Attention Over Attention Mechanism on Reviews for Rating Prediction

Roy Dor, Nir Kutsky
Ben Gurion University

ABSTRACT

Product reviews is becoming a popular way for users to find reliable, hands-on information when it comes to online purchase decisions in e-commerce platforms. Though not all reviews are equally beneficial, and for some users a certain review can be more helpful than others. Using the reviews information can help in the rating prediction task for item recommendation.

In this paper we present a Neural Attentional Regression model with Review-level Explanations (NARRE) for recommendations which was introduced by Chen et. al. [1], then we propose two extensions to the NARRE model. We compare the original NARRE model to our extensions idea on 2 categories of the Amazon 5-Core dataset [13], Toys & Games and Kindle Store. Our experiments show one of the extension suggested have high potential for improvements. The data and code for this study can be found at https://github.com/nirku/AoAM-NARRE.

1 INTRODUCTION

In recent years, due to the growing number of choices available online, there is a need for a recommender system in e-commerce to help users find items. Among various recommendation methods, collaborative filtering has drawn the great attention from both the academic and industrial communities [17]. Matrix factorization method is the most popular technique in collaborative filtering. Thanks to its easy extendibility and flexibility, there are several variants of matrix factorization methods, such as Non-negative Matrix Factorization (NMF) [23], Probabilistic Matrix Factorization (PMF) [12], etc. However, the rating data is usually highly sparse, leading to that the relevance between users/items is hard to be discovered. To address this problem, various kinds of auxiliary information have been introduced into matrix factorization, such as tags [21], [4], social relations [16], [2], reviews [24], [10], [3], and visual features [22]. In this auxiliary information, the reviews generated from the user behaviors can reflect user preference directly, and it is easy to provide reasonable explanations for recommendation results.

The integration of reviews and ratings has become a popular way to improve recommender system. Text reviews can provide rich useful semantic information for modeling users and items, which can benefit rating prediction in recommendation. Sentiment analysis is the most fundamental and important work in extracting user's interest preferences. In general, sentiment is used to describe user's own attitude on items. Generally, reviews are divided into two groups, positive and negative. Naturally we can see the connection between sentiment and ratings, higher rating - positive review.

However the usefulness of review is varied. Reviews can have many meanings, they can have different biases for different users. for example, some users may prefer the word “good” to describe an excellent product, and others may prefer it to describe an “alright” product. Information from review texts are more abundant than from ratings, and can help us find the user's preferences. In today's online environment you can probably find both positive and negative reviews on any product. Understanding users preferences can help in finding which users are similar in order to give a better recommendation, which consider more relevant reviews to the user. Both positive and negative reviews are valuable to be as reference. For positive reviews, we can know the advantages of a product. For negative reviews, we can obtain the disadvantages and problems. So it's worth to explore those reviewers who have obvious and objective attitude on items. Another challenge handling user generated content is the free-text. Users can write with spelling or typing errors, and even strange abbreviations like “gr8” for the word “great”. This makes in harder to understand what are actual words in the language, and makes our corpus larger.

In this paper we recreate the model offered by Chen et. al. [1], called NARRE, and evaluate it's performance on the updated Amazon 5-Core dataset [13] using Toys & Game and Kindle Store categories. Then we extend their model by changing the prediction layer in attempt to improve the model's rating prediction capabilities. The issue with the original model we have tried to solve is it's linearity. We've tried to change this with: 1. add non-linear Deep Neural Network (DNN); 2. add attention mechanism.

The main contributions of this paper as follows:

(1) Evaluation of the NARRE model on the new updated Amazon dataset.
(2) Extending NARRE with DNN and Attention, and comparing with the original NARRE.

2 PRELIMINARIES

2.1 Latent Factor Model

Latent Factor Model [7] (LFM) is a category of algorithms mostly based on matrix factorization techniques. One of the most popular algorithms of LFM predicts the rating $\hat{R}_{u,i}$, of item i by user u as follows [18]:

$$\hat{R}_{u,i} = q_u p_i^T + b_u + b_i + \mu \qquad (1)$$

Here the equation contains four components: global average rating μ, user bias b_u , item bias b_i and interaction of the user and item $q_u p_i^T$. Further, q_u and p_i are K-dimensional factors that represent user preferences and item features, respectively.

