mathbf{h}^i + \mathbf{b}_1), \tag{20}$$

where $\mathbf{W}_1 \in \mathbb{R}^{l \times f}$ is the transformation matrix and $\mathbf{b}_1$ is the bias vector. Finally, we form the context-aware latent feature vector for the user-item pair as follows:

$$\mathbf{z}_{\text{review}} = [\mathbf{e}_{u,i}^{\text{review}} \oplus \mathbf{t}_u \oplus \mathbf{t}_i] \tag{21}$$

$$\mathbf{e}_{u,i}^{\text{review}} = \mathbf{t}_u \odot \mathbf{t}_i, \tag{22}$$

where $\odot$ is the element-wise product of vectors and $\oplus$ is the vector concatenation operation. Recall that $\mathbf{t}_u$ and $\mathbf{t}_i$ are dependent on the user and item jointly (Equations (9-17)). The element-wise product $\odot$ used for $\mathbf{e}_{u,i}^{\text{review}}$ further enhances latent feature interactions. Hence, the derived latent feature vector $\mathbf{z}_{\text{review}}$ for the user-item pair captures both the individual characteristics and their interactions together. CARL then takes $\mathbf{z}_{\text{review}}$ through a linear combination and higher order interaction modeling by using an FM for rating prediction (ref. Equation (3)), denoted as $\hat{y}_{u,i}(\mathbf{z}_{\text{review}})$.

3.3 Interaction-Based Feature Learning

Although textual reviews provide rich information about user preferences and the characteristics of items, the latent features $\mathbf{z}_{\text{review}}$ learned earlier are just review-driven and hence do not represent the user's rating behavior to its fullest. Therefore, we devise a learning process for latent features extraction using the user-item rating scores.

We use a separate set of latent vectors for the users and items in interaction-based feature learning. Given the one-hot encoding of user/item identity $\mathbf{x}_u/\mathbf{x}_i$, we project it to its corresponding latent vector $\mathbf{p}_u/\mathbf{q}_i$ as

$$\mathbf{p}_u = \mathbf{P}\mathbf{x}_u \tag{23}$$

$$\mathbf{q}_i = \mathbf{Q}\mathbf{x}_i, \tag{24}$$

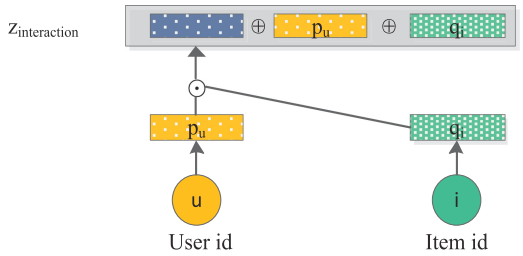


Fig. 3. The network architecture of the interaction-based learning.

where $\mathbf{P} \in \mathbb{R}^{l \times p}$ and $\mathbf{Q} \in \mathbb{R}^{l \times q}$ denote the latent feature matrices of users and items respectively,[3] and p and q are the numbers of users and items. Since no context information like textual reviews is available, we then use an element-wise product operation to extract interaction-based features not included in either $\mathbf{p}_u$ or $\mathbf{q}_i$:

$$\mathbf{e}_{u,i}^{\text{int}} = \mathbf{p}_u \odot \mathbf{q}_i. \tag{25}$$

Similar to Equation (21), we form the context-aware latent feature vector for the user-item pair as follows:

$$\mathbf{z}_{\text{interaction}} = [\mathbf{e}_{u,i}^{\text{int}} \oplus \mathbf{p}_u \oplus \mathbf{q}_i]. \tag{26}$$

Figure 3 depicts the interaction-based learning process on the basis of user-item pairs. We feed $\mathbf{z}_{\text{interaction}}$ as input into the FM for rating score prediction, denoted as $\hat{y}_{u,i}(\mathbf{z}_{\text{interaction}})$.

3.4 Fusion

The two learning components described earlier extract context-aware latent features from two different information sources. It is expected that the integration of the two components could complement each other and yield better prediction performance. A simple solution is to linearly interpolate the estimated rating score as follows:

$$\hat{y}_{u,i} = \alpha \hat{y}_{u,i}(\mathbf{z}_{\text{review}}) + (1 - \alpha)\hat{y}_{u,i}(\mathbf{z}_{\text{interaction}}) + b_u + b_i, \tag{27}$$

where parameter α works as a tradeoff between the two components. However, linearly fusing the two models with a static α may not be a suitable choice for rating prediction. It is likely that a user could give a high score for an item because of some specific preferred features but ignore other, moderate characteristics. Analogously, we introduce a dynamic weighting scheme by preferring the component with the higher rating prediction (i.e., either the knowledge from the explicit reviews or other factors). Specifically, we calculate α as follows:

$$\alpha = \frac{\hat{y}_{u,i}(\mathbf{z}_{\text{review}})}{\hat{y}_{u,i}(\mathbf{z}_{\text{review}}) + \hat{y}_{u,i}(\mathbf{z}_{\text{interaction}})}. \tag{28}$$

In Equation (28), parameter α becomes larger when the review-based component predicts a higher score than does the interaction-based component.

3.5 Model Optimization

The parameters of CARL are optimized based on Equations (4) and (27) with Stochastic Gradient Descent (SGD) and back-propagation. That is, the parameters for both learning components are

[3]For simplicity, we restrict the dimension size of user and item latent feature vectors to be identical in two learning components.

Table 1. Statistics of the Seven Datasets

Datasets	# Users	# Items	# Ratings	# Words per Review	# Words per User	# Words per Item	Density
Beer	7,725	21,976	66,625	17.31	34.06	103.20	0.039%
Musical Instruments	1,429	900	10,261	32.45	141.32	200.12	0.798%
Office Products	4,905	2,420	53,228	48.15	197.93	229.52	0.448%
Digital Music	5,540	3,568	64,666	69.57	216.21	266.51	0.327%
Video Games	24,303	10,672	231,577	72.13	188.79	260.60	0.089%
Tools Improvement	16,638	10,217	134,345	38.75	162.53	212.48	0.079%
Yelp 16-17	167,106	100,229	1,217,208	38.86	133.60	155.18	0.007%

jointly learned. For parameter update, we utilize RMSprop [35] over mini-batches. Additionally, to prevent overfitting, we adopt a dropout [32] strategy for the MLP layers.

4 EXPERIMENTS

In this section, we conduct extensive experiments on seven real-world datasets from different domains for performance evaluation. We also analyze the contributions of the two components and different settings for CARL.[4] Finally, a thorough analysis of review-based feature learning and a case study are presented.

4.1 Experimental Settings

Datasets. Five datasets are from Amazon 5-cores datasets[5] [12] : *Musical Instruments*, *Office Products*, *Digital Music*, *Video Games,* and *Tools Improvement*. The other two datasets are collected from the RateBeer website[6] and the Yelp challenge website,[7] named *Beer* [25] and *Yelp.,* respectively. Note that we also extract the 5 cores over *Beer* and *Yelp*, such that each user and item has at least 5 reviews. In addition, for *Yelp* we only select those records spanning 2016 to 2017 as the final dataset, denoted as *Yelp16-17*. These datasets consist of users' explicit ratings on items ranging from 1 to 5[8] and contain the textual reviews made by the users. Following the preprocessing steps used in Kim et al. [17], we perform the preprocessing for the review documents for all datasets as follows: (i) remove stop words and words that have the document frequency higher than 0.5; (ii) calculate tf-idf score for each word and select the top 20,000 distinct words as vocabulary; (iii) remove all words out of the vocabulary from raw documents; and (iv) amputate (pad) the long (short) review documents to the same length of 300 words. We further filter out the rating records which contain empty review after document preprocessing.

