A Review Semantics Based Model for Rating Prediction

RENHUA CAO\textsuperscript{a}, XINGMING ZHANG\textsuperscript{a}, AND HAOXIANG WANG\textsuperscript{b}

School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding author: Xingming Zhang (cszxm@scut.edu.cn)

This work was supported in part by the Guangdong Provincial Science and Technology Plan of China under Grant 2017B010111003 and the Fundamental Research Funds for the Central Universities of China under Grant D2190680.

ABSTRACT A review expresses the concerned aspects and corresponding assessments a customer has towards a particular item. Extracting the user’s interests and product’s features from their aggregated reviews and matching them together to predict the overall rating is a common paradigm in a review-based recommender. However, such a paradigm trains model on the aggregated historical review which grows with time and may have much conflicting semantics, thus the scalability and accuracy may be compromised. In this paper, a novel review semantics based model (RSBM) is proposed to enhance the performance of the review-based recommender. It consists of three parts: the review semantics extractor, the review semantics generator and the rating regressor. Firstly, the review semantics extractor uses a convolutional neural network (CNN) to extract the semantic features of a particular review text. Secondly, the semantics generator uses a memory-network liked structure and attention mechanism to simulate the decision-making process which assesses each concerned aspect of an item to generate the review semantics. In the training phase, the generated semantics is compared with the semantics extracted by the review semantics extractor. In the last, the generated semantic features are fed into the rating regressor to predict the overall rating. Experiments on a series of reality datasets show that the proposed model gains better performances than several state-of-the-art recommendation approaches in terms of accuracy and scalability.

INDEX TERMS Recommender system, deep learning.

I. INTRODUCTION

Modern e-commerce systems typically have billions of items for sale, making it difficult for users to pick out the items they need. The recommender system recovers from such a dilemma, which not only improves the user’s shopping experience but also increases platform revenue. To achieve this goal, the core task falls on how to accurately model the user’s preferences.

Collaborative Filtering (CF) is the most popular state-of-the-art recommendation methodology, which models the user’s preferences from their historical actions such as clicking, rating and viewing. However, such a methodology only models the action itself without understanding the motivation behind the action. Thus its accuracy and interpretability may be compromised. What’s more, it will meet a dilemma when the historical action records are sparse.

The associate editor coordinating the review of this manuscript and approving it for publication was Long Wang\textsuperscript{c}.

FIGURE 1. A sample user review on Amazon.

To address the limitations of collaborative filtering, much auxiliary information is used. The review text is one of them. Almost all e-commerce systems provide the review module. In each review action, the user not only expresses his concerned aspects and corresponding assessments in a review text but also provides an overall rating. For example, in figure 1, it shows a review record given by a user...
towards a cell phone on Amazon. With the flexibility of the natural language, the user is satisfied with the screen, design and camera of the mobile phone but complains about the battery life, thus the overall rating is 4. It is usual for the user to measure different aspects of the target item when making a purchase decision. Modeling such a phenomenon, the recommender system can capture the user’s fine-grained preferences to make more accurate recommendations.

There are a lot of approaches to utilize reviews [19], [31]. They usually follow the same paradigm, in which the user’s and the item’s historical reviews are aggregated into a long document. The topic model or deep neural network is used to learn the user’s and item’s representations, then these features are either incorporated with matrix factorization(MF) model or directly fed into a regression model to estimate the overall rating. Although they have achieved promising results, few of them explicitly model the fact that different users may have different assessments on different concerned aspects. What’s more, the length of aggregated reviews grows with time, which limits the scalability of such a paradigm.

This paper proposes a novel review semantics based model(RSBM), which uses a memory-network liked structure to explicitly model the user’s assessment action on each aspect. Then the overall assessment features are drawn to approximate the semantic features of the review text for more accurate rating prediction. What’s more, the proposed model does not rely on the review aggregation operation, thus it has a better training efficiency and scalability compared to previous models. The contributions of this paper are as follows:

- A novel review semantics based model(RSBM) is proposed, which explicitly models the user’s assessment action on each aspect and generates an overall assessment features based on that to approximate the semantic features extracted from the review for an accurate rating prediction.
- The model uses a different paradigm, in which the model can be trained on a single review text, thus it has a better training efficiency and scalability.
- Experiments on a series of datasets show that the proposed method has a better performance than several state-of-the-art baselines in terms of rating prediction.

The remaining of this paper is organized as follows: in Section 2, we review collaborative filtering and review-based recommendation methods related to the proposed model. In Section 3, we define the problem in a way different from the previous common paradigm. In Section 4, we describe the overall framework of the proposed model and implementation details for each part. In Section 5, we theoretically analyze the complexity of the model. In Section 6, a series of experiments are used to demonstrate the performance of the model. A brief conclusion and future work are given in Section 7.