2.2 TextCNN Processor

In recent years, many text processing methods based on deep learning have been proposed and have achieved better performance than traditional methods. Such as FastText [5], TextCNN [6], TextRNN, and paragraph vector[8]. In this paper, we process text using the same approach as the state-of-the-art method described in DeepCoNN [24]. The method will be referred in this paper as TextCNN Processor. The TextCNN Processor inputs a sequence of words and

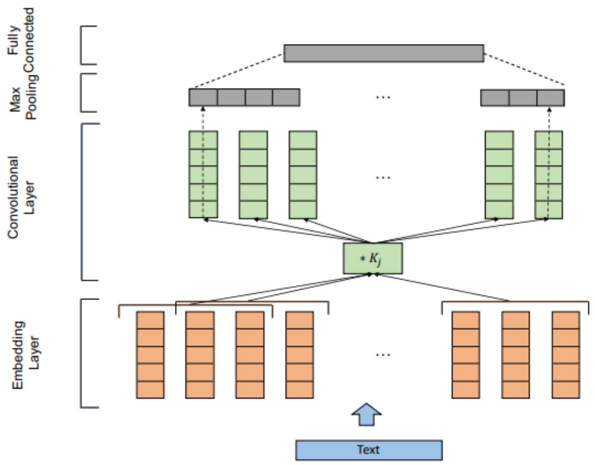

Figure 1: The CNNText Processor architecture.

outputs a n-dimensional vector representation for the input. Figure 1 gives the architecture of the CNNText Processor.

In the first layer, a word embedding function $f : M \rightarrow \mathbb{R}^d$ maps each word in the review into a d dimensional vector, and then the given text will be transformed to a embedded matrix with fixed length T (padded with zero wherever necessary to tackle length variations). The embedding is a pre-trained embedding like those trained on the GoogleNews corpus using Word2Vec [11] or on Wikipedia using GloVe [14].

Following the embedding layer is the convolutional layer. It consists of m neurons, and each associated with a filter $K \in \mathbb{R}^{(t \times d)}$ which produces features by applying convolution operator on word vectors. Let $V1 : T$ be the embedded matrix corresponding to the T-length input text. Then, j_{th} neuron produces its features as:

$$z_j = ReLU(V_{1:T} * K_j + b_j) \tag{2}$$

where b_j is the bias, * is the convolution operation and ReLU is a nonlinear activation function. Let $z1, z2, ...z_j^{(T-t+1)}$ be the features produced by the j_{th} neuron on the sliding windows t over the embedded text. Then, the final feature corresponding to this neuron is computed using a max-pooling operation. The idea behind max-pooling is to capture the most important feature-one with the highest value, which is defined as:

$$o_j = max(z_1, z_2, ...z_j^{(T-t+1)}) \tag{3}$$

the final output of the convolutional layer is the concatenation of the output from its m neurons, denoted by:

$$O = [o_1, o_2, ...o_m] \tag{4}$$

afterwards, the output O is then passed to a fully connected layer:

$$X = WO + g \tag{5}$$

where $W \in \mathbb{R}^{(m \times n)}$ is the weight matrix and $g \in \mathbb{R}^n$ is the bias.

3 RELATED WORK

Models integrating review texts with ratings in recommender systems have attracted a lot of attention. Lei et. al. [9] suggested to use social sentiment, based on user sentiment measurement which is also used to infer item's reputation. Fused user sentiment similarity, interpersonal sentimental influence, and item reputation similarity into a probabilistic matrix factorization. Qiu et. al. [15] modeled aspect-based latent factor. An aspect is not equivalent to topic or attribute. Two phones can both able to make calls (attribute). They found that aspect is more beneficial than topic when comparing items in the same category, because a review text express the user's specific feeling on a specific aspect of the item. This allows to transform features of users, items, words into the same vector space and obtain not only embeddings for users and items, but words latent representation as well. Wang et. al. [20] offer a neural hierarchical attentions and latent factors. In their paper they propose to encode reviews with word level attention followed by review level attention guided by latent factors. Fused together to focus on important words and informative reviews.