Table 1 summarizes detailed statistics about the seven datasets after preprocessing. We can see that the seven datasets hold different characteristics. The Yelp16-17 is a large-scale dataset while Beer is much smaller. They are both sparse in terms of interaction data and review data. On the contrary, the Digital Music dataset is dense on both data sources. The Music Instruments dataset is the smallest, but its interaction matrix is the densest among all seven datasets. In contrast, Video Games and Tools Improvement are two large datasets and are much sparser. In summary, these seven real-world datasets hold varying characteristics, covering a much broader range of real-world scenarios. For evaluation, we randomly select 80% of each dataset as the training set and the remaining 20% as the testing set. We further split 10% of the training set as the validation set

[4]The implementation is available at https://github.com/WHUIR/CARL.
[5]http://jmcauley.ucsd.edu/data/amazon/.
[6]https://www.ratebeer.com/.
[7]https://www.yelp.com/dataset/challenge.
[8]In Beer, we convert the rating range of the Overall Rating which is [4, 20] into [1, 5].

for hyper-parameter validation. The training sets are selected such that at least one interaction for each user and item should be included. Following the work in Catherine and Cohen [5], the reviews in the validation set and testing set are excluded since they are unavailable during rating prediction.

Baselines. We compare the proposed CARL against the following state-of-the-art rating prediction methods:

- PMF: Probabilistic Matrix Factorization is a standard matrix factorization model that uses only rating scores [28]. We use the Alternating Least Squares (ALS) techniques for model optimization.
- CDL: Collaborative Deep Learning [41] is the first hierarchical Bayesian model to build the connection between the deep learning technique (SDAE) [37] and the MF model. Following the adaption used in Kim et al. [17], we set the confidence parameter to 1 if the rating is observed and 0 otherwise.
- RBLT: The Rating-Boosted Latent Topics model integrates both the MF model and the topic model [34]. It proposes a rating-boosted approach which utilizes the rating-boosted reviews and rating scores together for rating prediction.
- CMLE: Collaborative Multi-Level Embedding Learning integrates a word embedding model with MF to learn user and item embeddings [44]. Given a new user-item pair, the rating can be predicted by the DP of its user and item embeddings.
- ConvMF: Convolutional Matrix Factorization integrates CNN into PMF for rating prediction [17]. The item latent features are extracted by using CNN over the item review documents.
- DeepCoNN: A Deep Cooperative Neural Network uses two parallel CNN networks to extract latent feature vectors from both user review documents and item review documents [45]. FM is then used for rating prediction.
- D-attn: The dual local and global attention model leverages global and local attentions to enable an interpretable embedding of users and items [30]. Finally, the rating can be estimated by the DP of the user and item embeddings.
- TransNets: TransNets extends the DeepCoNN model by adding an additional layer to represent the target user-item review, which is unavailable at test time [5]. Then TransNets can mimic the target user-item review representation at test time and thus improve the performance of rating prediction.

The first method listed is the conventional latent model that utilizes only the user-item rating scores. The rest are methods that utilize review documents for rating prediction. D-attn is an attention-based recommendation model.

Hyper-parameter Settings. We use grid search to tune the hyper-parameters for all the methods based on the setting strategies reported by their papers, and we report their performance over 5 runs on the testing set. The latent dimension size is optimized from $\{15, 25, 50, 100, 150, 200, 300\}$. The embedding dimension size of words in all models is set to 300. The batch size for Musical Instruments, Office Products, Digital Music, and Beer is set to 100. For the other three big datasets (i.e., Video Games, Tools Improvement and Yelp16-17), the batch size is set to 200. The number of convolution filters is set to 50. The statistical significance is conducted by applying the student *t-test*.

For CARL, the dimension size for user and item latent feature vectors is $l = 15$, window size is $s = 3$, and v is set to 50 for FM rating prediction. The dropout rate is set to 0.2. The learning rate is set to 0.001. The regularization parameter λ_Θ is tuned from $[0.05, 0.01, 0.005, 0.001]$ for the seven datasets.

Table 2. Overall Performance Comparison on Seven Datasets

Method	Musical Instruments	Office Products	Digital Music	Video Games	Tools Improvement	Beer	Yelp16-17
PMF	1.401†	1.091†	1.211†	1.669†	1.564†	1.636†	2.575†
CDL	0.861†	0.754†	0.882†	1.179†	1.033†	0.678†	1.727†
RBLT	0.815†	0.757†	0.872†	1.147†	0.983†	0.576†	1.570†
CMLE	0.818†	0.761†	0.883†	1.254†	1.023†	0.607†	1.592†
ConvMF	0.991†	0.960†	1.084†	1.449†	1.240†	0.853†	1.992†
DeepCoNN	0.814†	0.860†	1.060†	1.238†	1.063†	0.617†	1.592†
TransNets	0.799†	0.760†	0.910†	1.196†	1.008†	0.586†	1.525†
D-attn	0.984†	0.824†	0.914†	1.142†	1.046†	0.616†	1.575†
CARL	**0.776**	**0.722**	**0.831**	**1.065**	**0.942**	**0.557**	**1.446**
▲%	2.89%	4.24%	4.70%	6.74%	4.17%	3.30%	5.18%

The best and second best results are highlighted in boldface and underlined, respectively. ▲% denotes the improvement of CARL over the best baseline performer. † indicates that the difference from the best result is statistically significant at 0.01 level.

Evaluation Metric. The well-known Mean Square Error (MSE) is adopted for performance evaluation:

$$MSE = \frac{1}{|O_t|} \sum_{(u,i) \in O_t} (y_{u,i} - \hat{y}_{u,i})^2, \tag{29}$$

where O_t is the set of the user-item pairs in the testing set.

4.2 Performance Evaluation

The overall performance of all methods is reported in Table 2. The best and the second best results are highlighted in boldface and underlined, respectively. Here, we make the following observations.

First, we can see that PMF performs worst on all seven datasets. Among the seven datasets, PMF performs much worse on Video Games, Tools Improvement, Beer, and Yelp16-17 than on the other three datasets. This is reasonable since these four datasets have the sparsest user-item interaction data. All review-based rating prediction methods evaluated here perform much better than PMF, especially for those datasets with sparser interaction data (e.g., Beer and Yelp16-17). This proves that incorporating review information can provide more semantic information for understanding user rating behaviors than using a single rating score.

Second, among review-based baseline methods, RBLT performs relatively well across the seven datasets. By leveraging the user and item biases, RBLT successfully achieves the second best score on Digital Music, Tools Improvement, and Beer datasets. Also, RBLT obtains very close performance to the best baseline performer on the other datasets. Such good performance should be attributed to its rating-boost reviews, which ensure that the preferred features in high-rating reviews are extracted successfully. On the other hand, ConvMF and DeepCoNN achieve varying and unstable performance across the seven datasets. For the datasets with long reviews (e.g., Office Product, Digital Music, and Video Games), they obtain relatively poorer performance, which indicates that both ConvMF and DeepCoNN fail to extract relevant features from long reviews and that the prediction performance is adversely affected by noise and irrelevant information within the reviews. The extended DeepCoNN method (TransNets) outperforms DeepCoNN across the seven datasets, which is consistent with the results reported in Catherine and Cohen [5]. D-attn achieves marginal improvement over ConvMF on three datasets. However, on datasets with long review documents (e.g., Office Products, Digital Music, and Video Games), D-attn obtains significant

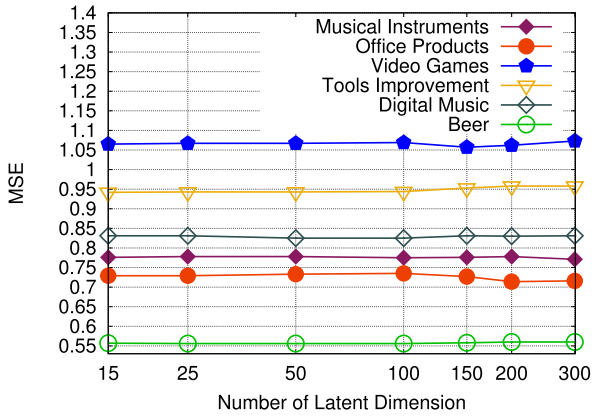


Fig. 4. Impact of dimension number l across the six datasets.

improvement over both DeepCoNN and ConvMF, which demonstrates that an attention mechanism does help CNN capture relevant information from reviews especially when the review data are ample. This observation is also in line with the experimental results reported in Seo et al. [30] and also validates the effectiveness of the attention mechanism on attending to important features in numerous features.