II. RELATED WORK

Collaborative filtering has proven to be one of the most effective methods in the recommender system. The latent factor model(LFM) is one of the most influential forms in collaborative filtering. LFM [1] represents the user and the item as latent factors in a shared latent space and uses the inner product of latent factors to predict the rating. A lot of works concentrate on enhancing the performance of LFM. PMF [21] assumes that the latent factors of users and items obey the Gaussian distribution, and the rating is estimated by another Gaussian distribution with the mean been set to the inner product of the latent factors. BMF [17] adds user’s bias, item’s bias and global bias to enhance the performance of the original matrix factorization. SVD++ [12] incorporates neighborhood-based information into LFM. There is a trend of exploiting deep learning in collaborative filtering. Many deep models such as Restricted Boltzmann machine(RBM) [22], autoencoder [23] and variational autoencoder(VAE) [18] have been applied to raw ratings. NCF [11] proposes NeuMF, in which a deep neural network is used to model the nonlinear relationship between user’s and item’s latent factors. LRML [26] uses the memory module and attention mechanism to make the model more flexible by directly modeling the latent relations instead of the latent factors. CMN [9] applies memory neural network to collaborative filtering to model the high order neighborhood-based relations between items. Although collaborative filtering has achieved remarkable results, it suffers from data sparsity and poor interpretability. Much auxiliary information such as images, texts, social relations is used to recover from such a dilemma [13], [27].

Reviews and ratings, as the most common user-generated information in the e-commerce system, are often combined to make more accurate and interpretable recommendations. The review-based recommendation methods can be generally categorized into sentiment-based, topic-based and deep learning based [7], [10].

Sentiment-based works first analyze the sentiment of reviews and use the sentiment information to enhance rating prediction. For example, [20] maps sentiments into ratings and combine them with neighborhood-based models for one-class collaborative filtering over TED talks. [30] uses phrase-level sentiment analysis to extract the product’s features and user’s opinions, then combines them with matrix factorization model which considers explicit features and hidden factors together. These methods rely on sentiment analysis tools and sentiment dictionaries to work normally, thus the performance may not be very stable.

Topic-based models apply Latent Dirichlet Allocation(LDA) [3] to reviews to capture the topic distributions. HFT [19] builds a link between the topic factors and the latent factors in matrix factorization with a transformation. Unlike HFT, the relation between topics and latent factors only exists for items or users, TopicMF [2] correlates topic factors with corresponding latent factors for both users and items. JMARS [8] proposes an entirely unsupervised probabilistic model to capture the interest distribution of users and the content distribution of movies. It models the fact that the user has different sentiments on a per-aspect basis.
The historical data can be denoted as a collection of quads \((v_u^d, v_i^d, d_{u,i}, r_{u,i})\), where \(v_u^d\) represents the identity of user \(u\), \(v_i^d\) represents the identity of item \(i\), \(d_{u,i}\) denotes the review text given by the user \(u\) to the item \(i\) and \(r_{u,i}\) is the corresponding rating. Hence, the representations of user \(u\) and item \(i\) can be formulated as
\[
\begin{align*}
p_u^d &= f_u(D_u|\theta_u), \\
q_i^d &= f_i(D_i|\theta_i),
\end{align*}
\]
where \(p_u^d\) is the representation of user \(u\) learned from the aggregated review and \(q_i^d\) is the representation of item \(i\). \(f_u(.|\theta_u)\) and \(f_i(.|\theta_i)\) are the representation learning model of users and items. Then these representations are combined with latent factors in the MF model and fed into a regression model to predict the overall rating.

### B. THE PROPOSED PARADIGM

The aggregated review grows with time, which results in non-negligible computational overload. We propose another paradigm to predict the overall rating via reviews. The overall rating is formulated as
\[
\hat{r}_{u,i} = \psi(s|\theta_r),
\]
where \(\psi(.|\theta_r)\) is the rating prediction model and \(s\) is the user’s overall assessment semantics extracted from the review. The semantics extraction process is formulated as
\[
s = \phi(d_{u,i}|\theta_g),
\]
where \(\phi(.|\theta_g)\) is the semantics extraction model and \(\theta_g\) is the corresponding model parameter. However, the purchase behavior always happens before the review action, so the user’s review is not available in the recommendation phase. To recover from such a dilemma, we define a semantics generation model to generate the overall assessment semantics of the review text. It is formulated as
\[
\hat{s} = g(v_u^d, v_i^d|\theta_g),
\]
where \(g(.|\theta_g)\) is the semantics generation model, \(\theta_g\) is the corresponding model parameter and \(\hat{s}\) is the generated semantics.

In conclusion, the task is to learn model \(\psi(.|\theta_r), \phi(.|\theta_g)\) and \(g(.|\theta_g)\) with constraints that the predicted rating \(\hat{r}_{u,i}\) is as close as possible to the ground truth rating \(r_{u,i}\) and the generated semantics \(\hat{s}\) is as approximated as possible to the extracted semantics \(s\) of the review text \(d_{u,i}\).

### IV. PROPOSED METHODOLOGY

In this section, we first introduce the overall framework and the idea of the model, then describe the implementation of each part in the model detailly.

#### A. FRAMEWORK

The idea of the model is motivated by generative adversarial networks and transfer learning. It models the fact that the user firstly assesses each aspect of the item and then draws an overall assessment based on assessments of these aspects. As the figure 2 shows, the model consists of three parts: the review semantics extractor, the review semantics generator and the rating regressor. The review semantics extractor uses CNN to extract the user’s assessment semantics of the review.
The review semantics generator uses an attention network to estimate the user’s assessment on each aspect of the item and these assessments are used as coefficients to linearly combine the embedding of each aspect to generate the overall assessment semantics. The rating regressor predicts the overall rating based on the input features. In the training phase, the rating regressor predicts the overall rating based on the semantics of the review extracted by the review semantics extractor and the generated semantics generated by the review semantics generator is required to approximate the semantics of the review extracted by the review semantics extractor. In the recommendation phase, to overcome the embarrassing situation that the review is not available, the rating regressor predicts the overall rating based on the generated semantics generated by the review semantics generator instead of the semantics of the review extracted by the review semantics extractor.