4 METHODOLOGY

In this section we present the NARRE model architecture. Then explain our changes to it's prediction layer in 4.2 and 4.3.

4.1 NARRE

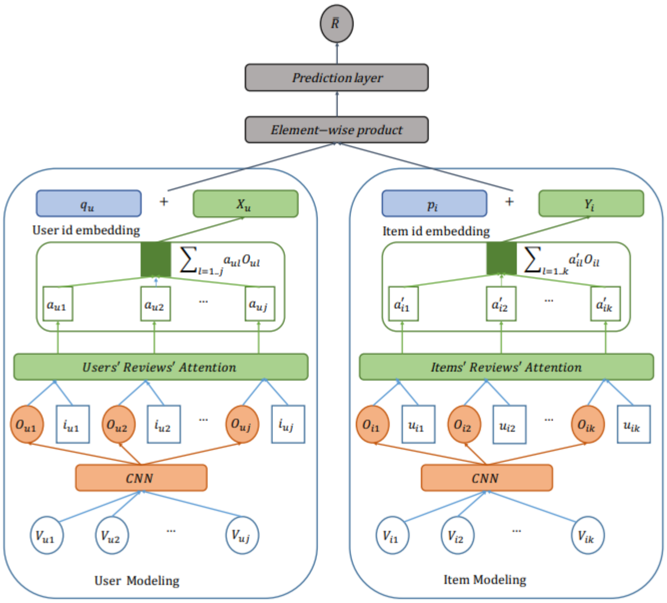

Figure 2: The neural network architecture of NARRE. Its attention model uses both IDs(i_{uj}, u_{jk}) and review content(O_{uj}, O_{jk}) to automatically assign weights to reviews.

The model consists of two parallel neural networks, one for user modeling (Net_u), and another for item modeling (Net_i). On the top of the two networks, a prediction layer is added to let the hidden latent factors of each network (user and item) interact with each other and calculate the final prediction of the model. The architecture of the model is shown in Figure 2.

In the following: we will describe in details each part of the model, since Net_u and Net_i are almost identical except their inputs, we will focus on the process for Net_i, but the same process is also applied for Net_u.

In the first part of Net_i, CNNText Processor is applied on the textual reviews of item i. Each review is first transformed into a matrix of word vectors, which is donated by $V_{i1}, V_{i2}, ..., V_{i_k}$. Then, this matrices are sent to the convolutional layer and transform into a feature vectors $O_{i1}, O_{i2}, ..., O_{i_k}$. Since each feature vector does not equally useful and representative to item i representation [1], we apply attention mechanism into the model, which can help to learn the weight of each review.

4.1.1 *Attention on Reviews.*

The goal of the attention on reviews in Net_i is to select reviews that are representative to item i's features and then aggregate the representation of informative reviews to characterize item i. The attention score a_{il} is being calculated using a two-layer network. First, the input contains the feature vector of the lth review of item i (O_{il}) and the user who wrote it (ID embedding, u_{il}). The ID embedding is added to the model in order to identify users who always write less-useful reviews, and in a more formal way, the attention is defined as:

$$a^*_{il} = h^T ReLU(W_O O_{il} + W_u u_{il} + b1) + b2 \tag{6}$$

where $W_O \in \mathbb{R}^{t \times k_1}, W_u \in \mathbb{R}^{t \times k_2}, b_1 \in \mathbb{R}^t, h \in \mathbb{R}^t, b_2 \in \mathbb{R}^1$ are the model parameters and t denotes as the hidden layer size of the attention network.

The final weights of reviews are obtained by normalizing the above attention scores using the softmax function, which can transform the given scores into probabilities in a range of [0-1].

$$a_{il} = \frac{e^{a^*_{il}}}{\sum_{l=0}^{k} e^{a^*_{il}}} \tag{7}$$

a_{il} represent the contribution of the lth review to the feature profile of item i. After obtaining the attention weight of each review, we then calculate the weighted feature vector of each review of item i and summing them together to get a single representation.

$$O_i = \sum_{l=1}^{k} a_{il} O_{il} \tag{8}$$

the output is a k_1 dimensional vector, which compresses all reviews of item i in the embedding space by distinguishing their contributions. Then it's sent to a fully connected layer with $W_0 \in \mathbb{R}^{n \times k_1}$ and bias $b_0 \in \mathbb{R}^n$, which computes the final representation of item i:

$$Y_i = W_0 O_i + b_0 \tag{9}$$

4.1.2 *Prediction Layer.*

The final part of the model is the combination of Net_u and Net_i that we discussed in the previous parts, together with the traditional latent factor model (LFM), which extends the user preferences and item feature using the LFM model. We created a neural form LFM for predicting ratings, based on ratings and reviews.

