Context-Aware User and Item Representations Based on Unsupervised Context Extraction From Reviews

PADIPAT SITKRONGWONG\textsuperscript{1}, ATSUHIRO TAKASU\textsuperscript{2}, (Member, IEEE), AND SARANYA MANEEROJ\textsuperscript{3}

\textsuperscript{1}The Graduate University for Advanced Studies, SOKENDAI, Tokyo 101-8430, Japan
\textsuperscript{2}National Institute of Informatics, The Graduate University for Advanced Studies, SOKENDAI, Tokyo 101-8430, Japan
\textsuperscript{3}Advanced Virtual and Intelligent Computing Center, Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University, Bangkok 10330, Thailand

Corresponding author: Padipat Sitkrongwong (padipat@nii.ac.jp)

This work was supported by the Cross-Ministerial Strategic Innovation Promotion Program (SIP) Second Phase, Big-Data and AI-Enabled Cyberspace Technologies from the New Energy and Industrial Technology Development Organization (NEDO).

\textbf{ABSTRACT} User reviews often supply valuable information to alleviate the rating sparsity problem that can occur in recommender systems. Recent work has employed deep learning techniques to learn user and item representations from reviews, which are then used to predict ratings. Such representations are usually learned by considering every word in previous reviews, including words that are irrelevant to user preferences or item features. Some approaches try to identify and extract significant words from reviews based on a predefined list of contexts, where contexts such as the season or weather could have strong influences on user decisions about items, and which are more relevant to their preferences or sought-after features. Specifying optimal values for contexts, however, is not a trivial task and the values are mostly restricted to a single word format. To overcome these limitations, we propose a novel unsupervised method for extracting contexts from reviews, which are then utilized to construct user and item representations. To this end, we adopt a region embedding technique to automatically extract a context as any word in a text region that influences patterns of rating distributions in reviews. Instead of considering every word in all previous reviews, our user and item representations are dynamically constructed based on different relevance levels among the extracted contexts from a particular review by applying our interaction and attention modules. Experiments demonstrated that utilizing our representations for rating prediction could outperform existing state-of-the-art context-aware and review-based recommendation techniques.

\textbf{INDEX TERMS} Attention mechanism, context-aware recommender systems, region embedding, representation learning.

\section{I. INTRODUCTION}
User-generated reviews can supply valuable information to alleviate the rating sparsity problem that can occur in the standard collaborative filtering-based (CF-based) recommendation approach, which utilizes rating data alone \cite{1}–\cite{3}. In particular, recent work has employed deep learning techniques and attention mechanisms to learn representations of users and items from reviews and use them for rating prediction \cite{4}–\cite{7}. Although these models utilize different types of networks to learn such representations, they share two similar principles that could limit their potential.

First, they consider every word in a review as an input when learning user and item representations. Given that some words are not relevant either to user preferences or item features, such words should not be given any weight when modeling their representations. For example, in hotel recommendations, words such as “clean” or “breakfast” are more relevant to a user preferences toward a hotel features than words such as “he” or “run”. If we can identify and utilize only the words relevant to a specific recommendation domain, the user and item representations could be constructed in a more efficient and meaningful way.

Moreover, the user and item representations are constructed in a static manner by aggregating their relevant previous reviews. This means that each user or item has one
fixed representation per review. We believe that, to predict a rating for a particular review, with the aim of modeling a user preferences and an item features for application to the user’s current situation, it is more important to concentrate and leverage the more relevant information embedded in that review. For example, the review in Fig. 1 mentions that the room offers breathtaking views of a city at night. To generate user and item representations for predicting a rating for this review, it would be beneficial to know how much the user prefers, and how much the hotel’s rooms are well known for, its city views at night. That is, our assumption is that the user and item representations should be dynamically constructed for each particular review, to capture the interactions between user preferences (or item features) and the relevant information in that review.

Visited this hotel during summer for family trip. The room offers a breathtaking view of the city at night. The bed is clean and comfortable, but smaller than our expectation. Have a freebuffet but weak signal. However, the staff is friendly and helpful. The continental breakfast is OK, but they should offer more selections.

FIGURE 1. Example of a user-generated review and rating.

Whereas the traditional CF-based approach considers rating data alone, context-aware recommendations improve the effectiveness of the recommendations by taking into account additional contextual information [8]. A piece of contextual information (or simply a “context”) such as location, time, or weather can have a major influence on users’ decisions when they are choosing items. For example, the type of clothes users would buy depends strongly on the season (e.g. winter or summer).

Many context-aware methods have been able to achieve improved prediction accuracy, compared to standard CF-based approaches [8]–[11]. However, two significant challenges remain. First, obtaining contexts is not a trivial task because they are rarely provided directly by users. Many context-aware datasets collect contextual information by predefining a list of contextual variables and possible corresponding values, as shown by the examples in Table 1, and asking users to select appropriate values for the contextual variables at the time they rate the items. To deliver the best possible predictive performance, this approach requires the expensive process of carefully predefining optimal values for the contextual variables, each of which tends to be associated with relatively few recommendation domains. Moreover, incorporating too many contexts tends to increase the dimensionality of the data, thereby triggering a sparsity issue. After obtaining the contexts, the second challenge is to identify and utilize only the contexts that are relevant to a specific recommendation task. Several methods define a relevant context as one that has a significant influence on the distribution of ratings [9], [18], [19]. Figure 2, for example, visualizes the rating distributions for two contextual variables, Companion and Season. Each cell contains the frequency of each contextual value for each rating value, and its grayscale shading emphasizes the density of the frequency. Note that each contextual value for Companion influences the distributions of ratings differently, whereas there is no significant difference in the distributions for Season. We can therefore hypothesize that Companion is a relevant context and Season is an irrelevant context for this data. In practice, most context-aware methods [18], [19] identify the relevance of contexts by applying statistical tests such as the paired t test to each contextual variable. However, this approach is only applicable when using predefined types of context (such as those presented in Table 1).

When writing reviews, users can express opinions describing their experiences and their satisfaction with items, which can be a valuable source of contexts [1]. As shown in Fig. 1, for example, underlined words such as “summer”, “family”, or “night” can be considered as contexts embedded in a review. However, extracting such contexts from reviews is not a trivial task. Most methods rely on text mining techniques to extract a context as a word that matches one of the pre-defined contextual values [12]–[14], which involves the same challenge of predefining optimal values for each contextual variable. To avoid this, several approaches have utilized (gives a stronger impression that corresponds to the number of citations unsupervised techniques for extracting contexts from reviews automatically [20]–[22]. By not having to predefine values for contexts, they hence become applicable to a wide variety of recommendation domains. However, these unsupervised methods lack a procedure for identifying the relevant contexts for each specific domain. Moreover, contexts extracted from most such methods have been restricted to a single word format, e.g. “family” or “breakfast”. In fact,

| Contextual variables | values |
|----------------------|--------|
| Daytype              | Weekday, Weekend, Holidays |
| Time of day          | morning, afternoon, evening, night |
| Weather              | sunny, cloudy, rainy, stormy, snowy |
| Season               | spring, summer, autumn, winter |
| Companion            | alone, partner, family, friends |
| Location             | home, public, office, school |

TABLE 1. Examples of predefined contextual variables and their values.
the precise meaning of a context might require more than one word for its expression, such as “family trip” or “continental breakfast.” In this work, we argue that effectively extracting and utilizing contexts in reviews could help overcome the challenges of obtaining and identifying relevant contexts in context-aware methods, in addition to the limitations of modeling user and item representations from reviews via deep learning techniques.

In our previous work [23], we proposed a novel unsupervised method for defining and extracting contexts from reviews. A relevant context in our work is defined not by a single word alone but by the word plus those of its neighboring words that influence the distributions of ratings. We develop a region embedding technique [24] to emphasize the words in a small text region for consideration as a context, and represent it by region embedding to be used for a rating prediction. By not having to predefine contexts, our method can be used to extract relevant contexts from reviews in any recommendation domain.

In this work, we propose an extension to the model developed for our context extraction method [23], namely context-aware region embedding (CARE). CARE derives region embedding representations for the extracted contexts that are output from our previous context extraction procedure. These are then input to our proposed rating prediction procedure, which contains two neural network modules for interaction and attention. Our interaction module first models the relevance of each context in a particular review based on its past interaction with an individual user preferences and item features. Our attention module then generates the user and item representations based on different relevance levels among contexts in each review. These two modules enable our model to dynamically construct unique user and item representations for each specific review, rather than have one static representation for all reviews, as found in most deep-learning-based methods. Finally, our user and item representations are used to predict ratings by exploiting a latent factor model [25].

We have evaluated the predictive performance of our CARE model on three well-known review datasets. Our experiments demonstrate that CARE outperforms existing state-of-the-art rating prediction methods that include both review-based and context-aware recommendation techniques [4], [7], [21], [22], [26], [27]. In addition, we provide an extensive discussion on the context extraction and rating prediction performance of the various aspects of CARE in various situations.

The main contributions of this work can be summarized as follows.

• We propose an unsupervised context extraction technique that is capable of extracting relevant contexts from review datasets in any recommendation domain.

• A relevant context in our work is defined not only by a single word format but also by including its adjacent and/or nonadjacent words that influence the distributions of ratings.

• Finally, our user and item representations are dynamically constructed for each particular review, to capture the different relevance levels among its extracted contexts to the individual user preferences and item features.

The remainder of this paper is organized as follows. We first describe the related work in Section II. Our proposed CARE, including the context extraction and rating prediction methods, is presented in Section III. We provide details of our experiments in Section IV and continue the discussion in Section V. Finally, we conclude our work in Section VI.

II. RELATED WORK

A. COLLABORATIVE FILTERING

By utilizing previous rating data, a CF-based approach creates a predictive model for estimating the ratings of unseen items for users. One effective approach in CF-based methods is the matrix factorization technique [25], a latent factor model that models a rating as an interaction between the latent features of user and item. Specifically, each user $u_i$ and each item $v_j$ are respectively associated with user and item latent-feature vectors $x_{iu}$ and $x_{jv} \in \mathbb{R}^k$, where $k$ is the number of latent dimensions. The rating of user $u_i$ for item $v_j$ is then estimated by

$$\hat{r}_{i,j} = x_{iu}^T x_{jv} + b_{ui} + b_{vj} + \mu,$$  (1)

where $b_{ui}$, $b_{vj}$, and $\mu \in \mathbb{R}$ respectively denote the bias for user $u_i$, the bias for item $v_j$, and the global bias. The model parameters are learned and optimized by minimizing the regularized square error through a loss function defined by

$$L = \sum_{(i,j) \in O} (r_{i,j} - \hat{r}_{i,j})^2 + \lambda (\|x_{iu}\|^2 + \|x_{jv}\|^2 + b_{ui}^2 + b_{vj}^2),$$  (2)

where $O$ denotes the set of observed user–item rating pairs, $r_{i,j}$ is the observed rating score of user $u_i$ toward item $v_j$, and $\lambda$ is a constant controlling the regularization rate. (The regularization term is added to avoid overfitting the observed rating data.)