Third, CARL consistently achieves the best MSE scores across the seven datasets. The last row indicates the relative improvements of CARL over the best baseline performer. We can observe that the improvements gained by CARL are consistent and stable. On average, the relative improvement of CARL against the best baseline is 4.46%. For the sparsest dataset—Yelp16-17—it gains a 5.18% relative improvement compared to the best baseline. Even for the sparse dataset with the fewer review data—Beer—it still gains a 3.30% relative improvement over the best baseline. This result implies that CARL is effective for rating prediction on datasets with different characteristics. Moreover, the significant performance gap between CARL and D-attn validates that the context-aware user-item representation learning devised for CARL captures more knowledge about users' diverse rating behaviors on items by considering both the individual characteristics of the user and item and their interactions.

4.3 Analysis of CARL

We now study the impact of different model settings for CARL.

Number of Dimensions. Figure 4 plots the performance of CARL by varying the latent dimension size l in $\{15, 25, 50, 100, 150, 200, 300\}$. Due to the large performance gap between Yelp16-17 and the other datasets, the performance on Yelp16-17 is omitted for better visualization of performance on the other datasets.[9] We can see that CARL performs consistently well across a wide range of l values (i.e., $[15, 300]$) with little performance variation. Even with a much smaller l value (i.e., $l = 15$), CARL achieves nearly optimal rating prediction accuracy in most datasets. We believe this is attributable to context-aware feature learning on the basis of the user-item pair. Although a much larger l (*e.g.*, $l \geq 150$) could further obtain a small performance gain for Office Products and Video Games, the resultant computation cost is much larger because FM utilizes second-order feature interactions for rating prediction. Accordingly, we use $l = 15$ in the experiments.

[9] A similar performance pattern is also observed on Yelp16-17.

Table 3. The Impact of the Two Components in CARL

Methods	Musical Instruments	Office Products	Digital Music	Video Games	Tools Improvement	Beer	Yelp16-17
Rating-int	0.796	0.753	0.955	1.280	1.017	0.677	1.604
Review-int	0.783	0.745	0.885	1.080	0.961	0.565	1.468
Rating	0.785	0.744	0.938	1.270	1.013	0.705	1.597
Review	0.782	0.740	0.862	1.087	0.955	0.566	1.450
CARL	**0.776**	0.722	**0.831**	**1.065**	**0.942**	0.557	**1.446**
CARL+LR	0.779	**0.714**	0.842	1.069	0.944	**0.553**	1.451
CARL+Const	0.793	0.729	0.868	1.077	0.948	0.564	1.450
CARL+Tower	0.781	0.729	0.863	1.076	0.946	0.561	1.453

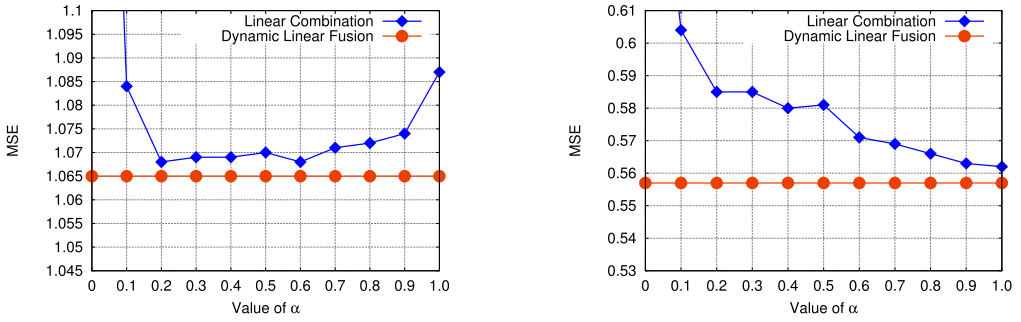
Review: Review-based feature learning. Rating: Interaction-based feature learning. Review-int: Review-based feature learning without $\mathbf{e}_{u,i}^{\text{review}}$. Rating-int: Interaction-based feature learning without $\mathbf{e}_{u,i}^{\text{int}}$. CARL+LR: A linear regression is utilized as the rating predictor. CARL+Const: A constant and equal-size fully connected layer is added before linear regression. CARL+Tower: A tower pattern fully connected layer is added before linear regression. The best results are highlighted in boldface.

The Impact of Two Learning Components. Recall that we use two independent learning components to derive context-aware latent features from the reviews and user-item rating scores, respectively. We further study the impact of each component (i.e., review-based feature learning and interaction-based feature learning). It is worthwhile to note that parameter l determines the model capacity. For fair comparison, we set the dimension size at $l = 30$ for each component, such that the final feature dimensions fed into FM is equivalent to CARL with $l = 15$ (ref. Equations (21) and (26)). Table 3 lists the prediction accuracy of the two components and CARL. We can see that, on the seven datasets, the performance of the fusion model CARL is much better than the performance of the two components. In essence, CARL can reap the benefits from both reviews and user-item rating scores since the two independent components complement each other, and therefore, the combination leads to better prediction accuracy.

In addition, we observe that both components achieve comparable or even better performance than all state-of-the-art baseline methods (ref. Tables 2 and 3). The interaction-based feature learning component (Rating) outperforms all baselines on the dense datesets (i.e., Musical Instruments and Office Products). Meanwhile, as an interaction-based method, it is comparable to some review-based methods on the other datasets.

Another observation is that using review-based feature learning (Review) alone obtains the best rating prediction accuracy against all baseline methods across the seven datasets (ref. Tables 2 and 3), although in some cases the prediction performance delivered by review-based feature learning is very close to those obtained by interaction-based feature learning. We need to emphasize that, for a sparse dataset, review-based feature learning substantially outperforms the interaction-based ones. This observation is in line with previous studies. Compared with ConvMF and DeepCoNN, the two CNN-based neural methods, review-based feature learning achieves significantly better rating prediction accuracy, though they all utilize convolution operations to extract semantic information from the reviews in the first place. Moreover, in contrast to D-attn, the attention-based neural method, review-based feature learning also gains a large improvement, which is attributable to the strength of dynamic attention. Overall, these observations confirm the superiority of context-aware user-item representation learning devised in CARL.