### B. REVIEW SEMANTICS EXTRACTOR

There are many ways to extract the semantic representation of text sequences, such as RNN and CNN. Similar to DeepCoNN, we use CNN to extract the semantic feature of the review because the computation of CNN is respectively faster than RNN. Unlike DeepCoNN, we use a single review text as input instead of the aggregated review, which greatly reduces the length of input and speeds up the calculation of CNN. Therefore less resource consumption is needed during training.

Give a review text \( d_{u,i} = \{w_1, w_2, \ldots, w_l\} \), which denotes the review written by the user \( u \) to the item \( i \), where \( w_k \) is the \( k \)-th word in the review and \( l \) is the length. Similar to TextCNN [14], we first process the review text through a word embedding layer. We apply word embedding \( f : w_k \rightarrow \mathbb{R}^d \) for each word in the review, then each word is represented as a dense vector. The parameters of the embedding layer can be trained by word2vec or randomly initialized. After the word embedding, a review text is represented by a matrix \( V \in \mathbb{R}^{l \times d} \), where \( d \) is the dimension of the embedded word vector. Next to the word embedding layer, a convolutional layer is used to extract the phrase-level semantics of the text through multiple filters of different sizes. In a convolutional layer, suppose there are \( m \) neurons. For each neuron, it uses \( K_j \) as a filter to perform a convolution operation in a sliding window of the document. The process is formulated as

\[
c_j = \text{relu}(V \ast K_j + b_j),
\]

In which, \( \ast \) denotes the convolutional operator. \( K_j \in \mathbb{R}^{t \times d} \) is the parameter of the filter kernel, where the kernel size is \( t \times d \). \( b_j \) is the bias and \( \text{relu}(.) \) is the activation function, which is defined as

\[
\text{relu}(x) = \max(0, x).
\]

To retain the main semantics and suppress the noise, we apply the max-pooling operation to the output of the convolutional operation immediately. The max-pooling operation is
formulated as
\[ o_i = \max([c_1, c_2, \ldots, c_{i-1} + 1]) \]  \tag{10}

To extract a variety of semantic features, multiple filters are applied. Finally, the output of the convolutional layer is
\[ o = [o_1, o_2, \ldots, o_n] \]  \tag{11}

A full-connected layer is used to transform the semantic features into the same dimension as the generated semantic features because the semantic features extracted by the convolutional layer may be inconsistent with the semantic features generated in the review semantics generator. So the final semantic features of the review are formulated as
\[ s = \text{relu}(W_s o + b_s) \]  \tag{12}

in which \( W_s \) is the weight of the full-connected layer while \( b_s \) is the bias.

C. REVIEW SEMANTICS GENERATOR

A user often assesses each aspect of the item. Purchasement is determined by the overall assessment, which is drown based on the assessment on each aspect of the item. We use a memory-network liked structure to simulate such a process. An attention neural network similar to the key address mechanism in a memory neural network is used to estimate the user's assessment on each aspect of the item, the user mate the user's assessment on each aspect to generate the overall semantic features. To estimate the user's assessment on each aspect of the item, then these assessments are regarded as coefficients to linearly combine the embedding of each aspect while the scale of \( a \) is the user's assessments. The process is formulated as
\[ s^i = \text{softmax}(W_l x^i + b) \]  \tag{14}

where \( P \in R^{U \times d_u} \) is the embedding matrix of users and \( Q \in R^{I \times d_v} \) is the embedding matrix of items. \( |U| \) denotes the number of users and \( |I| \) denotes the number of items. \( v^u_i \) is the one-hot encoding representation of the identity of user \( u \) and \( v^l_i \) is the one-hot encoding representation of the identity of item \( i \). Then these representations are concatenated as input to the attention network. It is formulated as
\[ z = [p_u, q_i] \]  \tag{15}

where \( z \) is the pair representation of the user \( u \) and item \( i \). \([::] \) is a vector concatenation operation. We use the concatenation in this case because concatenation losses less information. Other fusion methods such as dot product can also be used. Then the operation of the attention network is formulated as
\[ h_b = \text{relu}(W_h z + b_h) \]  \tag{16}

\[ a = \text{tanh}(W_o h_b + b_o) \]  \tag{17}

In which, \( a \) is the user's assessments. \( a_k \) as the k-th dimension of \( a \), the sign of \( a_k \) represents the user’s sentiment on the k-th aspect while the scale of \( a_k \) indicates the importance of that aspect. \( h_b \) is the hidden state of the attention network. \( W_h, W_o \) are the weights and \( b_h, b_o \) are the biases. It should be noted that we used \( \text{tanh}() \) as the final layer activation function instead of \( \text{softmax}() \). \( \text{tanh}() \) and \( \text{softmax}() \) are defined as
\[ \text{tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \]  \tag{18}

\[ \text{softmax}(x)_j = \frac{e^{x_j}}{\sum_i e^{x_i}} \]  \tag{19}

In the vanilla attention mechanism, \( \text{softmax}() \) is used. However, the user's assessment of each aspect can be negative or positive, which is naturally a value in \([-1, 1]\). The output of \( \text{tanh}() \) meets such a constraint and we also find \( \text{tanh}() \) has a better performance than \( \text{softmax}() \) in the experiment.