The parameters $x_{iu}$ and $x_{jv}$ denote the representations of user $u_i$ and item $v_j$, respectively, in the latent space. Because they are learned and optimized from observed ratings, their representation quality depends on the quantity of available historical ratings. For most rating datasets, however, users typically rate only a few items among all the available items in the system, which leads to a rating sparsity problem. Such a problem will directly affect the quality of the user and item representations.

B. REVIEW-BASED RECOMMENDATIONS

To alleviate the rating sparsity problem, review-based approaches aim to learn user and item representations by leveraging the content of review data, in addition to the simple rating data [1]. The idea is that reviews can offer richer and more meaningful information than numeric ratings for constructing user and item representations. For example, consider the review in Fig. 1. Even if this user only provided
one review, we could still mine very useful information from that review, such as having traveled in summer with family members, liking city views at night, and being concerned about the cleanliness of the room. This means that, if we can effectively extract such useful information from this review, we might be able to successfully model the user preferences and use them to construct a high-quality user representation.

Many deep-learning techniques have been adopted to model user and item representations from reviews [4]–[7]. For example, DeepCoNN [4] applied a convolutional neural network (CNN) [28] to learn such representations, which were then used to predict ratings based on the latent factor model. In an extension to the CNN, NARRE [7] applied an attention mechanism [29] to construct representations by considering different contributions from reviews based on their usefulness. Despite variations, these techniques share a common network framework for constructing representations, which is shown schematically in Fig. 3. To learn a representation for user \( u_i \), this technique first creates a user document for \( u_i \) by concatenating all the user’s previous reviews. Each of the \( M \) words in \( u_i \)’s document is then looked up and mapped with its word embedding, which can be initialized randomly or by utilizing a pretrained word embedding such as Word2Vec [30], GloVe [31], or BERT [32]. These word embeddings are then fed into the neural network components to learn \( x_{ui} \) as a representation of \( u_i \). Note that an item representation can be constructed in the same way as a user representation. The output of such a framework is a static representation for every user and item in the training data.

Second, to predict a rating for a review \( y_{ji} \), the representations of \( u_i \) and \( v_j \) are not (but should be) dynamically constructed by emphasizing the relevant information in the review, which explains the reason behind its rating, rather than relying on information from other reviews.

Extracting and exploiting contexts from reviews could be the key to overcoming these two limitations. Context words such as “friend” or “summer” are often related to a user preferences and an item features, and are therefore appropriate for use in constructing representations. Moreover, contexts help characterize the situation within which the rating was being given, which is unique and specific for each review. This means that user and item representations can be constructed dynamically by considering the contexts embedded in each particular review.

**C. CONTEXT-AWARE RECOMMENDATIONS BASED ON USER REVIEWS**

There are two main approaches to extracting contexts from reviews, namely supervised and unsupervised approaches. A supervised approach extracts contexts based on a predefined list of contextual variables and their corresponding values [12]–[17]. Using the predefined contexts in Table 1, words such as “summer”, “family”, or “night” could be extracted as contexts from the review in Fig. 1. However, non-predefined words in Fig. 1 such as “clean”, “free-wifi”, or “breakfast”, which could potentially be considered as contexts, are overlooked. For a supervised approach to be robust, therefore, it will require the contextual variables and their corresponding values to be predefined optimally for each specific recommendation domain.

In contrast, an unsupervised approach aims to infer contexts from reviews without having to redefine them [20]–[22]. Some of these approaches [20], [21] classify reviews into context-rich and context-free reviews, based on features of each review such as the number of words, verbs, and verbs in the past tense. The contexts are then extracted as those words or topics that occur more often in the context-rich reviews. These two methods, however, require manual annotation of the review data (as context or noncontext) as part of the training process. Recently, CARL [22] has applied CNN and word-level attention to semantically infer contexts from reviews. Its user and item representations are constructed by modeling the attention weight of each word as its influence in each context on a user–item pair. This method was, however, presented using the framework shown in Fig. 3, which means that it suffers from the limitations of utilizing irrelevant words and constructing only static representations.

Most context extraction methods [12]–[17], [20] define and extract a context in the form of a single word such as those shown in Table 1. However, when users write reviews, they have flexibility in how their contexts are presented, including using phrases in addition to single words. For example, some contexts from the review in Fig. 1 might be best extracted as “family trip”, “night city view”, or “friendly staff”, which are more meaningful than just “family”, “night”,

---

**FIGURE 3.** A neural network framework for constructing user representations from reviews.

As mentioned in Section I, constructing a representation using this framework has two limitations. First, words that are not related to a user preferences or an item features are (but should not be) used in constructing the representations.
or “friendly.” We believe that other words that often accompany (or are present in the same text region as) context words might help in capturing the appropriate meaning of contexts, and should therefore also be extracted to represent contexts accurately.

In our previous work [23], we proposed an unsupervised method for extracting contexts from reviews by utilizing a technique called “region embedding”, which had been introduced previously [24]. Our model has the ability to identify and construct a relevant context not only in single word format but also by including adjacent or nonadjacent words that occupy the same text region and that influence the distributions of ratings. In this paper, we aim to extend the model to include an efficient method for utilizing the extracted contexts when constructing user and item representations and using them to make rating predictions.

III. PROPOSED METHOD

In this section, we provide a detailed explanation of CARE, including our proposed method for learning user and item representations, together with our previous context extraction method. Figure 4 presents an overview of the workflow of CARE, which comprises two main parts, namely context extraction and rating prediction. In context extraction, we first identify a list of candidates to be considered as context words, extract their associated text regions, and generate their rating distributions. In rating prediction, we first construct dynamically the user and item representations from the region embeddings via our proposed interaction and attention modules. The interaction module models the relevance of each context to the individual user and item by its past interaction with the user preferences and the item features. The attention module then generates user and item representations based on the different relevance levels among contexts extracted from a particular review. Finally, these representations are used in a latent factor model that predicts the rating.

A. CONTEXT EXTRACTION

We first present our procedure for extracting contexts from reviews, which comprises three main steps, namely identifying candidate context words, extracting contextual regions, and learning the region embeddings.

1) IDENTIFYING CANDIDATE CONTEXT WORDS

We adopt the definition used in Odić et al. [19], which defines relevant contexts as those that contribute to explaining the variance in ratings. By applying this definition to review data, a context can be considered as any word in the reviews that influences the distribution of ratings. For example, Fig. 5 presents a word-rating co-occurrence matrix, which gives the word frequency for each rating value. From this figure, words such as “clean” or “good” have frequent mentions in reviews with more positive rating scores such as “4” and “5”, whereas “dirty” or “not” were mentioned more frequently for more negative scores such as “1” or “2”. The implication is that these words influence the distribution of ratings, and could therefore be considered as “candidates” for contexts.

To measure the significance of the influence of a word on the distribution of ratings, we first compute its variance on such a distribution, as also shown by the example in Fig. 5. After computing variances for every word in the review corpus, only those words having variances above our predefined minimum-variance threshold \( \min_{\text{var}} \) are selected as candidate context words and stored in the candidate list \( \text{Cand} \). Note that, in addition to direct context words (e.g. “clean”), \( \text{Cand} \) also includes opinion or sentiment words such as “good” or “not”.

---

**FIGURE 4.** An overview of the workflow architecture of CARE.

**FIGURE 5.** Example of word-rating co-occurrences and their corresponding variances.
if they also have significant influence on the distributions of ratings.

2) EXTRACTING CONTEXTUAL REGIONS

Depending solely on the candidate context words might not be sufficient to cover the variety of influences of contexts on the distributions of review ratings. This is because some words that often accompany candidate context words (neighboring words) might significantly alter the ways they influence the distributions of ratings. For example, as shown in Fig. 5, the word “service” has a mixed frequencies of rating scores “2”, “3”, and “4”, indicating a neutral distribution toward middle-rank ratings. However, if we consider its co-occurrence with the word “good” (i.e., “good service”), the rating distribution could change from neutral to strongly positive, whereas “worst service” could result in a strongly negative distribution, as shown in Fig. 6. This means that neighboring words might be opinion, sentiment, or other words that could change the semantic meaning of a candidate context word, and therefore influence their rating distributions. This emphasizes the importance of considering neighboring words in addition to the candidate context words if the modeling of the influence of contexts is to be effective.

Consider a candidate context word \( c_n \in \text{Cand} \). We define the neighboring words of \( c_n \) as any word \( w_l \) that occupies the same “text region” of \( c_n \). More specifically, we consider \( w_l \in \text{region}(c_n, d) \), where \( d \) is the window size for a region of length \( 2 \times d + 1 \). We call \( \text{region}(c_n, d) \) a contextual region of \( c_n \). Note that \( w_l \) can be in any position within \( \text{region}(c_n, d) \), not necessarily directly adjacent to \( c_n \). This takes account of the different writing styles users may adopt for the same meaning in writing reviews. For example, “view of a city at night” and “night view of a city” are simply alternative expressions of the same context.

We, therefore, formally define a relevant context in our work as “any word in a region of text that has an influence on the distributions of ratings.” Such context can be in a format of a single word, adjacent words, or even nonadjacent words, provided they occupy the same text region. To identify the positions of these words, we first need to extract their associated contextual regions. Let \( \text{Region} \) denote a set of all contextual regions extracted from reviews and let \( \text{region}(c_n, d)_m \in \text{Region} \) be a contextual region at index \( m \) in \( \text{Region} \). The positions of words to be constructed as context can then be identified by the following steps.

1) Generate all possible combinations of words \( w_l \in \text{region}(c_n, d)_m \) of size \( \theta \) (where \( \theta \leq 2 \times d + 1 \)) that include \( c_n \), denoted by \( \delta(c_n, w_l)_m \).