The Impact of Latent Feature Interactions. Latent feature interaction is an important building block for CARL since it captures some intricate relations between users and items. Actually, we exploit different kinds of latent feature interactions for context-aware user-item representation



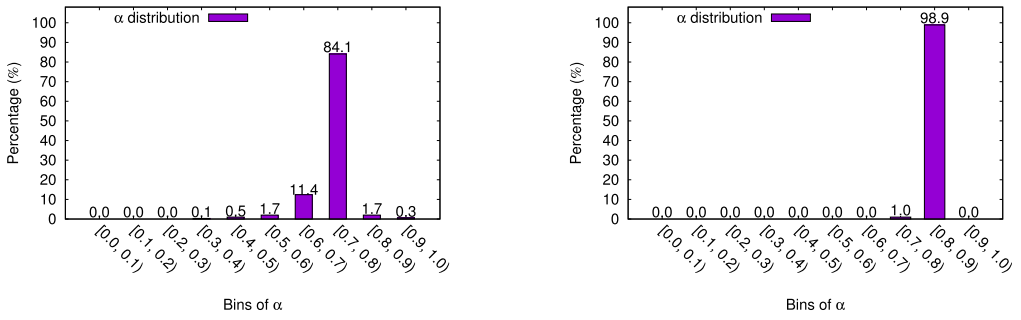
(a) Video Games (b) Beer

Fig. 5. Impact of dynamic linear fusion on Video Games (a) and Beer (b).

learning. The pair-dependent user and item latent features vectors $\mathbf{t}_u$ and $\mathbf{t}_i$ are derived by the review-based learning component based on the semantic interactions between the review documents. We further introduce $\mathbf{e}_{u,i}^{\text{review}}$ and $\mathbf{e}_{u,i}^{\text{int}}$ in the final user-item representation to boost latent feature interactions. Also, we utilize FMs (ref. Equation (1)) to model higher order feature interactions for rating prediction. Here, we first examine the impact of $\mathbf{e}_{u,i}^{\text{review}}$ and $\mathbf{e}_{u,i}^{\text{int}}$ via an ablation test for two learning components, respectively. We include the prediction performance by removing these two element-wise product-based latent features from the two components (Review-int/Rating-int) in Table 3. We can see that both components experience performance degradation in almost all cases, except for Rating-int on Beer dataset and Review-int on Video Games and Beer datasets. Another observation is that the interaction-based learning component achieves relatively better performance compared to those ones $\mathbf{e}_{u,i}^{\text{int}}$ ((i.e., Rating-int) on three smaller but denser datasets (i.e., Musical Instruments, Office Products, and Digital Music). We believe that the interaction-based knowledge learned through $\mathbf{e}_{u,i}^{\text{review}}$ and $\mathbf{e}_{u,i}^{\text{int}}$ can capture more relevant information.

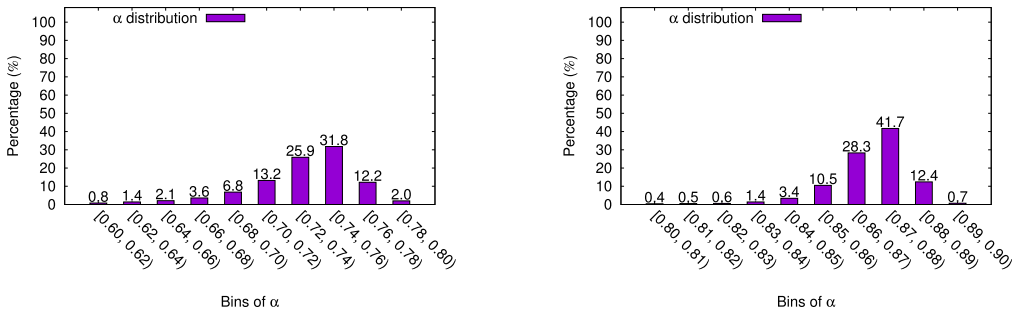
We further examine the impact of using an FM as a rating predictor in CARL. Here, we replace the FM with a linear regression over the user-item representations (i.e., $\mathbf{z}_{\text{review}}$ and $\mathbf{z}_{\text{interaction}}$) for rating prediction. The third to last row of Table 3 reports the prediction performance with this setting (CARL+LR). Although CARL achieves better prediction accuracy with FM in most cases, we observe that the advantage is relatively weak except for the Digital Music dataset. This is reasonable since the two learning components in CARL already model latent feature interactions explicitly from the reviews and rating scores, respectively, and the undiscovered knowledge regarding users' rating behaviors left for the FM would be marginal for many cases. We also evaluate the variation by adding one layer of MLP with nonlinearity within CARL+LR (i.e., CARL+Const and CARL+Tower). In detail, we utilize a one-layer MLP to further extract higher level features from both $\mathbf{z}_{\text{review}}$ and $\mathbf{z}_{\text{interaction}}$, respectively. Then, linear regression is used to calculate the prediction instead of using an FM. However, further worse performance is experienced (the last two rows of Table 3).

The Impact of Dynamic Linear Fusion. Recall that, in Equation (28), we adopt a dynamic linear fusion mechanism to prefer the component with a higher rating prediction. Figure 5(a) and 5(b) plots the performance comparison between dynamic linear fusion and the static linear combination by fixing α to a specific value in Equation (27). We can see that the dynamic linear fusion achieves better prediction accuracy than the linear combination with different α settings. What's more, dynamic fusion can eliminate the need of parameter tuning required in the static linear combination. Similar observations are also made in the other five datasets.



(a) Video Games (b) Beer

Fig. 6. The distribution of α on Video Games (a) and Beer (b).



(a) Video Games (b) Beer

Fig. 7. Finer analysis of α in the dominating range on Video Games (a) and Beer (b).

To better analyze the mechanism of dynamic linear fusion in CARL, we visualize the distribution of α by equally partitioning the value of α into 10 bins (plotted in Figure 6). We can observe that, to some extent, the major distribution of α fits the performance trends in the linear combination. For example, on the Video dataset, most α fall into the range in $[0.7, 0.8)$ while linear combination achieves the best performance when α is around 0.6. Nevertheless, an elastic and fine-granular assignment of α is conducted by dynamic linear fusion. To provide a finer analysis of this elastic mechanism, we zoom in to the distribution of α with respect to the dominating value range in Figure 7. We can see that, although the value of α is mainly falling into a specific dominating range on both Video Games and Beer (i.e., $[0.7, 0.8)$ and $[0.8, 0.9)$), the elastic assignment of α for different user-item pairs is clearly observed in both Figure 7(a) and 7(b). The improvement gained by the dynamic linear fusion also justifies the effectiveness of this elastic mechanism.

4.4 Analysis of Review-based Feature Learning

To get a better understanding of the review-based learning component, we further analyze the impact of two key layers: the attentive layer and the abstracting layer.

The Impact of Attentive Layer. In review-based feature learning (Review), we adopt an attentive layer to capture pair-based relevant information on the user-item pair. We evaluate the impact of the attentive layer via an ablation test by removing it from the review-based feature learning component. To eliminate the impact of the attentive layer, we set the attentive weights at 1 in Equations (14) and (15) (Review-att). Table 4 shows the performance comparison. We can see that

Table 4. The Impact of the Attentive Layer in the Review-Based Feature Learning

Methods	Musical Instruments	Office Products	Digital Music	Video Games	Tools Improvement	Beer	Yelp16-17
DeepCoNN	0.814	0.860	1.060	1.238	1.063	0.617	1.592
Review-att	0.813	0.766	0.933	1.108	1.028	0.587	1.554
Review	**0.782**	**0.740**	**0.862**	**1.087**	**0.955**	**0.566**	**1.450**
▲%	3.81%	3.39%	7.61%	1.89%	7.10%	3.58%	6.69%

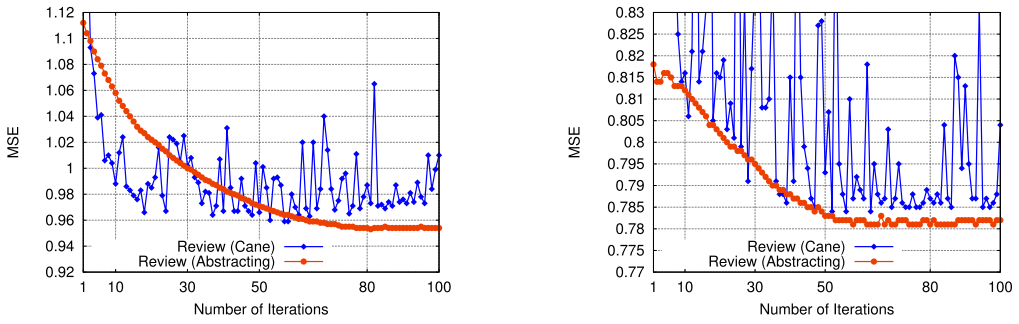
Review: Review-based feature learning. Review-att: Attentive layer is removed. The best and second best results are highlighted in boldface and underlined respectively. ▲% denotes the improvement of Review against Review-att.

when the attentive layer is applied, the performance of review-based component is significantly better than Review-att across the seven datasets. The relative improvement is 4.87% on average. The encouraging improvement confirms that the attentive layer is able to render the model to perform better prediction by identifying the pair-based relevant information for user-item pairs. A case study of the attentive layer for prediction interpretation is included in the next section.