Suppose there are \( d_u \) aspects that the user is concerned about of the item, where \( d_u \) is a hyperparameter similar to the number of topics in the topic model. Similar to the word embedding, for each aspect \( e_k \), we embed it into a dense vector \( m_k \). Then all aspects can be represented by a matrix \( M \in R^{d_u \times d} \), where \( d \) is the dimension of the embedding vector of each aspect. In the previous step, the user's assessments on these aspects are got by the output of the attention network, the overall assessment features can be regarded as the linear combination of the embedding vector of each aspect with the assessment on each aspect as a coefficient. So the overall assessment features are formulated as
\[ \hat{s} = a^T M \]  \tag{20}

In the training, we hope that the generated user’s overall assessment features are similar to the semantic features expressed in the review. Thus the \( L_2 \) loss is used to push them as close as possible, which is formulated as
\[ L_s = ||s - \hat{s}||_2 \]  \tag{21}

D. RATING REGRESSOR

Through the process of the review semantics extractor, the semantic features of the review are extracted. We predict the overall rating based on the semantic features. In this case, we use a multiple layer feedforward neural network(MLP) to learn the regression function. Supposing there are \( n_r \) layers in the rating regression network. The process is formulated as
\[ x^i = \sigma(W^i x^{i-1} + b^i) \]  \tag{22}

in which \( x^i \) is the output of the i-th layer. \( W^i, b^i \) are the weight and the bias of the i-th layer. \( \sigma(.) \) is the activation function. Without special statement, we set \( \sigma(.) \) to \( \text{relu}(.) \). The predicted overall rating is formulated as
\[ \hat{r}_{u,i} = x^\omega + b_u + b_i + b_g \]  \tag{23}

where \( b_u, b_i, b_g \) are the user bias, item bias and global bias. During the training phase, we set
\[ x^0 = s \]  \tag{24}

while in the recommendation phase, the semantic features of the review are not available, we set it to the generated overall assessment features, which is formulated as
\[ x^0 = \hat{s} \]  \tag{25}
E. LEARNING

There are three parts of the loss: the semantic features approximation loss $L_s$, the rating prediction loss $L_r$ and the parameter regularization loss $L_w$. The final optimization goal of the model is formulated as

$$\theta^* = \arg \min_{\theta} \sum_{(v^j_u, v^j_l, d_{u,i}, r_{u,i}) \in B} (L_r + L_s + \lambda L_w),$$

(26)

$$L_r = (r_{u,i} - \hat{r}_{u,i})^2,$$

(27)

$$L_w = \sum_{j=1}^{m} ||W^j||_2.$$

(28)

In which, $\lambda$ is the regularization parameter to balance the parameter regularization loss in the rating regressor to prevent overfitting. With the loss defined, we use the back-propagation algorithm (BP) to train the model parameters. Detail optimization steps are shown in algorithm 1.

Algorithm 1 Learning RSBM

Input: Training set $C = \{(v^j_u, v^j_l, d_{u,i}, r_{u,i})\}$, the user’s latent factor dimension $d_u$, the user’s latent factor dimension $d_i$, word embedding dimension $d_w$, number of aspects $d_a$, aspect embedding dimension $d$, batch size $batchSize$, number of epochs $numEpochs$, regularization term $\lambda$, learning rate $\alpha$.

Output: Optimal model parameter $\theta$;

1. initialize model parameter $\theta$.
2. $numBatches = |C|/batchSize$.
3. for $n = 1$ to $numEpochs$ do
4.   for $m = 1$ to $numBatches$ do
5.     $B \leftarrow$ Sample($C$, batchSize)
6.     for all $(v^j_u, v^j_l, d_{u,i}, r_{u,i}) \in B$ do
7.       Extract the semantic features $s$ of the review $d_{u,i}$ by Eq.8-12.
8.       Generate the user’s assessment features $\hat{s}$ by Eq.13-20.
9.       Estimate the overall rating $\hat{r}_{u,i}$ by Eq.22-24.
10.  Calculate the loss $L_s, L_r, L_w$ by Eq.21,27,28.
11. end for
12.  Optimize $\theta$ by Eq.26 with learning rate $\alpha$ through BP.
13. end for
14. end for

VI. EXPERIMENTS

In this section, we first introduce the datasets and evaluation metric used in the experiment. Then we describe baselines and corresponding settings. Finally, we analyze the experimental results and some of the characteristics of the model.

A. DATASETS AND EVALUATION METRIC

In the experiment, we mainly used the datasets from Amazon and Yelp to evaluate the proposed model.

Amazon\textsuperscript{1}: The data in the dataset is collected from Amazon, contains 142.8 million product reviews spanning May 1996 to July 2014. We used the processed 5-core datasets, which guarantee each user and each item has at least 5 reviews. The product is divided into 24 categories, we only use Baby, Beauty, Cell Phone and Accessories, Digital Music, Grocery and Gourmet Food, Musical Instruments, Patio, Lawn and Garden and Toys and Games.