2) Count the number of times each combination in \( \delta(c_n, w_l)_m \) co-occurs in the same region with each rating value on the entire training data and compute the variance from the frequency distribution of ratings.

3) Choose the combination that contributes the highest variance in rating distribution as context for \( \text{region}(c_n, d)_m \). If no combination has a variance above \( \min_{\text{var}} \), \( c_n \) alone is considered as context for that region. Store the rating distribution of this combination, \( \text{dist}(c_n, d)_m \in \mathbb{R}^{[\text{Rating}]} \) in the list of rating distributions \( \text{Dist} \) for index \( m \).

Figure 7 illustrates the procedure for identifying the highest contributed variance combination of size \( \theta = 2 \) for the region “bed is clean and cozy.”. Since, (“clean”, “cozy”) yields the highest variance, it is chosen to represent context for this region.

3) LEARNING THE REGION EMBEDDINGS

From the previous step, we are able to extract the contextual regions \( \text{Region} \) and their associated rating distributions \( \text{Dist} \) from the review data. We now want to utilize them to train our predictive model so that, given a contextual region \( \text{region}(c_n, d)_m \) as an input, it predicts the rating distribution \( \text{dist}(c_n, d)_m \) as an output. To achieve this, a model with an ability to identify those words in \( \text{region}(c_n, d)_m \) that contribute to \( \text{dist}(c_n, d)_m \) is required. We therefore adopt the model used for region embedding with local context proposed by Qiao et al. [24] as our training model. This technique learns two representations for each word, namely a word embedding of itself and a local contextual unit, which is a weight matrix for its interaction with its neighboring words. Our aim is that the local context unit can be modified to emphasize the positions of the words that have influence on the rating distributions and can therefore be considered as contexts for each contextual region.

Our derived region embedding method for context extraction in CARE is shown in Fig. 8. This technique is a simple feedforward neural network model that takes a text region \( \text{region}(c_n, d)_m \) as an input and produces a rating distribution vector \( \text{dist}(c_n, d)_m \) as an output. Every word \( w_l \in \text{Vocab} \) is mapped to its word embedding, whereas only a candidate context word \( c_n \in \text{Cand} \) is mapped to its local context unit, which is used to produce the projected word embeddings. Finally, a region embedding, which is a representation of a text region, is generated from the projected word embeddings for use in computing a rating distribution.

Formally, every word \( w_l \) has an associated word embedding \( \mathbf{e}_{w_l} \), which is stored in the column of the embedding matrix \( \mathbf{E} \in \mathbb{R}^{h \times |\text{Vocab}|} \), where \( h \) is the embedding size and \( \text{Vocab} \) is the vocabulary of all words in the training data. In addition to the word embeddings, a candidate context word \( c_n \) also has an associated local-context unit matrix \( \mathbf{K}_{c_n} \in \mathbb{R}^{h \times (2 \times d + 1)} \), which is stored in the tensor \( \mathbf{C} \in \mathbb{R}^{h \times (2 \times d + 1) \times |\text{Cand}|} \).

Given a contextual region \( \text{region}(c_n, d)_m \) as an input, the projected word embedding \( \mathbf{p}_{w_l} \) of word \( w_l \) at index position \( l \) of \( \text{region}(c_n, d)_m \) is calculated by

\[
\mathbf{p}_{w_l} = \mathbf{K}_{c_n,l} \odot \mathbf{e}_{w_l}.
\]

FIGURE 6. Example of rating co-occurrences with two bigrams containing the word “service”.

| bigram          | rating ‘1’ | rating ‘2’ | rating ‘3’ | rating ‘4’ | rating ‘5’ | variance   |
|-----------------|------------|------------|------------|------------|------------|------------|
| “good service”  | 7          | 15         | [24]       | 285        | 311        | 21103.3    |
| “worst service” | 351        | 291        | 78         | 8          | 1          | 26935.7    |
Finally, $\mathbf{y}_{c_n,m}$ is fed into the fully connected layer to calculate the rating distribution $\text{dist}(c_n, d)_m$. Its objective is to predict a vector of rating distributions and we adopt a multivariate linear-regression model for the prediction, as expressed by

$$\text{dist}(c_n, d)_m \approx \mathbf{W}_f \cdot \mathbf{y}_{c_n,m} + \mathbf{b}_f. \quad (5)$$

Here, $\mathbf{W}_f \in \mathbb{R}^{\text{Rating} \times h}$ and $\mathbf{b}_f \in \mathbb{R}^{\text{Rating}}$ are the weight matrix and bias vector in the fully connected layer, respectively, where $\text{Rating}$ is the size of the categorical rating scores (e.g. $\text{Rating} = 5$ for a five-point rating score). We chose L2 as the loss function, following Qiao et al. [24], and Adam [33] as the optimizer. No regularization was applied.

After all model parameters are learned, each contextual region $\text{region}(c_n, d)_m$ can now be mapped with its region embedding representation $\mathbf{y}_{c_n,m}$. Because the region embeddings are trained to represent the rating distributions of their associated contextual regions, we will discuss the quality of this representation further in Section V-D.

In the next section, we show how the extracted contextual regions and their associated region embeddings can be utilized for rating predictions in CARE.

### B. RATING PREDICTION

The previous section described the extraction of contexts from reviews as contextual regions and their representation in terms of their associated region embeddings. In this section, we present an efficient rating prediction method for CARE that utilizes the extracted contexts to dynamically construct user and item representations. First, we give an overview and brief explanation of the architecture of our model. This is followed by a more detailed explanation of the proposed interaction and attention modules that are used in constructing the representations for users and items. We then describe the final step of the procedure, in which the rating of a particular review can be predicted using the derived representations.

#### 1) MODEL OVERVIEW

To utilize the region embeddings for making personalized rating predictions, it is important to model the influence of each contextual region on the user preferences and the item features, which are used in determining a particular review’s rating. To achieve this, we consider two important aspects of each contextual region, namely its relevance to the region embeddings are trained to represent the rating distributions of their associated contextual regions, we will discuss the quality of this representation further in Section V-D.

In the next section, we show how the extracted contextual regions and their associated region embeddings can be utilized for rating predictions in CARE.
each user preferences or item features and its contribution to a particular review’s rating. First, the relevance of each contextual region to a user’s personal preferences depends on how it has previously been expressed by that user in previous reviews. For example, many of the user’s hotel reviews might have contained words such as “cheap”, “expensive”, or “worth”, whereas “small”, “large”, or “spacious” might have been used less often. The implication is that this user is highly interested in the price when choosing a hotel, but is less concerned about the size of the room. Therefore, those contextual regions containing words related to the price of a room should be more relevant to this user preferences than those containing words related to the size of a room. The same assumption can also be applied to the relevance of contextual regions to the item’s unique features. Depending on what has been frequently described in their reviews, some hotels, for example, might be famous for their service, whereas others are better known for their convenient location.

Moreover, a review usually contains more than one contextual region and different contextual regions might have unequal influences on the user’s decision about the item, which would consequently affect the rating. We believe that, the more relevant a contextual region is to each individual user preferences or item features, the more it should contribute to the rating of a particular review compared with the other regions within the same review.

By properly analyzing the relevance and the contribution made by contexts in a particular review, we are able to dynamically construct user and item representations that are specific and unique for that review. To implement this, we propose a rating prediction model, called the attentional interaction model for CARE (CARE-AttnIntr), whose architecture is shown schematically in Fig. 9. Our model is composed of two parallel neural networks, one each for user-context and item-context modeling. The model takes a review of user $u_i$ on item $v_j$, denoted by $y_{i,j}$, that contains $M$ contextual regions, denoted by $\text{Region}(y_{i,j})$, as an input. By looking up its corresponding word embeddings and local context unit (learned from the context extraction step), a region embedding for each contextual region is generated. The region embeddings are then fed into the user-context and item-context modeling networks. Each of these networks comprises two modules, namely an interaction module that models the relevance of contextual regions and an attention module that learns the contributions of those regions to a review’s rating. Finally, the outputs of the user-representation and item-representation networks are fed into the prediction layer, which generates the final prediction of a review’s rating by using a latent factor model.

2) INTERACTION MODULE

To model the relevance of contextual regions to user preferences and item features, we introduce a user-context interaction matrix $T_u \in \mathbb{R}^{\frac{\text{User}}{h} \times h}$ and a item-context interaction matrix $T_v \in \mathbb{R}^{\frac{\text{Item}}{h} \times h}$. Each row $t_u \in T_u$ and $t_v \in T_v$ contains a vector representing the interaction with the contextual regions for user $u_i$ and item $v_j$, respectively. To fully capture the interactions with the contextual regions, the dimensionalities of $t_u$ and $t_v$ are set to $h$, which is the dimensionality of the region embedding. We then model the interaction of contextual region $\text{region}(c_n, d_m) \in \text{Region}(y_{i,j})$ with user $u_i$ and item $v_j$ by using element-wise multiplication between its region...
embedding \( \mathbf{y}_{c,a,m} \) and \( \mathbf{t}_{u} \) or \( \mathbf{t}_{v_j} \), respectively, as expressed by
\[
\mathbf{y}_{c,a,m,i} = \mathbf{t}_{u} \odot \mathbf{y}_{c,a,m}, \quad \mathbf{y}_{c,a,m,j} = \mathbf{t}_{v_j} \odot \mathbf{y}_{c,a,m}. \tag{6}
\]

The vectors \( \mathbf{t}_{u} \) and \( \mathbf{t}_{v_j} \) can be considered as projection vectors for converting the region embedding \( \mathbf{y}_{c,a,m} \) to the user-relevance region embedding \( \mathbf{y}_{(c,a,m),i} \) and item-relevance region embedding \( \mathbf{y}_{(c,a,m),j} \), respectively. They are learned with the main objective of capturing previous interactions of a contextual region with each individual user preferences and a specific item features. If a contextual region \( \text{region}(c_a,d)_m \) was mentioned a significant number of times in user \( u_i \)'s reviews, its interaction with \( \mathbf{t}_{u} \) will result in high values for \( \mathbf{y}_{(c,a,m),i} \), indicating that it is highly relevant to \( u_i \)'s preferences.