Specifically, the latent factor of user and item are first mapped to the same hidden space learned from the reviews, the interaction between user u and item u is modelled as:

$$h_0 = (q_u + X_u) \odot (p_i + Y_i) \tag{10}$$

where q_u and p_i are user preferences and item feature based on ratings as Eq.(1), X_u and Y_i are user preferences and items features obtained from Net_u and Net_i, and $\odot$ denotes the element-wise product of vectors. The output is n dimensional vector, then it is passed to the prediction layer to get a real-valued rating $\hat{R}_{u,i}$:

$$\hat{R}_{u,i} = W_1^T h_0 + b_u + b_i + \mu \tag{11}$$

where $W_1 \in \mathbb{R}^n$ denotes as the weights of the prediction layer, b_u, b_i and μ denotes user bias, item bias ad global bias respectively.

4.2 NARRE DNN

The first extension we've examined is adding a DNN part to the prediction layer. By adding non-linear layers to the prediction layer of NARRE we expect the model to find non-linear connections between the interactions of user u and item i described in Eq. 10. The model output layer remains as described in equation 11, only the value of h_0 has changed to the output of the DNN. The DNN part contain several fully connected layers with ReLu as activation function.

Experiments were conducted on various parameters such as activation functions, number of layers, adding dropout etc. We've found the parameters described in section 5.1.3 to perform best.

4.3 NARRE Feature Attention

The second hypothesis we've examined is applying feature-level attention mechanism [19] which focuses on the contribution of different features to relation extraction, instead of the simple concatenation that was done on the features we received from both models: Net_u and Net_i.

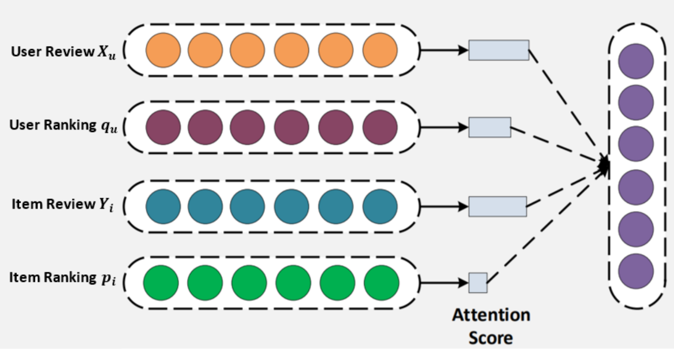

Figure 3: Feature-level Attention Mechanism.

Each of the features is getting an attention score that will help to calculate the final representation as it described in figure 3. The formalization of feature-level attention mechanisms is defined as:

$$Q = \tanh([W_1^T \cdot W_2] + b_\omega) \tag{12}$$

where W_1 is one of the features vector, $W_2 \in \mathbb{R}^{n \times k_1}$ is the attention weights and $b_\omega \in \mathbb{R}^n$ is the bias. To get the attention scores we need to squash the weights into probabilities between [0-1], we will use the softmax function that was mentioned in Eq. 7:

$$Q' = SoftMax(Q) \tag{13}$$

Q' is the vector of probabilities, which focus on the contribution of each element, and will help to compute the final feature representation using element-wise product:

$$V = Q' \odot W_1 \tag{14}$$

after obtaining the final representation for each of the features we will pass it to the same fully connected layers that was described in the original paper.

5 EVALUATION

5.1 Experiments

This section conducts extensive experiments to quantitatively and qualitatively to evaluate our model. Section 5.1.1 gives a brief introduction to the datasets and preprocessing steps used throughout the experiments. In Sect. 5.1.2, the evaluation metrics used to compare between the different models. Section 5.1.3 presents hyper-parameters used for training the models.