The Impact of Abstracting Layer. Apart from the attentive layer, the abstracting layer is another pivotal layer in review-based feature learning. Recall that Review-att has eliminated the impact of the attentive layer and thus can be regarded as an extension of Deep-CoNN, which adds an abstracting layer after convolution operations. In Table 4, we observe that Review-att consistently outperforms DeepCoNN on the seven datasets. In particular, for the Office Products, Digital Music, and Video Games datasets which contain relatively long review documents, Review-att gains substantial improvement over DeepCoNN. We attribute the improvement to the abstracting layer to the fact that a larger context is considered for higher level semantic information extraction via stacking the convolution operations.

In contrast to applying the abstracting layer, Tu et al. [36] propose a simple alternative by summing up the weighted contextual representations into one single vector as derived from vertex embedding (CANE). Although CANE has proved effective in modeling varying relations between vertices for network modeling, we have to point out that the textual information in Tu et al. [36] is different from ours. In that work [36], the textual information is research papers written by the corresponding authors (i.e., vertices) while the textual information we are dealing with is the concatenation of reviews on different user-item pair. Apparently, review documents are less coherent and more diverse than research papers. Lots of noise is likely to be included by merely summing up the weighted contextual features. To tackle this issue, we employ the abstracting layer on the attentive layer. Furthermore, we believe that an abstracting layer can extract higher level semantic features from the weighted contextual features over a larger range. Figure 8(a) and 8(b) shows the performance comparison per iteration between review-based feature learning with an abstracting layer and with a CANE-based alternative on Video Games and Musical Instruments, respectively. First, we observe that better prediction accuracy is achieved with the inclusion of the abstracting layer. Second, in terms of convergence, it is obvious that the performance by including the abstracting layer gradually becomes stable while the CANE-based strategy remains fluctuating within a large range. With the abstracting layer, the review-based learning component almost reaches convergence after 50 iterations on both datasets. These observations demonstrate that the abstracting layer is capable of extracting higher level semantic features from the weighted contextual features and therefore enhances the performance and robustness of the model.

In the abstracting layer, we choose to utilize the mean-pooling strategy instead of the prevalent max-pooling strategy. The reason lies in the assumption that we believe a mean-pooling operation would retain a greater number of important features than would the max-pooling operation. The performance comparison between the review-based feature learning with a mean-pooling strategy



(a) Tools Improvement (b) Musical Instruments

Fig. 8. Performance comparison per iteration between review-based feature learning with Abstracting layer and with CANE-based strategy on Tools Improvement (a) and Musical Instruments (b).

Table 5. The Impact of the Pooling Strategy in Review-Based Feature Learning (Review)

Methods	Musical Instruments	Office Products	Digital Music	Video Games	Tools Improvement	Beer	Yelp16-17
Review-max	0.783	**0.726**	0.865	1.096	0.982	0.577	1.543
Review-avg	**0.782**	0.740	**0.862**	**1.087**	**0.955**	**0.566**	**1.450**

Table 6. The Time Cost in Seconds for One Epoch Model Training (T) and Prediction on Test Phase (P) by Different Models

Methods	Musical Instruments		Office Products		Digital Music		Video Games		Tools Improvement		Beer	
	T	P	T	P	T	P	T	P	T	P	T	P
DeepCoNN	3.18	0.25	17.11	1.27	23.05	1.63	79.62	5.79	42.88	3.38	16.14	1.48
CARL	4.31	0.281	22.54	1.46	29.62	1.79	103.08	6.39	56.52	3.74	23.06	1.79
D-attn	8.79	0.76	45.28	4.07	54.63	5.00	198.03	17.71	114.44	10.35	53.45	4.93
TransNet	27.12	0.23	143.87	1.22	182.49	1.52	644.98	5.41	361.07	3.12	156.11	1.47

(Review-avg) and the review-based feature learning with a max-pooling strategy (Review-max) is reported in Table 5. Note that although Review-max achieves the best performance on Office Products dataset, it underperforms Review-avg on all other datasets. Especially for the sparse datasets, Review-avg gains substantial improvement over Review-max. Note that, on Yelp16-17, the improvement gained by Review-avg is much larger over Review-max. We believe this is due to fact that the user-generated review in Yelp is likely to consist of the user's evaluations of different aspects, each of which is related to the final rating. These results validate that the mean-pooling strategy extracts more useful features compared to Review-max. Accordingly, we employ a mean-pooling operation in the abstracting layer.

4.5 Visualization of Attentive Layer

To further understand the review-based feature learning component of CARL, we manually check whether the attentive layer can identify pair-based relevant semantic information from both user and item review documents on the basis of the user-item pair. Hence, we sample a user u and two user-item rating records associated with u in the testing set of Musical Instruments. The heat maps of the review documents for both sampled pairs are visualized in Table 7. The important words identified by Equations (12-13) are highlighted in yellow and golden yellow, respectively. Specifically, for a word w_j in the user review document with attention weight a_j^u, we calculate the

Table 7. Highlighted Words by Attentive Weights in the Review Documents of Two User-Item Pairs for a Sampled User u