After all region embeddings \( \mathbf{y}_{c,a,1}, \cdots, \mathbf{y}_{c,a,M} \) of \( \text{region}(c_a,d)_1, \cdots, \text{region}(c_a,d)_M \in \text{Region}(y_{i,j}) \) are converted into user-relevance and item-relevance region embeddings, they are fed into the attention module to compute their contributions to a rating of the review \( y_{i,j} \).

3) ATTENTION MODULE

The contribution of a contextual region \( \text{region}(c_a,d)_m \) to a particular review’s rating depends on its degree of relevance to the user preferences and the item features compared with the other regions in that review. Because a user-relevance region embedding \( \mathbf{y}_{(c,a,m),i} \) and an item-relevance region embedding \( \mathbf{y}_{(c,a,m),j} \) indicate the relevance of \( \text{region}(c_a,d)_m \) to \( u_i \)'s preferences and \( v_j \)'s features, we can utilize them for modeling their contributions. We adopt an attention mechanism that has been successfully utilized in many deep-learning-based recommendation methods [6, 7]. Specifically, the attention network for modeling the contribution of \( \text{region}(c_a,d)_m \) in a user-context modeling network is defined by
\[
a_{y_{(c,a,m),i}}^u = \mathbf{W}_{\text{attu}}^T \mathbf{W}_{\text{attn}} \mathbf{y}_{(c,a,m),i} + b_{\text{attu}} + b_{\text{attn}}, \tag{7}
\]
where \( \mathbf{W}_{\text{attn}} \in \mathbb{R}^{k_1 \times h}, \mathbf{W}_{\text{attu}} \in \mathbb{R}^{k_1 \times h}, b_{\text{attn}} \in \mathbb{R}^{k_1}, \) and \( b_{\text{attu}} \in \mathbb{R} \) are model parameters. The size of the hidden layer in the attention network is denoted by \( k_1 \) and \( g \) is a nonlinear activation function. We then apply a softmax function to compute a normalized attention score for a contextual region \( \text{region}(c_a,d)_m \) with respect to the other \( M-1 \) contextual regions in review \( y_{i,j}, \text{Region}(y_{i,j}) \) as
\[
a_{y_{(c,a,m),i}}^u = \frac{\exp(a_{y_{(c,a,m),i}}^u)}{\sum_{m=1}^{M} \exp(a_{y_{(c,a,m),i}}^u)}. \tag{8}
\]

The attention scores are then used to compute the weighted sum of the user-relevance region embeddings, which are fed into a fully connected layer to create a user representation of user \( u_i \) that is specific to rating \( y_{i,j}, x_{u_i,y_{i,j}}, \) as expressed by
\[
x_{u_i,y_{i,j}} = \mathbf{W}_u \sum_{m=1}^{M} a_{y_{(c,a,m),i}}^u \mathbf{y}_{(c,a,m),i} + b_u, \tag{9}
\]
where \( \mathbf{W}_u \in \mathbb{R}^{k_2 \times h} \) and \( b_u \in \mathbb{R}^{k_2} \) are a weight matrix and bias vector, respectively, in a fully connected layer with a hidden layer of size \( k_2 \). This user representation is dynamically generated to give different levels of relevance among the contexts in \( y_{i,j} \). A user who has rated \( N \) reviews will therefore have \( N \) review-based representations, rather than the one static representation across all reviews used in most deep-learning-based methods. (We omit the computational details for the item representation \( x_{v_j,y_{i,j}} \) because they are very similar to those for the user representation.)

4) PREDICTION LAYER

We now describe how our user and item representations can be used for the final rating prediction task. We adopt a latent factor model [25], which has been shown to be effective in many deep-learning-based prediction approaches [4, 5, 7], to model the interaction between \( x_{u_i,y_{i,j}} \) and \( x_{v_j,y_{i,j}} \). Specifically, the rating of user \( u_i \) toward item \( v_j \) is estimated by
\[
\hat{r}_{i,j} = x_{u_i,y_{i,j}}^T x_{v_j,y_{i,j}} + b_u + b_v + \mu, \tag{10}
\]
where \( b_u, b_v, \) and \( \mu \in \mathbb{R} \) respectively denote the bias for user \( u_i \), the bias for item \( v_j \), and the global bias. If a contextual region in \( y_{i,j} \) is highly relevant to both user \( u_i \)'s preferences and item \( v_j \)'s features, it should result in a high rating score. For example, (10) implies that a user who likes a city view at night will be recommended a hotel famous for its nighttime city view. We use Adam [33] as our optimizer, and, to prevent overfitting, we apply a dropout operation on the hidden layer. We also use L2 as our loss function, with regularization expressed as
\[
L = \sum_{(i,j) \in O} (r_{i,j} - \hat{r}_{i,j})^2 + \lambda \| \Theta \|^2, \tag{11}
\]
where \( O \) denotes the set of observed user–item rating pairs, \( r_{i,j} \) is the observed rating score of user \( u_i \) toward item \( v_j \), and \( \Theta \) denotes our model parameters.

IV. EXPERIMENTAL EVALUATION

A. DATA PREPARATION

For our experiments, we used three publicly available datasets. The first was from TripAdvisor,\(^1\) which contains hotel review data. The two other datasets, Amazon Software and Amazon Movies & TV, are members of the Amazon 5-core datasets\(^2\) [34]. All datasets use a five-point rating system (users select an integer from 1 through 5). We performed some preprocessing on the review text for these two datasets as follows: 1) tokenize the review text and convert all words to lower case; 2) remove all punctuation marks and infrequently used words (i.e., those of appearance frequency below 0.01% in all reviews); 3) remove all stopwords listed by NLTK;\(^3\) except for those indicating sentiment meanings such as “very” or “not”. After the preprocessing, the one-word reviews were discarded as uninformative. The statistics of these two datasets after preprocessing are summarized

---

1. http://www.cs.cmu.edu/~jiweil/html/hotel-review.html
2. https://nijianmo.github.io/amazon/index.html
3. https://www.nltk.org/nltk_data/
TABLE 2. Statistics for the three review datasets.

| Dataset          | Reviews | Users | Items | Density | Rates/User | Rates/Item | Average Rating | Vocab. Size | Word/Review | Candidates | Cand./Review |
|------------------|---------|-------|-------|---------|------------|------------|----------------|-------------|-------------|------------|--------------|
| Amazon Software  | 12,405  | 1,802 | 801   | 0.008594| 6.884      | 15.487     | 3.871          | 35,103     | 104,193     | 257        | 37,369       |
| TripAdvisor      | 873,214 | 574,450| 2,940 | 0.000386| 1.388      | 221,628    | 3.937          | 22,353     | 82,984      | 3,012      | 31,931       |
| Amazon Movies & TV| 3,166,002| 293,867| 60,169| 0.000179| 10.774     | 52.618     | 4.202          | 20,949     | 47,472      | 128        | 47,472       |

These datasets differ in some respects. For example, Amazon Software is the smallest and densest, whereas Amazon Movies & TV is the largest and also the sparsest among the three datasets. Most of the reviews in all datasets were rated with very high scores, with Amazon Movies & TV having the highest average score. Despite being the smallest dataset, Amazon Software contains the highest number of unique words (i.e., vocabulary) in its reviews and also has the longest reviews, on average. This contrasts with the Amazon Movies & TV dataset, which is the largest dataset, but which contains the smallest vocabulary and the shortest reviews.

After applying our context extraction method, the number of candidate context words and their frequencies per review were found to correspond with the size of the vocabulary and the average number of words per review for each dataset.

B. BASELINES

To establish baselines for evaluation, we compared the performance of our proposed CARE with seven existing state-of-the-art rating prediction models: PMF, NMF, RC-Topic, RC-Word, DeepCoNN, NARRE, and CARL. The comparative characteristics of the baselines and our proposed CARE are listed in Table 3. The first two methods involve latent factor models that utilize ratings alone to learn user and item representations. The remaining methods incorporate review data to learn such representations, except for RC-Topic and RC-Word, which learn a representation from each review. DeepCoNN and NARRE are review-based methods that do not consider contexts in reviews. RC-Topic, RC-Word, CARL, and our CARE consider the influence of contexts in reviews on rating predictions. More details about each method are summarized as follows.

TABLE 3. Comparison of the characteristics of all methods.

| Method | Rating | Review | Represent | Attention | Contexts | Interaction |
|--------|--------|--------|-----------|-----------|----------|-------------|
| PMF    | ✓      |        | User/Item | -         | -        | -           |
| NMF    | ✓      |        | User/Item | -         | -        | -           |
| RC-Topic| ✓      | ✓      | Review    | Topic     | -        | -           |
| RC-Word| ✓      | ✓      | Review    | Word      | -        | -           |
| DeepCoNN| ✓      | ✓      | User/Item | -         | -        | -           |
| NARRE  | ✓      | ✓      | User/Item | Word      | -        | -           |
| CARL   | ✓      | ✓      | User/Item | Word      | Region   | Region      |
| CARE   | ✓      | ✓      | User/Item | Region    | Region   | Region      |

C. EXPERIMENTAL SETTINGS

For our evaluations, we randomly selected 80% of each dataset as the training set, 10% as the validation set, and the remaining 10% as the test set. Because our model includes two main parts (context extraction and rating prediction), we required different experimental settings.

1) SETTINGS FOR CONTEXT EXTRACTION

To extract a list of candidate context words, we first created a word-rating co-occurrence matrix for each dataset. Here, the main problem is that many datasets contain biases in the proportion of ratings provided by users. For example, in the TripAdvisor dataset, more than 80% of all reviews were rated as “4” or “5”, meaning that most users preferred to provide high rating scores to most hotels. This causes almost every word in the corpus to be distributed toward high rating scores, as shown by the example in Fig. 10 (a). This is in contrast to the fact that reviews containing words such as “rude” should be rated with a low rating score, rather than a high rating score. To properly analyze the actual influence of a word on the rating distribution, we therefore applied a data standardization technique, expressed by

\[ x_{t,r}^{new} = \frac{x_{t,r} - \mu_r}{\sigma_r}, \]
where \( x_{r,t} \) is the original frequency of word \( w_t \) given for rating \( r \), \( \mu_r \) is the average of the frequencies of all words given for rating \( r \), and \( \sigma_r \) is the standard deviation of the frequencies of all words given for rating \( r \). The rating distribution after applying this standardization is shown in Fig. 10(b). The frequencies of ratings with the word “rude” are now distributed toward low rating scores, which is appropriate to its negative meaning.