Yelp\textsuperscript{2}: This dataset is provided by Yelp for Dataset Challenge 2014, which contains business attributes, check-in records, user information and user reviews. We only extract user Id, business ID, stars and the corresponding review from the original review record as our experimental dataset.

Detailed statistics for each dataset are listed in table 1. Similar to other rating prediction recommender methods, we use mean square error (MSE) as the evaluation metric. It’s

| # | #users | #items | #ratings&reviews | density(%) |
|---|---|---|---|---|
| Baby | 19443 | 7050 | 166792 | 0.11 |
| Beauty | 22363 | 12101 | 198502 | 0.073 |
| Digital Music | 5541 | 3568 | 64706 | 0.327 |
| Patio, Lawn and Garden | 1686 | 962 | 13272 | 0.818 |
| Toys and Games | 19412 | 11924 | 167997 | 0.072 |
| Cell Phone and Accessories | 27879 | 10429 | 194439 | 0.067 |
| Musical Instruments | 1429 | 900 | 10261 | 0.798 |
| Grocery and Gourmet Food | 14661 | 8713 | 151254 | 0.11 |
| Yelp2014 | 43904 | 11537 | 229907 | 0.063 |

\textsuperscript{1}http://jmcauley.ucsd.edu/data/amazon/

\textsuperscript{2}https://www.yelp.com/dataset/challenge

V. THEORETICAL ANALYSIS

Assuming each review has an average length of $l$; each user has an average of $N^U$ historical reviews; each item has an average of $N^I$ historical reviews; there are total $|C|$ reviews. It is difficult to analyze the computation complexity because each model has diverse structures that depend on the setting. To make the comparison more simplified, we assume that the computation complexity of operations in each model is constant. Therefore, the overall computation complexity of the model is proportional to the size of the input. For models in the aggregated paradigm, they use the aggregated reviews to learn the representations of users and items, thus the computation complexity is $O(|C|(|N^U I + N^I I|)).$ If the model uses the topic modeling technique to learning the representations of users and items, which uses sampling methods such as Gibbs sampling to learn the parameters, the computation complexity may even get worse. For the proposed method, the model is trained on a single review, so its complexity is $O(|C|I)$ which is much lower than that in the aggregated paradigm. What’s more, when the user posts a new review, methods based on aggregated reviews need to reconstruct the input and the aggregated reviews will grow with time, thus it results in a huge additional computation. For our method, the new review can be directly applied to the original model via incremental learning, so our model is more suitable for online scenarios and has finer flexibility.
defined as
\[
MSE = \frac{1}{|\Omega|} \sum_{(u,i) \in \Omega} (\hat{r}_{u,i} - r_{u,i})^2,
\]
where \(\Omega\) is the test set and \(|\Omega|\) denotes the size of the test set. \(\hat{r}_{u,i}\) is the predicted rating while \(r_{u,i}\) is the ground truth.

\section*{B. BASELINES AND SETTINGS}
A variety of state-of-the-art methods are compared to demonstrate the superiority of the proposed method. The baselines can be mainly divided into collaborative filtering based, topic-based and deep learning based.

BMF [17]: this is the latent factor model that represents users and items as latent factors and uses Alternating Least Squares(ALS) to estimate the latent factors from historical ratings.

NeuMF [11]: this is a state-of-the-art collaborative filtering based method. It replaces the inner product operation in MF by MLP to capture the nonlinear relationship between the user’s latent factors and the item’s latent factors.

HFT [19]: this is a topic-model based method which combines the reviews and ratings. It links latent factors to topic factors by a transformation function.

DeepCoNN [31]: this is a state-of-the-art method, which utilizes the reviews to enhance the accuracy of rating prediction purely based on deep learning. Two CNN processors are used to extract the user’s profile and the item’s property. Then the Factorization Machine is employed to match the user’s profile and the item’s property together to predict the rating.

A3NCF [6]: this is a state-of-the-art method that combines the topic model and deep learning. It fuses the topic factors and latent factors. The attention mechanism is used to discriminate the importance of each factor.

For each dataset, like previous work, we first apply regular expression \(^1\) to each raw review to retain the text content, and then use the NLTK\(^2\) toolkit to perform preprocessing operations such as removing stop words. For each dataset, we randomly divide it into training, validation and test set with ratio 8:1:1. The validation set is used to determine the optimal value of the hyperparameter, once it is set, the result on the test set is reported with the optimal parameter. For BMF and NeuMF, we implement them in tensorflow, the number of latent factors is picked from \(\{8, 16, 32, 64\}\). For the topic-based model, HFT and A3NCF, we used the implementation provided by [6], [19]. The optimal number of topics is picked from \(\{5, 10, 15, 20, 25\}\) according to the validation performance similar to that in A3NCF. For the DeepCoNN, we used the implementation in [5]. It should be noted that the problem of sample leakage is avoided in our experiment.