$\text{Pair}_1(user_u, item_1, 5.0)$	$\text{Pair}_2(user_u, item_2, 3.0)$
$user_u$'s review document:	$user_u$'s review document:
So good that I bought another one. Love the heavy cord and gold connectors. Bass sounds great. I just learned last night how to coil them up. I guess I should read instructions more carefully. But no harm done, still works great! . . . , then it worked like a charm! Glad I got them, much less hassle tuning up for a gig. This tiny little amp is great for practicing outside or in a canoe or while hang gliding. But it is not very good at parties. It cannot compete with the noise level that 20 people in conversation can make. It is cheap, so you should just get one. I am having great fun experimenting around with different settings, instruments, and power feed voltage levels. (see the danelectro power supply for this thing, really cool!) · · · it would be nice if it remembered what channel it was last on so when you power cycle it doesn't switch on you. I mostly use it to feed one amp from two different instruments. · · · I use this cable to patch a preamp to an amp. it works really well. having the two different style of ends gives you good options when putting a system together. Nice cable, could be a little heavier. I managed to pull the wires apart on the straight plug, · · · · · · it is easy and fun to use! I stopped messign with the reverb in my amp and now only use this for a much nice sounds from my guitar. I actually bought it to try some experiments with my harmonica, but those were dismal failures. Big win for the guitar though! · · ·	So good that I bought another one. Love the heavy cord and gold connectors. Bass sounds great. I just learned last night how to coil them up. I guess I should read instructions more carefully. But no harm done, still works great! . . . , then it worked like a charm! Glad I got them, much less hassle tuning up for a gig. This tiny little amp is great for practicing outside or in a canoe or while hang gliding. But it is not very good at parties. It cannot compete with the noise level that 20 people in conversation can make. It is cheap, so you should just get one. I am having great fun experimenting around with different settings, instruments, and power feed voltage levels. (see the danelectro power supply for this thing, really cool!) · · · it would be nice if it remembered what channel it was last on so when you power cycle it doesn't switch on you. I mostly use it to feed one amp from two different instruments. · · · I use this cable to patch a preamp to an amp. it works really well. having the two different style of ends gives you good options when putting a system together. Nice cable, could be a little heavier. I managed to pull the wires apart on the straight plug, · · · · · · it is easy and fun to use! I stopped messign with the reverb in my amp and now only use this for a much nice sounds from my guitar. I actually bought it to try some experiments with my harmonica, but those were dismal failures. Big win for the guitar though! . . . . .
$item_1$'s review document:	$item_2$'s review document:
· · · These are really great bang for the buck, however, like many people said, they do not work well for all my pedals, round ends don't fit in everything.I mean these are cables. Defiantly a space saver. I can put all of my effects side to side ad hook em up with these patch cables. It's very convenient that they are angled like so, for low profile fits. My only complaint is that they easily fall out of the socket if you jostle them too much. These are very good quality in spite of their reasonable price. I love the flat, low profile, right angle plugs. My only complaint is I needed some in varying lengths · · · I'm running 6 pedals and no issues like crackling or cutting out. One thing that is important to note is that I'm not hard on my gear. I'm a home noodler, so I can't say if these patches can take road/stage abuse and lots of setup/tear-down cycles. They seem pretty buff and capable, but I can only report what I've experienced.	As with all Boss/Roland products, the Boss FS-6 performs as designed. I purchased it to control both my Boss ME 20 Multi effects pedal and my Fender Blues Jr amp · · · Everything works exactly as planned. The FS-6 rugged, compact (fits right into my into ME 20 carrying case) and easy to use.I read a few reviews where people were complaining about batteries · · · something to complain about. As with most guitar players I don't use batteries in my effects pedals I use an 9 volt AC/DC adapter. Visual Sound offers the "One Spot" AC/DC adapter kit that has a 9 volt battery snap connector adapter. I removed the battery cover, · · · I then put rubber feet (purchased at ACE Hardware) on the bottom to allow the snap connector cable · · · battery problem solved · · · The possibilities are limited only to your imagination. It is somewhat expensive but it is a Great Product!! If you need a duel switch then I highly recommend purchasing the FS-6.

(Continued)

Table 7. Continued

· · · the design means that you can save space in arranging your effects pedals, with enough length and flexibility to arrange them in a semicircle if, like me, you have a lot of pedals. I use these patch cables on my pedal board and they save valuable space. They work well and I haven't had any problems with any of them shorting out or making static or other noises. I highly recommend them. Plus, they look nice, unlike the various colors that come with the bulkier, cheaper 1' cords. · · · Simple. Solid. Compact. Convenient. Audio quality is unchanged from my no name metal patch cables. · · ·	I needed a footswitch for a variety of applications to include an acoustic amp, guitar amps, and other guitar/audio gear. Built like a tank and completely configurable. Well done again, Roland! · · · I use the to go up and down through my banks I have programmed on the ME-70. Build like a tank. Simply to use. Simply to hook up. I hear it can be used on other Boss and Roland gear. being new to guitar this is just way too much fun, makes my playing acutally sound like I may know what I am doing, easy to setup and easy to use. . . . . .
$user_u$'s review for $item_1$:	$user_u$'s review for $item_2$:
compact heads allow more pedals to be loaded onto your overpriced pedal board. the quality is good. I like how they are easy to take apart and modify and put together again. very handy for making it all just right. I will buy these again for sure.	Neither delighted nor disappointed. I have not found it to be intuitive to use and the battery compartment is a joke, but otherwise the construction is good. I dont use it very much because I havent really figured out what I want to do with it, and it is too big to fit in my briefcase full of blues harps, so I have the behringer A/B switch doing the job there.

score for w_j as follows:

$$\xi(w_j) = \frac{1}{Z}\left(a_j^u + \sum_{k=j-\frac{s-1}{2}}^{j+\frac{s-1}{2}} a_k^u/s\right), \tag{30}$$

where s is the window size of the convolutional layers (ref. Equations (5) and (18)) and Z is the normalization term. In Equation (30), the attention weights of the contextual $s-1$ neighboring words are also considered to compute the importance of word w_j, which is consistent with the convolution operations mentioned earlier. Here, we adopt a max-normalization scheme for parameter Z. The words whose $\xi(w)$ are larger than 0.6 and 0.8 are highlighted with yellow and golden yellow, respectively.

A triple ($user_u$, $item_i$, $y_{u,i}$) denotes a user-item interaction such that user u rates item i with a score $y_{u,i}$. Note that the two items demonstrated in Table 7 receive quite different rating scores from the same user (i.e.,5.0 vs. 3.0). Although there are many different aspects mentioned in the user review document, it is clear to see that the two attentive heat maps of the same user review document for two different items are quite different. For $item_1$, lots of positive words, such as "charm," "fun," "cool," "cheap," and "win," are highlighted with larger attentive weights. On the contrary, less positive words are found to be highlighted in the user review document for $item_2$. From the highlighted information in the two heat maps, we can speculate that "tiny shape" and "practicing outside" aspects about $item_1$ are the focus of user u, while the "noise level" and "power voltage" aspects about $item_2$ could be more important instead.

A further close look at the heat maps of the two item review documents confirms a lot about these speculations. For $item_1$, it is not diffcult to see that some aspects, such as "save valuable space," "flexibility," and "low profile" are highlighted. The reasonable price, good quality, compactness, and low-profile design are found to be relevant information. From $item_2$'s review document, we can see that many aspects related to the battery, its expensive price, and the construction of the product are highlighted. The observed correlations between the heat maps of the user review

document and the heat maps of the item review document reveal that CARL is effective in capturing pair-based relevant semantic information from the corresponding reviews for different user-item pairs.

Note that the real user review of an item is excluded from both user and item review documents for rating prediction since the user review is unavailable before her consumption. For the two user-item pairs studied earlier, we then list the corresponding real reviews provided in the original dataset in the last row of Table 7. We manually highlight the opinions expressed by the user in red. Note that good quality, compact shape, and convenient design of $item_1$ are highly appreciated, whereas the user has neutral sentiment towards item 2 due to the annoying battery compartment. This ground-truth knowledge further validates that the context-aware user-item representation learning devised in CARL enables a better understanding about users' diverse rating behaviors on items. These observations also suggest that the pair-based relevant semantic information identified by CARL in the user and item review documents facilitates an accurate interpretation about the recommendation decisions.

For an easier comparison of the attention distribution with respect to the item, we also sample another user-item pair based on $item_1$. A similar observation can be obtained in Table 8 and justifies the consistent performance of CARL on both users and items.

Here, we provide some analyses via conducting a comparison between CARL (the review-based feature learning) and three state-of-the-art deep review-based recommendation methods (DeepCoNN, D-attn, and TransNet). Figure 9 demonstrates the architectures of the four methods. We can make the following two observations.

First, DeepCoNN, D-attn and TransNet all belong to the Siamese architecture [4, 15], which performs late interaction on the latent representations of user and item only if they are mature. For instance, in DeepCoNN, the latent representations of user and item are formed without knowledge of each other. Therefore DeepCoNN runs the risk of losing some important contextual features which are only available through performing the interaction for the user-item pair. Thus, it is unsurprising that this simple architecture usually gains the worst performance across the datasets (ref. Table 2). The other two methods extend DeepCoNN in different ways. TransNet adds an additional layer to regularize the representations of the user review document and the item review document. D-attn tries to attend important local and global semantic features in user and item review documents. However, these two extensions still ignore the important contextual features for the user-item pair. We speculate that such interaction-based features play vital roles in rating prediction. For example, an appearance-valuing user may give a low rating to a plain-looking electric razor. However, in most cases, the appearance of an electric razor is less likely to be a universally recognized important characteristic. To model the diverse preferences of users for items, CARL employs an attention mechanism that refines the representation of the user based on the characteristics of the item and refines the representation of item based on the preferences of the user. That is, the context-aware latent representations of user and item are derived through the attention mechanism.