After standardization, we computed the variance of the rating distribution of each word, and selected the words with variances exceeding \( \min_{var} = 1 \) as a set of candidate context words for each dataset. The numbers of such words extracted from each dataset and the average number for each review are given in Table 2. To extract the contextual regions for each candidate context word, we set a region size of five and applied a padding of length \( d = 2 \) to the head and tail of every review (a candidate context word might be the first or last word in a review). Because some candidate context words might be associated with millions of contextual regions, using all of them for training could cause scalability problems. In fact, this is unnecessary because using only a sampled portion can cover all unique patterns in the rating distributions. We therefore defined a criterion for sampling a subset of the contextual regions for a candidate context word \( c_n \). Specifically, let \( \text{Region}_{c_n} \) denote the set of all contextual regions of \( c_n \). If \( |\text{Region}_{c_n}| > 100k \), only a 10% subset was used for training. If \( 10k \leq |\text{Region}_{c_n}| \leq 100k \), 10k were used. If \( |\text{Region}_{c_n}| < 10k \), all were used. To assign a rating distribution to each contextual region, we set the size of word combination \( \theta = 2 \), and selected those with variances exceeding \( \min_{var} = 1 \).

In the training process, our word embeddings \( E \) and local context units \( K \) were initialized randomly in terms of a uniform distribution with values between \(-1\) and \(+1\). The embedding size \( h \) for all datasets was set to 300. The learning rate was optimized from \([0.0001, 0.001]\), and the batch size was selected from \([128, 256, 512, 1024, 2048]\), using a validation set.

2) SETTINGS FOR RATING PREDICTION

Because the predictions from all comparative methods are based on the latent factor model, they require a significant number of ratings per user or per item to learn high-quality representations. This significant number was set as 5 for the Amazon Software and TripAdvisor datasets and 20 for the Amazon Movies & TV dataset. We therefore eliminated those reviews that did not meet this significance criterion. Furthermore, we assumed that the review texts would be available for both training and testing stages because this is exploited by RC-Topic, RC-Word, and CARE when extracting contexts and making rating predictions.

For PMF and NMF, the number of latent dimensions was set to 15, the learning rate to 0.005, and the regularization parameter to 0.001.

For RC-Topic and RC-Word, we separated the reviews by applying K-means clustering for the set of review features defined by Bauman and Tuzhilin [20], including the number of words, number of verbs, and number of verbs in the past tense. The number of top contextual topics was selected from \([20, 30, 50, 100, 200]\), and the number of top contextual words was selected from \([1000, 2000, 5000, 10000]\). We used LibFM [35] to implement the FM, following Peña [21]. All FM parameters were set to default values.

For DeepCoNN, NARRE, and CARL, the number of convolutional kernels was selected from \([50, 100]\), the window size for CNN was set to 3, the learning rate was selected from \([0.0001, 0.0005, 0.001]\), the regularization parameter was selected from \([0.001, 0.01, 0.1]\), and the dropout rate was optimized between 0.1 and 0.5. The number of latent dimensions was set to 32 and the embedding size was set to 300 for all these models.

For the prediction model in CARE, the learning rate was selected from \([0.0001, 0.0005, 0.001, 0.005, 0.01]\), the regularization parameter \( \lambda_w \) was selected from \([0.01, 0.1, 1]\), and the dropout rate was set to 0.2. The numbers of latent dimensions \( k_1 \) and \( k_2 \) were both set to 32. We selected the hyperbolic tangent (tanh) as the activation function for (7). Because different reviews contain different numbers of contextual regions, we set 128 as the maximum number of contextual regions that would be extracted from each review. The batch sizes for both the proposed model and the baseline models were optimized from among \([16, 32, 64, 128, 256]\).

3) EVALUATION METRICS

We evaluated the performance of CARE against the baseline systems in terms of prediction accuracy using three ranking evaluation metrics. The first evaluation metric was the normalized discounted cumulative gain (NDCG), which evaluates the ranking accuracy of the Top-\(K\) recommendation list for each user, as expressed by

\[
\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}
\]

where \( \text{IDCG@K} \) is the ideal DCG@K. The second metric was the mean average precision (MAP), which measures the average precision of all queries. The third metric was the hit rate (HR), which measures the proportion of relevant items that are recommended.

Here, we set \( rel_j \in \{1,2,3,4,5\} \) as the actual rating scores for an item at rank position \( j \). Because most users in Amazon Software and TripAdvisor had rated items less than seven times, we chose NDCG@3, NDCG@5, and NDCG@7 for their evaluation, whereas for Amazon Movies & TV, with more than 10 ratings per user, we chose NCDG@5, NCDG@10, and NCDG@15 for that dataset.
TABLE 4. The NDCG values for the compared methods on the three review datasets.

|               | Amazon Software |                     | TripAdvisor |                     | Amazon Movies & TV |                     |
|---------------|-----------------|---------------------|-------------|---------------------|-------------------|---------------------|
|               | @3              | @5                  | @7          | @3                  | @5                | @7                  | @10                  | @15                  |
| PMF           | 0.89160         | 0.90009             | 0.87302     | 0.87673             | 0.78935           | 0.77752             | 0.90051              | 0.87458              | 0.86025              |
| NMF           | 0.88848         | 0.84849             | 0.81870     | 0.88341             | 0.82420           | 0.80706             | 0.89939              | 0.87110              | 0.85774              |
| RC-Topic      | 0.89944         | 0.95023             | 0.93930     | 0.91597             | 0.86226           | 0.84023             | 0.91223              | 0.89489              | 0.88759              |
| RC-Word       | 0.93513         | 0.93698             | 0.89889     | 0.89744             | 0.81977           | 0.80363             | 0.90175              | 0.87999              | 0.86520              |
| DeepCoNN      | 0.92396         | 0.92357             | 0.86373     | 0.92963             | 0.87715           | 0.87026             | 0.94070              | 0.92895              | 0.92294              |
| NARRE         | 0.93901         | 0.93200             | 0.86880     | 0.92841             | 0.89495           | 0.87020             | 0.94018              | 0.92757              | 0.91909              |
| CARL          | 0.92811         | 0.91501             | 0.91289     | 0.92335             | 0.86649           | 0.85077             | 0.92798              | 0.91375              | 0.90308              |
| CARE          | 0.92224         | 0.98448             | 0.97039     | 0.95147             | 0.91826           | 0.93168             | 0.91707              | 0.93527              | 0.92328              |

In addition to NDCG, we also evaluated the performances by calculating the hit ratio (HR) and mean reciprocal rank (MRR). We first classified reviews in our test data that had ratings above “3” as positive reviews, and the remainder as negative reviews. An HR@$K$ is calculated as the number of positive reviews appearing in the Top-$K$ recommendation list for each user, whereas MRR is computed as the rank of the first positive review in each user recommendation list, as given by (14) and (15), respectively.

\[
HR@K = \frac{\sum_{j=1}^{K} rel_j}{K}
\]

\[
MRR = \frac{1}{|User|} \sum_{u \in User} \frac{1}{rank_{u_i}}
\]

Here, we set \(rel_j = 1\) for a positive review and 0 otherwise. The value for \(rank_{u_i}\) is the first rank position of a positive review in \(u_i\)’s recommendation list. We chose HR@5 in evaluating the performance for all datasets.

D. EXPERIMENTAL RESULTS

The values for NDCG, HR, and MRR for all baseline systems and CARE with each of the three datasets are presented in Table 4 and Table 5. From Table 4 and 5, note that CARE achieves the highest accuracy for almost every MRR and rank of NDCG and HR across all datasets. Next are the deep-learning-based methods that utilize review data (DeepCoNN, NARRE, and CARL). Although CARL, which is a context-aware method, performs quite well on Amazon Software, DeepCoNN and NARRE perform better on TripAdvisor and Amazon Movies & TV datasets. DeepCoNN and NARRE obtain very similar results across all datasets, although DeepCoNN seems to perform slightly better on TripAdvisor and Amazon Movies & TV. Furthermore, the other two context-aware baseline systems (RC-Topic and RC-Word) achieved quite good NDCG values on Amazon Software but were less effective on the other two datasets, as was CARL. Finally, PMF and NMF, which do not consider review information, yielded the lowest accuracies on all evaluation metrics when compared with the other systems.

V. DISCUSSION

In this section, we provide a detailed analysis of the performance of the proposed method and the baseline systems. We consider first the predictive performance, followed by a detailed analysis of our model in various aspects. We then review the list of contexts extracted from various recommendation domains. We analyze the quality of the region embeddings learned by our extraction method to investigate its suitability for the rating prediction task. Finally, we analyze the merits of defining and extracting contexts as contextual regions in terms of an illustrative example and a sentiment classification task.

A. PREDICTIVE PERFORMANCE

Here, we discuss three main aspects of predictive performances. First, we analyze the effectiveness of leveraging review content in making predictions. We then analyze how identifying and incorporating contexts from reviews affect the prediction accuracy. Finally, we discuss how the dynamic modeling of user and item representations differs from and improves on the use of the static-representation approach.

1) UTILIZING REVIEW DATA

We first discuss the merit of utilizing review data for making recommendations. As shown by Tables 4 and 5, all methods that leverage review content to learn user and item representations (DeepCoNN, NARRE, CARL, and our CARE) returned better values for NDCG, HR, and MRR than the standard CF-based methods that ignore reviews (PMF and NMF). This demonstrated that the rich and useful information embedded in reviews helps the learning of more appropriate representations, which more accurately capture the personal preferences of a user or the unique features of an item. Utilizing these representations consequently resulted in more accurate rating predictions.

Although leveraging review data, the RC-Topic and RC-Word methods did not always provide better prediction accuracy than the standard CF-based methods that did not...
consider reviews. For example, they gave lower HR and MRR values on the TripAdvisor dataset than PMF and NMF. This might be because both RC-Topic and RC-Word utilize review content to learn a representation of the review itself and use it directly for rating prediction, rather than learning representations for users and items based on a latent factor model, as do the other methods. We believe that their review representations do not capture the personalized information relating to user preferences or item features, which makes their predictions less accurate than latent factor-based methods.

2) INCORPORATING CONTEXTUAL INFORMATION

We now analyze the effect of incorporating contextual information hidden in reviews on the rating prediction. By comparing the results of the review-based context-aware methods (RC-Topic, RC-Word, CARL, and CARE), only CARE could surpass the accuracy of the review-based methods that did not consider contexts (DeepCoNN and NARRE).