We set the word embedding size to 100 and set the other parameters to the optimal values described in DeepCoNN. For our proposed method-RSBM, we select the number of latent factors and aspects in \(\{8, 16, 32, 64\}\). In the review semantics extractor, we set the filter size to \(\{2, 3, 5\}\) and the number of each filter is picked from \(\{16, 32, 100\}\). In the rating regressor, we use a feedforward neural network having a similar pyramid structure mentioned in [11] and the regularization parameter \(\lambda\) is tuned in \(\{0, 0.01, 0.05, 0.1, 0.5, 1\}\). In the training, we pick the optimal learning rate from \(\{0.1, 0.01, 0.001, 0.0001, 0.00001\}\) and use the Adam [15] optimizer for all methods. We select the optimal batch size from \(\{64, 128, 256\}\). For a fair scalability comparison experiment, we set the batch size to the same value for all comparison methods. All experiments are conducted on a desktop computer with a 16G memory and a 1080Ti GPU.

\section*{C. PERFORMANCE EVALUATION}
The performances of baselines and the proposed model are reported in table 2. For MSE, the smaller is better. The best result is highlighted with bold font and the most competitive baseline is marked by underline.

From the experimental result, we have the following observations. For collaborative filtering based methods, the performance of BMF is far worse than that of NeuMF, indicating that it is important to replace the inner product by the deep neural network to model the nonlinear relationship between latent factors, which is consistent with the conclusion in [11]. In most cases, the review based methods have better performance than pure collaborative filtering based methods. For topic-based methods or deep learning methods, the implicit information in the reviews is used to enhance the prediction of ratings. However, the topic model belongs to the bag-of-words model, which ignores the contextual information of reviews. Deep learning based methods can take advantage of such contextual information, thus we can see DeepCoNN is usually better than HFT and A3NCF. In the topic-based model, A3NCF is better than HFT in some datasets, which proves the benefits of the fusion of topic factors and latent factors and the significance of using attention mechanisms to distinguish the importance of each factor. A3NCF trains the topic model and predicts ratings in separate phases, that’s why it is worse than HFT in some datasets, which couples topic factors and latent factors together through a transformation. The proposed model, RSBM, is better than all baselines on each dataset. Even on extremely sparse datasets, such as Toys and Games and Beauty, a significant improvement can be seen. There are two main reasons. Firstly, we use the deep leaning method to extract the semantic features of reviews, which can capture the contextual information. Secondly, in the review semantics generation, we simulate the process that the user firstly assesses each aspect of the item and then draws the overall assessment based on the assessment of each aspect, which implicitly expresses that each aspect may have different importance for different users. For the DeepCoNN, we can see that it has a very poor performance on Yelp2014 because each review in Yelp2014 is relative long than the others. It also shows that models in the aggregated paradigm will meet some problems when dealing with long reviews.

\(^1\)http://www.nltk.org/

\(^2\)https://tensorflow.google.cn/
**TABLE 2.** Performance comparisons on nine datasets for all methods (MSE). The best result is marked by bold font and the best baseline is marked by underline. $\Delta$ is used to denote the improvement percent compared to the best baseline.

| #       | BMP   | NeuMF  | HPT    | DeepCoNN | A3NCF  | RSBM   | $\Delta(\%)$ |
|---------|-------|--------|--------|----------|--------|--------|--------------|
| Baby    | 1.2878| 1.1913 | 1.1836 | 1.1871   | 1.1683 | 1.1246 | 3.74         |
| Beauty  | 1.3105| 1.2814 | 1.2341 | 1.2265   | 1.2281 | 1.1659 | 4.94         |
| Digital Music | 0.9575 | 0.8750 | 0.8006 | 0.8569   | 0.8702 | 0.8466 | 1.20         |
| Patio, Lawn and Garden | 1.1125 | 1.0796 | 1.0574 | 0.9136   | 1.0263 | 0.9052 | 0.92         |
| Toys and Games | 0.9450 | 0.8857 | 0.8609 | 0.8475   | 0.8727 | 0.8016 | 5.42         |
| Cell Phones and Accessories | 1.4747 | 1.4374 | 1.3711 | 1.4294   | 1.3728 | 1.3087 | 4.55         |
| Grocery and Gourmet Food | 1.1017 | 0.8923 | 0.8393 | 0.8261   | 0.8684 | 0.7944 | 3.84         |
| Yelp2014 | 1.4187 | 1.3525 | 1.2683 | 1.3346   | 1.2822 | 1.2425 | 2.03         |

**FIGURE 3.** A comparison of different activation functions in the attention network.

**FIGURE 4.** The impact of a different number of aspects.

**FIGURE 5.** The impact of a different dimension of latent factors.

**D. MODEL ANALYSIS**

1) **PARAMETER IMPACT ANALYSIS**

In the vanilla attention mechanism, softmax(.) is used as the activation function in the last layer. In this case, we use tanh(.). To demonstrate their different effects, we set the other parameters to the optimal value and use different activation functions. The results are shown in figure 3. We can see that tanh(.) is better than softmax(.) in most cases because in this case, the output of the attention network represents the user’s weighted sentiments on each aspect. It is more natural to map the weighted sentiment to a value in the range of $[-1, 1]$, so tanh(.) is intuitively more reasonable, and the experimental results verify this intuition.