Second, we can observe that DeepCoNN, D-attn, and TransNet merely utilize the separated latent representations of user and item for rating prediction. The element-wise latent features interaction which carries the intricate relation between a user and an item in the latent space is ignored. To fill up the empty space, CARL leverages the element-wise product of user vector and item vector to further improve the accuracy of rating prediction (ref. Table 3).

On the other hand, both computation efficiency and response time are important considerations for a real-time recommendation scenario. Though CARL outperforms most Siamese architectures in terms of rating prediction, we still want to investigate whether such improvement is achieved by the oversacrifice of computation efficiency.

Table 8. Highlighted Words by Attentive Weights in the Review Documents of Two User-Item Pairs for a Sampled Item 1

$\text{Pair}_1(user_u, item_1, 5.0)$	$\text{Pair}_2(user_v, item_1, 3.0)$
$user_u$'s review document:	$user_v$'s review document:
So good that I bought another one. Love the heavy cord and gold connectors. Bass sounds great. I just learned last night how to coil them up. I guess I should read instructions more carefully. But no harm done, still works great! · · ·, then it worked like a charm! Glad I got them, much less hassle tuning up for a gig. This tiny little amp is great for practicing outside or in a canoe or while hang gliding. But it is not very good at parties. It cannot compete with the noise level that 20 people in conversation can make. It is cheap, so you should just get one. I am having great fun experimenting around with different settings, instruments, and power feed voltage levels. (see the danelectro power supply for this thing, really cool!) · · · it would be nice if it remembered what channel it was last on so when you power cycle it doesn't switch on you. I mostly use it to feed one amp from two different instruments. · · · I use this cable to patch a preamp to an amp. it works really well. having the two different style of ends gives you good options when putting a system together. Nice cable, could be a little heavier. I managed to pull the wires apart on the straight plug, · · · · · · it is easy and fun to use! I stopped messign with the reverb in my amp and now only use this for a much nice sounds from my guitar. I actually bought it to try some experiments with my harmonica, but those were dismal failures. Big win for the guitar though! · · · · · · · · ·	This is a simple preamp booster that adds a bit of dirt so if your looking for a clean boost this isn't it try something else. I find This best used to push the front end of a tube amp. its really similar to an overdrive except this boosts more than a normal one and is good for cutting through the mix on a solo or fattening up single coil pickups. overall great pedal and amazing price highly recommended. · · ·, it sounds really good. It maybe less durable than a metal pedal but who cares for this price I wouldn't mind buying a new one if mine broke and considering there is hardly any companies that actually make pitch vibrato's(uni-vibe/Vibe pedals are not pitch vibrato pedals)buying a new one isn't a problem. This pedal is ok at best there are a lot better metal pedals out there the only thing this one really has going for it is it has a built in noise gate other wise this pedal is pretty thin sounding and has far to much gain then you will ever need. I would personally recommend the Electro-Harmonix Metal Muff over this any day it is a better sounding pedal for about the same price. This mic is excellent, it's pretty much a straight up clone of an sm-57 and for the money there are no mic's even close to this one. Also the price at orange county speaker's website is $29 and $25 in store so I'm not entirely sure why it's $50 on Amazon. These will not wear down which is amazing and they have all the same great qualities as all the other jazz III's. the only problem I have with them is dirt can get between the grip pretty easily which makes them slip but if you clean them well then its not a problem.
$item_1$'s review document:	$item_1$'s review document:
· · · These are really great bang for the buck, however, like many people said, they do not work well for all my pedals, round ends don't fit in everything.I mean these are cables. Defiantly a space saver. I can put all of my effects side to side ad hook em up with these patch cables. It's very convenient that they are angled like so, for low profile fits. My only complaint is that they easily fall out of the socket if you jostle them too much. These are very good quality in spite of their reasonable price. I love the flat, low profile, right angle plugs. My only complaint is I needed some in varying lengths · · ·	· · · These are really great bang for the buck, however, like many people said, they do not work well for all my pedals, round ends don't fit in everything. I mean these are cables. Defiantly a space saver. I can put all of my effects side to side ad hook em up with these patch cables. It's very convenient that they are angled like so, for low profile fits. My only complaint is that they easily fall out of the socket if you jostle them too much. These are very good quality in spite of their reasonable price. I love the flat, low profile, right angle plugs. My only complaint is I needed some in varying lengths · · ·

(Continued)

Table 8. Continued

I'm running 6 pedals and no issues like crackling or cutting out. One thing that is important to note is that I'm not hard on my gear. I'm a home noodler, so I can't say if these patches can take road/stage abuse and lots of setup/tear-down cycles. They seem pretty buff and capable, but I can only report what I've experienced.	I'm running 6 pedals and no issues like crackling or cutting out. One thing that is important to note is that I'm not hard on my gear. I'm a home noodler, so I can't say if these patches can take road/stage abuse and lots of setup/tear-down cycles. They seem pretty buff and capable, but I can only report what I've experienced.
· · · the design means that you can save space in arranging your effects pedals, with enough length and flexibility to arrange them in a semicircle if, like me, you have a lot of pedals.	· · · the design means that you can save space in arranging your effects pedals, with enough length and flexibility to arrange them in a semicircle if, like me, you have a lot of pedals.
I use these patch cables on my pedal board and they save valuable space. They work well and I haven't had any problems with any of them shorting out or making static or other noises. I highly recommend them. Plus, they look nice, unlike the various colors that come with the bulkier, cheaper 1' cords.	I use these patch cables on my pedal board and they save valuable space. They work well and I haven't had any problems with any of them shorting out or making static or other noises. I highly recommend them. Plus, they look nice, unlike the various colors that come with the bulkier, cheaper 1' cords.
· · · Simple. Solid. Compact. Convenient. Audio quality is unchanged from my no name metal patch cables. · · ·	· · · Simple. Solid. Compact. Convenient. Audio quality is unchanged from my no name metal patch cables. · · ·
· · · not a single problem! I've changed my setup like crazy, sometimes even the cables had to be twisted in some certain position on a pedalboard. Been bent on the edge of the cable right before the connector, · · ·	· · · not a single problem! I've changed my setup like crazy, sometimes even the cables had to be twisted in some certain position on a pedalboard. Been bent on the edge of the cable right before the connector, · · ·
. . . . .	. . . . .
$user_u$'s review for $item_1$:	$user_v$'s review for $item_1$:
compact heads allow more pedals to be loaded onto your overpriced pedal board. the quality is good. I like how they are easy to take apart and modify and put together again. very handy for making it all just right. I will buy these again for sure.	all 12 of mine broke within a year and the jacks got bent pretty easily. so I switched back to the plastic ones which have lasted me over 2 yearsHosa CFS606 6 Inch Right Angle FX Pedal Cable, 6 Packand they are still going strong.

Here, we compare the training time and the prediction time taken for the deep models. The efficiency results are reported in Table 6, in ascending order. Specifically, we record the training time per epoch and the prediction time for each model under the same computing environment by using a Nivdia GTX TITAN X (12GB). The prediction time is the seconds required for a given model to finish the rating prediction for the whole testing set. We can observe that DeepCoNN takes the least training time per epoch while CARL is the second most efficient model in training time. D-attn lags far behind CARL as the local attention mechanism over multisize sliding kernels incurs extra burden. Similarly, in each epoch, TransNet has to pretrain the target network for a good representation of the target user-item review to better regularize the representations of user and item, which results in the longest training time. For rating prediction, TransNet attains a faster time than DeepCoNN. DeepCoNN and CARL are ranked in the second and third places, respectively. It is obvious that the additional computation cost incured by CARL over TransNet and DeepCoNN is relatively small. Overall, CARL obtains promising performance in terms of both effectiveness and efficiency.