We begin with an analysis of RC-Topic and RC-Word, which are review-based context-aware methods based on topic modeling and TF-IDF representations. Although they mostly performed better than the standard-CF based methods (PMF and NMF), they were the least accurate of the review-based methods. This might be because the performance of their context extraction depends on the quality of the reviews. Based on the definition used in RC-Topic and RC-Word, contexts can be inferred only from reviews of high quality, which often contain a significant number of words. For review data containing many low-quality reviews, the context extraction would be less effective. In contrast, CARE is capable of extracting contexts from any kind of review, provided there is at least one candidate context word embedded in that review. This makes our CARE more robust toward review quality than RC-Topic and RC-Word. We analyze the impact of review quality on our CARE further in Section V-B3.

We now analyze the results from CARL, which is a context-aware method based on a deep-learning technique. First, with the advantage of utilizing both deep learning and an attention mechanism, CARL outperformed both RC-Topic and RC-Word on almost every dataset. However, although it outperformed the noncontext-deep-learning techniques (DeepCoNN and NARRE) on Amazon Software, it obtained less accurate results on the other two datasets. According to our assumption, we believe this occurred for two possible reasons. First, CARL considers the contribution of every word in reviews to the rating as the influence of contexts. Some of these words, however, are irrelevant to the user preferences or item features and could degrade the quality of the representations. The second reason is that CARL computes its attention score based on words from all previous reviews, rather than focusing on those contained in a recent review. In context-aware recommendations, contexts are relevant at the time the rating is created and, therefore, only applicable to a particular review and not to others. By considering words from all previous reviews, CARL incorporates irrelevant contexts that are not associated with the current rating situation, thereby constructing less effective representations for users and items. On the other hand, CARE constructs the representations based only on those words (and their neighbors) in a particular review that influence the rating distribution as a context and achieved better results on all datasets. This supports our assumption that considering only the words in a single review that are relevant to the user preferences and item features is better for capturing the contextual information and results in constructing more effective and meaningful representations.

3) STATIC AND DYNAMIC REPRESENTATIONS

Finally, we analyze the effectiveness of constructing user and item representations dynamically, rather than relying on a static set of representations.

First, we consider the predictive performance of the methods that learn static user and item representations (DeepCoNN, NARRE, and CARL). Although the attention-based baseline systems (NARRE and CARL) outperform DeepCoNN on the Amazon Software dataset, they deliver lower overall NDCG, HR, and MRR values on the other two datasets. This implies that applying an attention mechanism does not always improve the prediction accuracy if it is not properly integrated into the model. One concern about the attention mechanisms of both NARRE and CARL is that their attention scores are computed from the contents of previous user or item reviews, rather than utilizing the content of the target review (the review for which we want to predict a rating). The representations of NARRE, CARL, and DeepCoNN consider a user’s past preferences or item’s past features but barely capture more relevant information such as contexts, which can be extracted only from the target review.

In contrast, our dynamic approach for constructing user and item representations achieves a higher overall prediction accuracy than the static-representation approach. This focuses on utilizing the text of the target review as the main source from which to extract the relevant contexts for a user–item pair. We first apply our interaction module to model the relevance of each extracted context in the review to the individual user preferences and item features. We then compute the attention score for each context, based on its relevance level when compared with the other contexts embedded in the same review. This helps to construct fine-grained user and item representations that dynamically capture the relevance of contexts in a particular review to the user preferences and item features.

In the next section, we analyze further the performance of CARE from various angles and in various situations.

B. MODEL ANALYSIS

In this subsection, we first study the impact of the attention and the interaction modules in CARE. Next, we analyze the performance of our model for various parameter settings. We then consider the impact of the review quality on the performance of context-aware methods. Finally, we study the robustness of CARE in situations of rating sparsity.
All evaluations in this section were conducted on the TripAdvisor dataset.

1) ATTENTION AND INTERACTION MODULES

To evaluate the effect of incorporating attention and interaction modules, we created four variants of CARE based on the four possible combinations of these two modules, as listed below.

- **CARE-AttnIntr**: a model that incorporates both attention and interaction modules, that is, our main model.
- **CARE-Intr**: a model that considers only the interaction module, with the attention module being ignored. The user and item representations are constructed only from their previous interactions with contextual regions in a review. Specifically, instead of computing an attention score and using it as a weight for each region embedding in (9), we directly average all projected region embeddings obtained from (6) to form the user and item representations.
- **CARE-Attn**: a model that considers only the attention module, with the interaction module being ignored. This model aims to find the contribution of each contextual region to a review’s rating, when compared with the other regions in the same review. Technically, we use the region embeddings generated from context extraction directly as an input for (7), without applying the projection operation of (6).
- **CARE-Base**: a model without attention or interaction modules. The user and item representations are obtained by directly averaging all input region embeddings for a review. In the fully connected layer, we use one shared weight matrix for all users, and another for all items.

The hyperparameters of each variant are tuned with the same settings as the main model, as described in Section IV-C2. Their predictive performances are presented in Fig. 11.

**FIGURE 11.** The NDCG values for variants of CARE.

From Fig. 11, the CARE-Base model (no attention or interaction module) gave the lowest overall NDCG values of all the variants. The implication is that applying our attention and/or interaction modules improves the performances of our proposed model.

By modeling the varying influences of contexts in a review to the rating by applying the standard attention mechanism, the CARE-Attn model made more accurate predictions than the base model. This demonstrates that different contexts can have different impacts on the rating behaviors of all users and items. However, because CARE-Attn does not consider the relevance of each context to an individual user preferences or item features, two reviews containing the same set of contexts would produce the same user and item representations. That is, the representations generated by this model depend only on the different contributions of contexts embedded in the review, regardless of any personalized interaction with the user or item. This makes CARE-Attn suitable for a sparse dataset where most users participate only rarely or most items are rated only infrequently. However, if a review contains too few or too many candidate context words, CARE-Attn might not be able to exploit the information effectively, thereby degrading the recommendation quality. There is more analysis of the impact of the number of candidates in Section V-B2.

Although applying an attention mechanism gives improved accuracy compared with the base method, CARE-Attn still did not match the performance of CARE-Intr, which models the relevance of contexts to an individual user preferences and item features. The representations for this model are unique for each user or item even for reviews containing the same set of contexts, which is more appropriate when making a personalized recommendation. However, because CARE-Intr relies on previous interactions with contexts, it requires a significant number of reviews to precisely capture the relevance of each context. Moreover, unlike CARE-Attn, this model does not consider the varying influence of different contexts on a review’s rating, which should be a factor in improving the recommendation quality.

So far, we have explained that both CARE-Attn and CARE-Intr have their own advantages, which contribute to improved predictive performances. By recognizing the trade-off between these two methods, we found that a combined model, CARE-AttnIntr, achieved the best predictive performance among all the variants. This is convincing evidence that the influence of each context can be adequately modeled based not only on its relevance to the target user preferences and item features but also to its contribution to the rating of a particular review when compared with other contexts in the same review. This model, however, also inherits the characteristics of both CARE-Attn and CARE-Intr, in that it requires an appropriate number of candidate context words per review and enough reviews per user and per item to be able to learn high-quality user and item representations.

2) PARAMETER SENSITIVITY

We now study the impact of our model parameters on the performance of CARE. These parameters include the number of candidate context words, the region size, and the embedding size.

Figure 12 shows the impact of the number of candidate context words per review on the prediction accuracy. Because different reviews contain different numbers of candidate context words, we should address how these numbers affect the
We now investigate the impact of differently sized text regions. Increasing the region size means having more neighboring words to be identified and incorporated as contexts together with the candidate context words. Figure 13 shows the predictive performance and the validation loss in context extraction for each region size in {1, 3, 5, 7}. (Region size = 1 means that a context is constructed only from the candidate context word itself, without any consideration of its neighboring words.) Despite having a very high validation loss at the beginning, using this region size completed its training with the lowest loss when compared with other region sizes. This is because it only learns one word embedding to represent one rating distribution for each candidate, which has minimal complexity. However, its prediction accuracy is also the lowest for the various region sizes. This implies that considering only a single word as context is insufficient to accurately capture the actual influence of relevant contexts on the distributions of ratings. Because increasing the region size increases the possible number of combinations between neighboring words and the candidates to be constructed as contexts, it significantly increases the number of corresponding rating distributions, raises the model complexity, and increases the validation loss. The prediction accuracy, however, improves significantly as the region size increases from 1 to 5. This supports our assumption that incorporating neighboring words benefits the construction of more relevant contexts for capturing rating distributions, resulting in more accurate recommendations. However, an excessively large region size may also degrade the performance, as shown by the NDCG@7 value for region size = 7. This is because the model might mistakenly incorporate words from different phrases or sentences to construct incorrect contexts. For example, suppose that we extracted a context from the text region “is really worst services, the only good”. If we set the region size = 3 or 5, the context could be extracted as “worst services”, but if we set region = 7, the context might be constructed as “services good.” Exploiting such unintended
contexts could consequently affect the rating prediction score. We therefore selected the region size $= 5$ for our parameter setting because of its optimal performance.

Finally, we analyze the impact of different embedding sizes, which involves the dimensions of our word embeddings, the local context units, and the corresponding region embeddings. Figure 14 gives the predictive performance and the validation loss of our model when trained with different embedding sizes chosen from $\{50, 100, 150, 300, 450\}$. The loss values in Fig. 14 (b) show that the larger embedding sizes are more effective for representing the rating distributions, although the improvement is small between 300 and 450. Accordingly, the larger embedding sizes also produce more accurate recommendations. A size of 300 was chosen for our model setting because it gives a near-optimal overall prediction accuracy and requires less computation time than using a size of 450.

3) IMPACT OF REVIEW QUALITY
We now consider how the quality of reviews affects our context extraction and rating prediction. First, we need a method for dividing reviews with respect to their quality. We follow Bauman and Tuzhilin [20] in classifying reviews into context-rich and context-free reviews, based on the richness of contexts. The criteria used for classification includes the review features such as a number of words, number of verbs and number of verbs in past tense. After reviews are classified, we apply our context extraction method to extract the candidate context words and their associated contextual regions from each type of reviews. Table 6 shows a comparison of the statistics for each type of review on all three datasets. These statistics include the average number of words and the average number of extracted contexts (candidate context words) per review. As shown in Table 6, the number of words in context-rich reviews is significantly higher than in context-free reviews. Applying our context extraction method consequently resulted in more candidate context words being extracted from each context-rich review for all datasets. This indicates that the quality of reviews significantly impacts the amount of contexts being extracted by our extraction method.