There are two key parameters in the review semantics generator, one is the dimension of latent factors of a user(or an item) and the other is the number of aspects the user concerned about. We set other parameters to the optimal values and the latent factors to varying dimensions. The result is shown in figure 5. From the result, we can see that with the increase of the dimension of latent factors, the performance gets worse, because a larger dimension means more parameters to be learned and it also introduces the risk of overfitting. We conduct similar experiments with the number of aspects the user concerned about. The result is shown in figure 4. We can see that the number of aspects the user concerned about has less influence on the model because the final semantic features are obtained by the linear combination of the embedding of each aspect, so more aspects mean that each aspect has a finer granularity.

2) **ASPECT VISUALIZATION**

The embedding of each aspect is automatically learned by the model, to investigate the meaning of each aspect more detailly. We set the number of aspects to 32, and run the model on Cell Phones and Accessories. After the learning converged, we save the embeddings of each aspect and each word, then the cosine similarity between them is calculated to measure the relevance. We select some human-readable aspect to report, the most relevant word of these aspects is reported in table 3. From the most relevant word of each aspect, we can infer the meaning of each aspect. For example, the No.3 aspect may express something about the weight of the item and the No.13 aspect may represent the price.

3) **COLD START PROBLEM**

The cold start problem is prevalent in recommender systems [31]. To investigate the performance of the proposed model, we select the users who have reviews less than 4 as cold start users and all records associated with them are
TABLE 3. The most relevant word for some aspects on cell phones and accessories.

| aspect          | word      | aspect   | word  |
|-----------------|-----------|----------|-------|
| 3               | heavy     | 12       | qualification |
| 4               | gesture   | 13       | cheaper   |
| 5               | multilanguage | 20   | installer |
| 8               | fragil    | 22       | texture   |
| 11              | charge    | 26       | bluetooth |

TABLE 4. Performances (MSE) in the setting of cold start.

|            | DeepCoNN | RSBM   |
|------------|----------|--------|
| Baby       | 1.4552   | 1.3561 |
| Beauty     | 1.6268   | 1.4901 |
| Toys and Games | 1.3199 | 1.0046 |
| Yelp2014   | 2.8180   | 2.0181 |
| Patio, Lawn and Garden | 1.7762 | 1.4235 |

TABLE 5. Input length statistics, $l_u$, $l_i$ and $l_s$ denote the length of the user’s aggregated review, the length of the item’s aggregated review and the length of a single review.

| dataset           | $l_u$  | $l_i$  | $l_s$  |
|-------------------|--------|-------|-------|
| Toys and Games    | 1057   | 1917  | 310   |
| Digital Music     | 3146   | 5372  | 591   |
| Patio, Lawn and Garden | 1308   | 2615  | 444   |
| Cell Phones and Accessories | 652    | 2431  | 314   |
| Grocery and Gourmet Food | 1306   | 2416  | 262   |

collected as the dataset to conduct the experiment in the setting of cold start. In the experiment, we randomly reserve one record as the testing set for each user, and the others are used as the training set. The performances of RSBM and DeepCoNN are reported in table 4. From the result, we can see that although performances of both DeepCoNN and RSBM are degraded, RSBM is still better than DeepCoNN. In the setting of cold start, records are not sufficient to learning the parameter of the model, so the performance is more friendly for smaller capacity models. In RSBM, the embeddings of aspects are shared by all users and items, which reduce the load of parameter learning, that is why it still works better in the setting of cold start.

4) SCALABILITY ANALYSIS

DeepCoNN, as a model in the aggregated paradigm, is trained based on the user’s aggregated review and item’s aggregated review. However, the proposed model-RSBM is only trained on a single review, which results in better training efficiency. To prove this point, we report the input length statistics in table 5 and the time to run a batch during the training and prediction phase on the same hardware. From the result, we can see the training and prediction time is propositional to the input length for DeepCoNN. Although the running time increases with the input length in RSBM, a single review length is relatively shorter than the sum of the aggregated review length of users and items, thus it is faster than DeepCoNN in training phase. During the prediction phase, RSBM only uses the review semantics generator and rating regressor, which is free of the review, so the running time is nearly a constant on all datasets.

VII. CONCLUSION AND FUTURE WORK

In this paper, we use a model paradigm that differs from the existing review-based recommender. A single review instead of an aggregated review is used to train the model for improving efficiency and scalability. Specifically, we use a memory-network liked structure to generate the semantic features of the review. In the generation process, we first use a attention network to estimate the assessment of a user on each aspect of an item, then regard these assessments as coefficients to linearly combine the embedding of each aspect to generate the overall assessment semantics, which is similar to the behavior of the user when making a purchase decision. Through a series of experiments on real-world datasets, the proposed model outperforms state-of-the-art methods whatever in rating prediction accuracy or scalability.

In the future work, we will try to extract the aspect keywords and combine them with users and items to build a knowledge graph. The graph-based techniques such as Graph Convolution Network (GCN) [16] can be applied to it to make more accurate recommendations.