5 CONCLUSION

In this article, we propose a novel fused context-aware neural model to learn user-item representations for rating prediction, named CARL. Both reviews and user-item rating scores are well exploited in CARL. By learning a representation for a user-item pair based on their individual

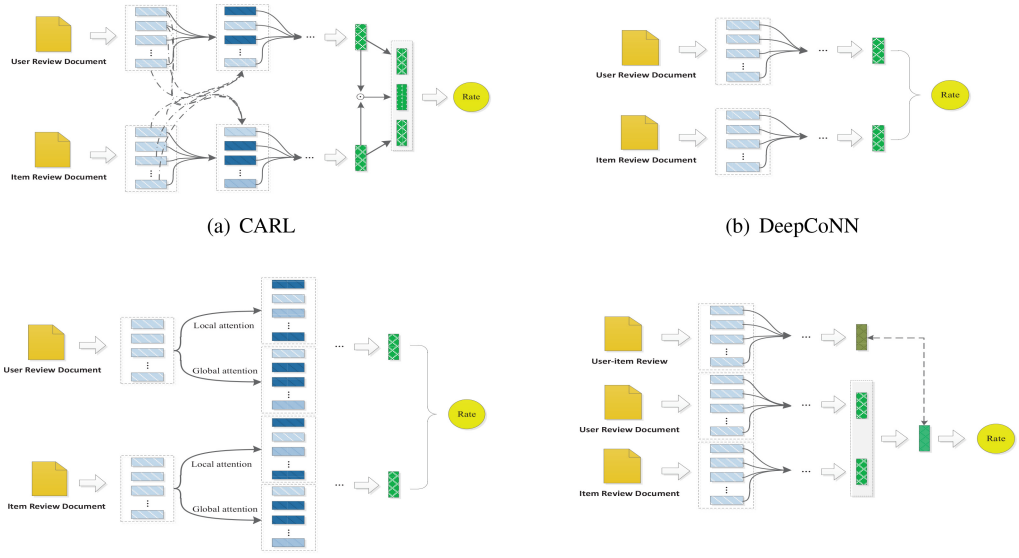


Fig. 9. Workflow comparison between CARL and the other three deep review-based recommendation methods.

characteristics and their interactions together, CARL yields a better understanding of user rating behaviors. The experimental results show that CARL consistently outperforms the existing state-of-the-art alternatives over seven real-world datasets. In addition, the attention mechanism utilized by CARL for review processing enables us to further provide semantic interpretations about recommendation decisions. Inspired by the promising performance delivered by RBLT and the case studies conducted for CARL, we plan to incorporate sentiment factors into CARL to enhance rating prediction. Also, the two learning components in CARL are only coupled late in a linear fusion manner. As a part of future work, we plan to incorporate the two information sources (i.e., reviews and rating scores) into a unified neural model that jointly derives a latent representation for a user-item pair.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research. Chenliang Li is the corresponding author.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473 (2014).

[2] Yang Bao, Hui Fang, and Jie Zhang. 2014. TopicMF: Simultaneously exploiting ratings and reviews for recommendation. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, Québec.* AAAI, 2–8.

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

[4] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data: Application to word-sense disambiguation. *Machine Learning* 94, 2 (2014), 233–259.

[5] Rose Catherine and William W. Cohen. 2017. TransNets: Learning to transform for recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems, Como, Italy.* ACM, 288–296.

[6] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural attentional rating regression with review-level explanations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, Lyon, France*. ACM, 1583–1592.

[7] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo*. ACM, 335–344.

[8] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12 (2011), 2493–2537.

[9] Mukund Deshpande and George Karypis. 2004. Item-based top-*N* recommendation algorithms. *ACM Transactions on Information Systems* 22, 1 (2004), 143–177.

[10] Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *CoRR* abs/1602.03609 (2016).

[11] Kostadin Georgiev and Preslav Nakov. 2013. A non-IID framework for collaborative filtering with restricted boltzmann machines. In *Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA*. JMLR.org, 1148–1156.

[12] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web, Montreal, Canada*. ACM, 507–517.

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web, Perth, Australia*. ACM, 173–182.

[14] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, Pisa, Italy*. ACM, 549–558.

[15] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of the Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems, Montreal, Quebec*. Curran Associates, Inc., 2042–2050.

[16] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD*. The Association for Computer Linguistics, 655–665.

[17] Dong Hyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA*. ACM, 233–240.

[18] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar*. ACL, 1746–1751.

[19] Yehuda Koren. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining Las Vegas, Nevada*. ACM, 426–434.

[20] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *IEEE Computer* 42, 8 (2009), 30–37.

[21] Daniel D. Lee and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *Proceedings of the Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS), Denver, CO*. MIT Press, 556–562.

[22] Guang Ling, Michael R. Lyu, and Irwin King. 2014. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender Systems, Foster City, Silicon Valley, CA*. ACM, 105–112.

[23] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Coevolutionary recommendation model: Mutual learning between ratings and reviews. In *Proceedings of the Conference on World Wide Web, Lyon, France*. ACM, 773–782.

[24] Julian J. McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems, Hong Kong, China*. ACM, 165–172.

[25] Julian J. McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *Proceedings of the 12th IEEE International Conference on Data Mining, Brussels, Belgium*. IEEE Computer Society, 1020–1025.

[26] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan M. Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining, Pisa, Italy*. IEEE Computer Society, 502–511.

[27] Steffen Rendle. 2010. Factorization machines. In *Proceedings of the 10th IEEE International Conference on Data Mining, Sydney, Australia*. IEEE Computer Society, 995–1000.

[28] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic matrix factorization. In *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems 20*. 1257–1264.

[29] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey E. Hinton. 2007. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning, Corvallis, Oregon*. ACM, 791–798.

[30] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Proceedings of the 11th ACM Conference on Recommender Systems, Como, Italy*. ACM, 297–305.

[31] Alex Smola. 2012. Recommender systems lecture. http://alex.smola.org/teaching/berkeley2012/slides/8_Recommender.pdf.

[32] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[33] Xiaoyuan Su and Taghi M. Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009 (2009), 421425:1–421425:19.

[34] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-boosted latent topics: Understanding users and items with ratings and reviews. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence, New York, NY*. IJCAI/AAAI Press, 2640–2646.

[35] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning* 4, 2 (2012), 26–31.

[36] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. CANE: Context-aware network embedding for relation modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics Vancouver, Canada*. Association for Computational Linguistics, 1722–1731.

[37] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research* 11 (2010), 3371–3408.

[38] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A deep architecture for semantic matching with multiple positional sentence representations. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence, Phoenix, Arizona*. AAAI Press, 2835–2841.

[39] Bingning Wang, Kang Liu, and Jun Zhao. 2016. Inner attention based recurrent neural networks for answer selection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, Germany*. The Association for Computer Linguistics, 1288–1297.

[40] Chong Wang and David M. Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA*. ACM, 448–456.

[41] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW*. ACM, 1235–1244.

[42] Xuejian Wang, Lantao Yu, Kan Ren, Guanyu Tao, Weinan Zhang, Yong Yu, and Jun Wang. 2017. Dynamic attention deep model for article recommendation by learning human editors' demonstration. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining Halifax, NS*. ACM, 2051–2059.

[43] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics* 4 (2016), 259–272.

[44] Wei Zhang, Quan Yuan, Jiawei Han, and Jianyong Wang. 2016. Collaborative multi-level embedding learning from reviews for rating prediction. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence, New York, NY*. IJCAI/AAAI Press, 259–272.

[45] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining, Cambridge*. ACM, 425–434.

Received July 2018; revised October 2018; accepted November 2018