In addition, we further analyze how this difference in number of contexts would affect the prediction capability of CARE. To do so, we further evaluated the predictive performance of CARE on each type of reviews. We divided the reviews in test data of all three datasets into context-rich and context-free reviews, and evaluated the performance for each set of reviews separately.

Figure 15 shows the comparison of NDCG@5 of CARE on each type of reviews from three datasets. From the figure, the prediction accuracies of our CARE on context-rich reviews were significantly higher than those in context-free reviews. The differences in accuracies were even more significant on Amazon Software dataset, in which the number of extracted contexts from two types of reviews were significantly different. We can then infer from the result that the review quality also have significant impact on our predictive performance. With high quality reviews, we can extract more contexts, and having more information to model the rating—resulted in more accurate prediction.

4) PERFORMANCE ON SPARSE DATA
Finally, we analyze the performance of CARE under conditions of rating sparsity. We modified the training data from the TripAdvisor dataset for two types of rating sparsity,
namely user-rating sparsity (each user provided only one rating) and item-rating sparsity (each item is rated only once). The sparsity ratios for the user-rating sparsity data and the item-rating sparsity data were 0.99937 and 0.99974, respectively. The NDCG@5 values achieved by CARE and all the baseline systems when using the sparsity-modified data are given in Fig. 16.

From Fig. 16, all methods that utilize review data for constructing user and item representations (DeepCoNN, NARRE, CARL, and CARE) produced more accurate predictions than those constructing such representations using only rating data (PMF and NMF). This implies that, in rating-sparsity situations, review content can be used to construct more effective user and item representations than utilizing only the ratings, leading to more accurate rating prediction.

However, utilizing review data does not always solve the sparsity problem. Figure 16 shows that both RC-Topic (in particular) and RC-Word produced significantly lower prediction accuracies than the other review-based methods and even lower prediction accuracies than the rating-based methods (PMF and NMF). This may be because both RC-Topic and RC-Word depend heavily on word frequency when building their review representations. Although the review data might be sparse, its vocabulary size could still be very large and have very low word frequencies. These two methods, therefore, do not have sufficient data to effectively model their topic or word distributions from which to build high-quality review representations. Utilizing these poor representations will then result in low-accuracy rating predictions.

Finally, we discuss the predictive performance of CARE, compared with the other deep-learning-based representations (DeepCoNN, NARRE, and CARL). Figure 16 shows that CARE is able to achieve the highest NDCG@5 values for both the user-rating and item-rating sparsity datasets. Note that in DeepCoNN, NARRE, and CARL, these representations are generated statically from corresponding previous reviews. Such representations, however, tend to overfit with sparse data, for which each user or item provides only one or very few reviews. Utilizing these representations to predict ratings for reviews of unseen items will then be less effective. In contrast, CARE utilizes previous reviews only to model the interactions of users and items with contexts, whereas the representations themselves are generated dynamically based on any context being extracted from each particular review. This makes CARE’s representations less affected by the sparsity in previous reviews, thereby achieving more accurate predictions.

C. CONTEXT ANALYSIS

In this subsection, we aim to show that by defining contexts in reviews as words that influence the distribution of ratings, our proposed method has the flexibility to extract contexts from review data across a variety of recommendation domains. To achieve this, we analyzed the list of candidate context words extracted from multiple review datasets across different domains. In addition to TripAdvisor, Amazon Software, and Amazon Movies & TV datasets, we also incorporated datasets from Yelp, which contains hotel and restaurant reviews, and another four of Amazon Product’s different categories, namely Fashion, Grocery & Gourmet Food, Toys & Games, and Digital Music. Examples of candidate context words extracted for each dataset are given in Tables 7 and 8.

We first analyze the list of candidate context words extracted from TripAdvisor and Yelp, which involve similar recommendation domains. As highlighted in Table 7, our extraction method was able to discover mutual words across these two datasets, such as “area”, “clean” and “friendly”, which indicate the user preferences toward the hotel features. This demonstrates that our proposed extraction method would have the flexibility to extract contexts from many kinds of review-rating datasets, independent of the dataset’s domain. Moreover, in addition to these exclusive words from particular domains, our method was able to extract sentiment

TABLE 7. Examples of candidate context words extracted from the TripAdvisor and Yelp datasets.

| TripAdvisor      | Yelp     |
|------------------|----------|
| area             | helpful  |
| bathroom         | manager  |
| breakfast        | business |
| central          | clean    |
| comfortable      | delicious|
| convenient       | dinner   |
| dirty            | family   |
| downtown         | service  |
|                 | friendly |
|                 | staff    |
|                 | happy    |
|                 | tasty    |

4https://www.yelp.com/dataset
and polarity words such as “great”, or “not” from across all the domains. This supports our assumption that these words also influence the distribution of ratings and should therefore accompany the context words used in making rating predictions.

**D. EMBEDDING ANALYSIS**

In the previous subsection, we showed that our extraction method is able to extract words representing contexts from multiple recommendation domains. In this subsection, we aim to show that the region embeddings, which are generated from context words and their neighboring words, accurately capture the rating distributions of their corresponding contextual regions and are therefore useful for rating prediction. Our assumption is that contextual regions that contribute similar rating distributions should generate region embeddings that are close to each other in the embedding space.

To investigate this assumption, we first define a method for categorizing the distributions of ratings into classes. Our approach assigns a class to each rating distribution based on its direction (positive or negative). For example, the frequencies of ratings in the distribution $dist(c_n,d_1) = [8, 25, 34, 56, 95]$ are positively distributed toward high rating scores, whereas those of $dist(c_n,d_2) = [103, 75, 41, 18, 3]$ are negatively distributed toward low rating scores. We would then categorize $dist(c_n,d_1)$ as belonging to a positive class, whereas $dist(c_n,d_2)$ should belong to a negative class. To implement this categorization, we chose the Pearson correlation coefficient to compute a correlation score between the rating distribution and an ordinal rating vector, as expressed by

$$
\rho_{dist(c_n,d_m),score_R} = \frac{\text{cov}(dist(c_n,d_m),score_R)}{\sigma_{dist(c_n,d_m)}\sigma_{score_R}} ,
$$

where cov denotes the covariance function, $\sigma$ is a standard deviation, and $score_R \in \mathbb{Z}^{|\text{Rating}|}$ is an ordinal rating score vector (sorted in ascending order) for which $|score_R| = |dist(c_n,d_m)|$. For example, $score_R = [1, 2, 3, 4, 5]$ could be used for rating data via a five-point rating score.

After computing the correlation score for each $dist(c_n,d_m)$ using $score_R$, we can then assign it to a class by using the categorization criteria given in Table 9, where we categorize the rating distributions into five classes: Strong Positive, Positive, Neutral, Negative and Strong Negative.

To visualize the subtle differences between the region embeddings, we sampled the contextual regions from the Trip Advisor and Amazon Movies & TV datasets, categorizing them based on their corresponding rating distributions and generating their region embeddings. Figures 17 and 18 were obtained by applying t-distributed stochastic neighbor embedding (t-SNE) [36] to the sampled region embeddings from each dataset, where the color of each point denotes the class of its associated rating distribution. In Fig. 17, for each dataset, we sampled 50 contextual regions from each class (250 in total) and plotted their corresponding region embeddings. Note that the group of region embeddings representing positive and negative classes is fairly distinguishable. This supports our assumption that contextual regions with similar rating distributions are mapped close to each other in the embedding space.

We can analyze the region embeddings in more detail by visualizing those that are associated with the contextual regions of each candidate context word. As shown in Fig. 18, we selected two candidate context words, “location” from Trip Advisor and “acting” from Amazon Movies & TV; we sampled 10 contextual regions that contained them from each class (50 in total) and plotted their corresponding region embeddings. Note that words that contribute positive distributions such as “great”, “good”, or “excellent” are grouped close to each other and are visually separated from negatively distributed words such as “not”, “bad”, or “but”. This again supports our assumption that neighboring words in the same text region as a candidate context word influence the distribution of ratings, and should be considered when extracting contextual information from reviews.

**E. CONTEXTS AS REGIONS**

In the previous subsection, we demonstrated that the region embedding representations of contextual regions were effective in capturing their associated rating distributions. In this

---

**TABLE 8. Examples of candidate context words extracted from six categories within the Amazon product dataset.**

| Fashion | Grocery & Food | Software | Toys & Games | Digital Music | Movies & TV |
|---------|----------------|----------|--------------|---------------|-------------|
| comely  | awful          | buggy    | birthday     | album         | acting      |
| dancing | best           | computer | broken       | beautiful     | action      |
| fit     | coffee         | crashes  | christmas    | catchy        | boring      |
| lightweight | delicious  | digital  | cute         | classic       | cast        |
| looking | disappointed  | features | daughter     | collection    | character |
| training | flavor       | installation | fun         | download      | dvd         |
| tightly | fresh         | interface | game         | instrument    | enjoy       |

| great   | good          | very      | nice        | love         | not         | but        | however    | even       | all        |

**TABLE 9. Criteria for categorizing a rating distribution based on correlation score.**

| Correlation | Class      |
|-------------|------------|
| $\rho \geq 0.9$ | Strong Positive |
| $0.4 \leq \rho < 0.9$ | Positive |
| $-0.4 < \rho < 0.4$ | Neutral |
| $-0.9 \leq \rho < 0.4$ | Negative |
| $\rho \leq -0.9$ | Strong Negative |
subsection, we further analyze the merits of defining and extracting a context as a contextual region. First, we show that the rating distributions captured by contextual regions explain the polarity of a review more effectively than those captured by single words. We investigate this assumption by utilizing the embeddings of single candidate context words and the corresponding contextual regions for review sentiment classification.

1) REVIEW POLARITY
Figure 13 (a) in Section V-B2 showed that utilizing a contextual region of size = 1 (i.e., considering only candidate context words not influenced by neighboring words) produced the least accurate rating prediction among those for a range of region sizes. This indicated that defining a context as a single word is less effective in modeling a review’s rating than defining it as a text region. To support this finding, we aim here to visualize the difference in influence on a review’s polarity between considering a context as a single word and considering it as a text region. To demonstrate this, we applied our context extraction method to an example review from the TripAdvisor dataset for the two types of context, as shown in Fig. 19. Figure 19 (a) shows the contexts extracted as single words, whereas Fig. 19 (b) shows them extracted as contextual regions. The highlight colors reflect the class of the associated rating distribution for each context, as defined in Section V-D.