REFERENCES

[1] D. Agarwal and B.-C. Chen, “Regression-based latent factor models,” in Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), New York, NY, USA: ACM, 2009, pp. 19–28.
[2] Y. Bao, H. Fang, and J. Zhang, “Topicmf: Simultaneously exploiting ratings and reviews for recommendation,” in Proc. 28th AAAI Conf. Artif. Intell. (AAAI), Québec City, QC, Canada: AAAI Press, 2014, pp. 2–8.
[3] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet allocation,” J. Mach. Learn. Res., vol. 3, pp. 993–1022, Mar. 2003.
[4] R. Catherine and W. Cohen, “Transnets: Learning to transform for recommendation,” in Proc. 11th ACM Conf. Recommender Syst. (RecSys), New York, NY, USA: ACM, 2017, pp. 288–296.
[5] C. Chen, M. Zhang, Y. Liu, and S. Ma, “Neural attentional rating regression with review-level explanations,” in Proc. World Wide Web Conf., Geneva, Switzerland, 2018, pp. 1583–1592.
[6] Z. Cheng, Y. Ding, X. He, L. Zha, X. Song, and M. Kankanhalli, “A3ncf: An adaptive aspect attention model for rating prediction,” in Proc. 27th Int. Joint Conf. Artif. Intell. (IJCAI), Stockholm, Sweden: AAAI Press, 2018, pp. 3748–3754.
[7] Z. Cheng, Y. Ding, L. Zhu, and M. Kankanhalli, “Aspect-aware latent factor model: Rating prediction with ratings and reviews,” in Proc. World Wide Web Conf., Geneva, Switzerland, 2018, pp. 639–648.
[8] Q. Diao, M. Qiu, C.-Y. Wu, J. A. Smola, J. Jiang, and C. Wang, “Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS),” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), New York, NY, USA: ACM, 2014, pp. 193–202.
[9] T. Ebessu, B. Shen, and Y. Fang, “Collaborative memory network for recommendation systems,” in Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR), New York, NY, USA: ACM, 2018, pp. 515–524.
[10] X. Guan, Z. Cheng, X. He, Y. Zhang, Z. Zhu, Q. Peng, and T.-S. Chua, “Attentional aspect modeling for review-aware recommendation,” ACM Trans. Inf. Syst., vol. 37, no. 3, Mar. 2019, Art. no. 28.

[11] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, “Explicit factor models for explainable recommendation based on phrase-level sentiment analysis,” in Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR). New York, NY, USA: ACM, 2014, pp. 83–92.

[12] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft, “Joint representation learning for top-n recommendation with heterogeneous information sources,” in Proc. ACM Conf. Inf. Knowl. Manage. (CIKM). New York, NY, USA: ACM, 2017, pp. 1449–1458.

[13] R. Cao

[14] L. Hu, A. Sun, and Y. Liu, “Your neighbors affect your ratings: On geographical neighborhood influence to rating prediction,” in Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR). New York, NY, USA: ACM, 2014, pp. 345–354.

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

[16] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in Proc. 26th Int. Conf. World Wide Web, Geneva, Switzerland, 2017, pp. 173–182.

[17] X. Wang, X. Yang, L. Guo, Y. Han, F. Liu, and B. Gao, “Exploiting social memory-based attention for collaborative ranking,” in Proc. World Wide Web Conf. (WWW). New York, NY, USA: ACM, 2015, pp. 1746–1751.

[18] P. D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. 3rd Int. Conf. Learn. Representations, (ICLR), San Diego, CA, USA, 2015.

[19] N. T. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in ICLR, Toulon, France, 2017.

[20] Y. Tan, M. Zhang, Y. Liu, and S. Ma, “Rating-boosted latent topics: Understanding rating dimensions with review text,” in Proc. 7th ACM Conf. Recommender Syst. (RecSys). New York, NY, USA: ACM, 2013, pp. 165–172.

[21] N. Pappas and A. Popescu-Belis, “Sentiment analysis of user comments for one-class collaborative filtering over ted talks,” in Proc. 26th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR). New York, NY, USA: ACM, 2013, pp. 773–776.

[22] R. Salakhutdinov and A. Mnih, “Probabilistic matrix factorization,” in Proc. 20th Int. Conf. Neural Inf. Process. Syst. (NIPS). New York, NY, USA: Curran Associates, 2007, pp. 1257–1264.

[23] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted boltzmann machines for collaborative filtering,” in Proc. 24th Int. Conf. Mach. Learn. (ICML). New York, NY, USA: ACM, 2007, pp. 791–798.

[24] X. Tan, M. Zhang, Y. Liu, and S. Ma, “Rating-boosted latent topics: Understanding users and items with ratings and reviews,” in Proc. 25th Int. Joint Conf. Artif. Intell. (IJCAI). New York, NY, USA: AAAI Press, 2016, pp. 2640–2646.

[25] Y. Tay, L. A. Tuan, and S. C. Hui, “Multi-pointer co-attention networks for recommendation,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD). New York, NY, USA: ACM, 2018, pp. 2309–2318.

[26] Y. Tay, L. A. Tuan, and S. C. Hui, “Latent relational metric learning via memory-based attention for collaborative ranking,” in Proc. World Wide Web Conf., Lyon, France, Apr. 2018, pp. 729–739.

[27] X. Wang, X. Yang, L. Guo, Y. Han, F. Liu, and B. Gao, “Exploiting social review-enhanced convolutional matrix factorization for social recommendation,” IEEE Access, vol. 7, pp. 82826–82837, 2019.

[28] W. Zhang and J. Wang, “Integrating topic and latent factors for scalable personalized review-based rating prediction,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 11, pp. 3013–3027, Nov. 2016.