Consider first the contexts extracted as single words in Fig. 19 (a). Note that many extracted words in this review such as “not”, “disappointed”, “crowd”, or “noise” were classified as contexts with negative rating distributions. Utilizing those words individually could negatively affect the polarity of the review, and consequently result in a low rating prediction score, which does not accord with the actual rating score. This implies that extracting contexts as single words might not adequately explain the polarity of a review. This can happen because some single words fail to capture the actual rating distribution patterns of contexts. In fact, their combination with some neighboring words could radically alter their rating distribution pattern and lead to a totally different interpretation for a rating prediction score. For example, “not disappointed” is classified as a context with a positive rating distribution even though both “not” and “disappointed” are associated with negative rating distributions.
By extracting contexts as contextual regions, we are able to assign a more appropriate rating distribution class to each context. As illustrated in Fig. 19 (b), some contextual regions such as “not disappointed”, “close enough for an easy walk”, and “but far from crowd and noise” are no longer assigned with negative rating distributions. Many positive rating distributions could positively affect the polarity of this review and produce a high rating score that is closer to the actual rating score. From this example, we can hypothesize that extracting contexts as contextual regions is more effective in explaining the polarity of a review than extracting them as single words. We can investigate this hypothesis by evaluating the comparative performance using single context words against that using contextual regions for the review sentiment classification task.

2) REVIEW SENTIMENT CLASSIFICATION

Here, we aim to demonstrate that, in addition to their use in making rating predictions, the use of contextual regions is more effective in modeling the polarity of a review than the use of single context words. This demonstration involves the sentiment classification of reviews.

For comparison, we set up two models for review classification, namely a “single” and a “region” model. For the single model, we utilized the word embeddings of all candidate context words learned by our context extraction method for region size = 1. We then averaged the word embeddings of all candidate context words in each review to create an embedding representation for that review. For the region model, we computed the region embeddings for all corresponding contextual regions, using word embeddings and local context units learned by our context extraction method for region size = 5. We then created a representation of each review by averaging all region embeddings of all contextual regions in that review.

To evaluate the classification accuracy of these two models, we chose logistic regression as a binary classifier of the review representations and thereby producing more accurate rating predictions.

| Dataset                      | Context | Classification Accuracy | F1 Score  |
|------------------------------|---------|-------------------------|-----------|
| Amazon Software              | Single  | 0.71861                 | 0.82371   |
|                              | Region  | 0.74561                 | 0.83584   |
| TripAdvisor                  | Single  | 0.79399                 | 0.83615   |
|                              | Region  | 0.83858                 | 0.89189   |
| Amazon Movies & TV           | Single  | 0.81664                 | 0.88300   |
|                              | Region  | 0.84097                 | 0.89842   |

VI. CONCLUSION

We have proposed a novel unsupervised method for extracting contexts from reviews, together with a predictive model that utilizes the extracted contexts for rating prediction. Unlike previous context-aware rating mechanisms, a context in our work can be automatically extracted not only in single word format but also in combination with those neighboring words from the same text region that influence the distributions of ratings. This makes our approach applicable to a wide variety of recommendation domains and suitable for extracting contexts from sets of reviews that may involve a variety of styles. In making rating predictions, our user and item representations are generated dynamically for each specific review through our proposed interaction and attention modules, based on the relevance of the contexts extracted from that review to a user preferences and an item features. Utilizing our representations for making rating predictions is more accurate than using state-of-the-art deep-learning-based representation techniques that do not properly consider the relevance of a context. This point is strongly supported by our experimental results, in which our approach exhibited the best or near-best performance among many state-of-the-art competitors.

In future work, we aim to develop a unified model that combines context extraction and rating prediction into a single step, which we believe will be more efficient for learning the user and item representations and thereby producing more accurate rating predictions.

ACKNOWLEDGMENT

This work was supported in part by the National High Technology Research and Development Program of China (Grant No. 2019YFF1400100), in part by the National Key R&D Program of China (Grant No. 2018YFB1307603), and in part by the National Natural Science Foundation of China (Grant No. 61772470).
[10] L. Baltrunas, B. Ludwig, and F. Ricci, “Matrix factorization techniques for context aware recommendation,” in Proc. 5th ACM Conf. Recommender Syst. (RecSys), 2011, pp. 301–304.

[11] P. Sitkrongwong, S. Maneeroj, P. Samathiyadikan, and A. Takasu, “Bayesian probabilistic model for context-aware recommendations,” in Proc. 17th Int. Conf. Inf. Integ. Web Appl. Services (iWAS), 2015, pp. 1–10.

[12] Y. Li, J. Nie, Y. Zhang, B. Wang, B. Yan, and F. Weng, “Contextual recommendation based on text mining,” Proc. 23rd Int. Conf. Comput. Linguistics. Posters (COLING), 2010, pp. 692–700.

[13] N. Hariri, M. Mobasher, B. Burke, and Y. Zheng, “Context-aware recommendation based on review mining,” in Proc. 9th Int. Workshop Intell. Techn. Web Personalization Recommender Syst. (ITWP), 2011, pp. 30–36.

[14] G. Chen and L. Chen, “Recommendation based on contextual opinions,” in User Modeling, Adaptation, and Personalization. Cham, Switzerland: Springer, 2014, pp. 61–73.

[15] F. Z. Lahlou, H. Benbrahim, A. Mountassir, and I. Kassou, “Inferring context from users’ reviews for context aware recommendation,” in Research and Development in Intelligent Systems XXX. Cham, Switzerland: Springer, 2013, pp. 227–239.

[16] S. Aciar, “Mining context information from consumers reviews,” in Proc. Workshop Context-Aware Recommender Syst., vol. 2010.

[17] F. Z. Lahlou, A. Mountassir, H. Benbrahim, and I. Kassou, “A text classification based method for context extraction from online reviews,” in Proc. 8th Int. Conf. Intell. Syst. Appl. (SITA), Rabat, May 2013, pp. 1–5.

[18] L. Liu, F. LeCun, N. Meandjiev, and L. Xu, “Using context similarity for service recommendation,” in Proc. IEEE 4th Int. Conf. Semantic Comput., Pittsburgh, PA, USA, Sep. 2010, pp. 277–284.

[19] A. Odíć, M. Tkalič, J. F. Tasič, and A. Košir, “Predicting and detecting the relevant contextual information in a movie-recommender system,” Interacting Comput., vol. 25, no. 1, pp. 74–90, Jan. 2013.

[20] K. Bauman and A. Tuzhilin, “Discovering contextual information from user reviews for recommendation purposes,” in Proc. CEUR Workshop, vol. 1245, 2014, pp. 2–8.

[21] F. J. Peña, “Unsupervised context-driven recommendations based on user reviews,” in Proc. 11th ACM Conf. Recommender Syst., Aug. 2017, pp. 426–430.

[22] L. Wu, C. Qian, C. Li, Q. Wang, B. Zheng, and X. Luo, “A context-aware user-item representation learning for item recommendation,” ACM Trans. Inf. Syst., vol. 37, no. 2, pp. 1–29, Mar. 2019.

[23] P. Sitkrongwong and A. Takasu, “Unsupervised context extraction via region embedding for context-aware recommendations,” in Proc. 23rd Int. Database Appl. Eng. Symp. (IDEAS), 2019, pp. 1–10.

[24] C. Qiao, B. Huang, G. Niu, D. Li, D. Dong, W. He, D. Yu, and H. Wu, “A new method of region embedding for text classification,” in Proc. 6th Int. Conf. Learn. Representations (ICLR), Vancouver, BC, Canada, 2018, pp. 1–12.

[25] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[26] R. Sablakhtdinov and A. Mnih, “Probabilistic matrix factorization,” in Proc. 20th Int. Conf. Neural Inf. Process. Syst. (NIPS), 2007, pp. 1257–1264.

[27] D. D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in Proc. 13th Int. Conf. Neural Inf. Process. Syst. (NIPS), 2000, pp. 535–541.

[28] Y. Kim, “Convolutional neural networks for sentence classification,” in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), 2014, pp. 1746–1751.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS), 2017, pp. 6000–6010.

[30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Proc. 26th Int. Conf. Neural Inf. Process. Syst., vol. 2, 2013, pp. 3111–3119.

[31] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), 2014, pp. 1532–1543.

[32] J. Devlin, M. Chang, K. Lee, and T. Kristina, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in Proc. NAACL-HLT, 2019, pp. 4171–4186.

[33] K. Diederik and J. Ba, “Adam: A method for stochastic optimization,” in Proc. 3rd Int. Conf. Learn. Representations (ICLR), San Diego, CA, USA, 2015, pp. 1–15.

[34] J. Ni, J. Li, and J. McAuley, “Justifying recommendations using distantly-labeled reviews and fine-grained aspects,” in Proc. Conf. Empirical Methods Natural Lang. Process., 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), Hong Kong, 2019, pp. 188–197.

[35] S. Rendle, “Factorization machines with libFM,” ACM Trans. Intel. Syst. Technol., vol. 3, no. 3, pp. 1–22, May 2012.

[36] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” J. Mach. Learn. Res., vol. 9, pp. 2579–2605, Nov. 2008.

PADIPAT SITKRONGWONG received the B.S. and M.S. degrees from Chulalongkorn University, Thailand, in 2013 and 2015, respectively. He is currently pursuing the degree SOKENDAI (The Graduate University for Advanced Studies), Japan. His research interests include recommender systems and machine learning.

ATSUHIRO TAKASU (Member, IEEE) received the B.E., M.E., and Dr.Eng. degrees from The University of Tokyo, Japan, in 1984, 1986, and 1989, respectively. He is currently a Professor with the National Institute of Informatics, Japan. His research interests include data engineering and data mining. He is also a member of ACM, IEICE, IPSJ, and JSAI.

SARANYA MANEEROJ received the B.S. degree from Chulalongkorn University, Thailand, in 1996, and the M.E. and Dr.Eng. degrees from The University of Electro-Communications, Japan, in 2001 and 2005, respectively. She is currently an Associate Professor with the Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University. Her research interests include recommender systems and data mining.