Table 1: Statistical details of the datasets

	Toys_and_Games	Kindle_Store
# Users	19,412	68,223
# Items	11,924	61,934
# Ratings/Reviews	150,838	884,358

5.1.1 *Datasets.*

The model evaluation was made on the public Amazon 5-Core dataset [13][1]. The original dataset consists of 29 domains, covering 233.1 million reviews collected between May 1996 - Oct 2018. Amazon is the largest Internet retailer that has accumulated large-scale user-generated reviews. The helpfulness of the reviews is rated by online customers, offering an ideal source for the review helpfulness prediction task. Amazon product reviews are predominantly used and analyzed in previous studies. Thus, adopting Amazon reviews allows for fair comparison with previous studies. The analysis results can also provide practical insights into online business and user-generated content quality evaluation.

For experiment purposes, two domains are selected for evaluation, **Kindle_Store**, a large dataset that contains almost 900 thousand reviews and **Toys_and_Games** the smallest dataset which contains about 150 thousand reviews. Since the raw data is very large and sparse, preprocess was done to ensure that all users and items have at least five ratings. The ratings of these datasets are integers in the range of [1, 5]. Since the length and the number of reviews have a long tail effect, we only keep the length and the number of reviews covering $p = 0.9$ percent users and items respectively. The characteristics of our datasets are summarized in Table 1.

5.1.2 *Evaluation metrics.*

The task we are dealing within this paper is rating prediction, which is a regression problem. For regression, a commonly used objective function is the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE). Since those metrics measure errors, our goal is to minimize them.

$$RMSE(y, \hat{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{15}$$

$$MAE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{16}$$

5.1.3 *Hyperparameters.*

For best performances we have tested a range of hyper-parameter values that are summarized in table 2. The best results were obtained when we used embedding layer with 300 dimensions, convolutional layers with 100 filters and size (3x3), dropout prob. of 0.2, l2-lambda regresssion of $1e^{-3}$, hidden latent size of 32, batch size of 100 and 40 epochs.

Table 2: Ranges of Hyperparameters Searched For Optimization of Performances

Hyperparameter	Range
Embedding Dimensions	{ 100, 300, 500 }
Filter Sizes	{ 2, 3, 4 }
No. of Filters	{ 100, 128, 256 }
Dropout Keep Probability	{ 0.2, 0.3 }
L2 Lambda Regression	{ $1e^{-3}$, $1e^{-4}$, $1e^{-5}$ }
Latent Size	{ 8, 16, 32, 64 }
Batch Size	{ 32, 64, 100 }
No. of Epochs	{ 20, 30, 40 }

5.2 Performance Comparison

The rating prediction results of the NARRRE model and our extensions of it on all datasets are given in Table 3. From the results, several observations can be made:

On both datasets NARRE's MAE performance was the worse in comparison to the DNN and attention models. Even when they performed better on RMSE, such as on the Toys dataset where the NARRE had the best RMSE score. This may occur due to the relation between RMSE and MAE metrics. RMSE gives a relatively high weight to large error. Taking the square root of the average squared errors has some interesting since the errors are squared before they are averaged. Meaning that RMSE is less impact by errors smaller than 1, and heavily impact by errors greater than 1, while MAE penalties are linear. Cases when RMSE dropped but MAE increased, as to when comparing results of NARRE to the other models on Toys, means the NARRE model is better at accounting for extreme cases, but the solution may be less robust.

The DNN variation didn't had much impact to the original NARRE model. While experimenting on many sizes and variations of DNN architectures, we didn't find a setting where the improvement was made.

The attention model outperformed the other models when evaluating the Kindle dataset on both metrics. It's improvement is by 3.4% for RMSE and 5.6% on MAE. Our hypothesis to why the attention improved performance on Kindle and not Toys is the size of the training data. The Kindle dataset is almost 6 times the Toys dataset, allowing the attention mechanism to learn more and make a significant impact.

Table 3: Performance Comparison

	Toys		Kindle	
	RMSE	MAE	RMSE	MAE
NARRE	**0.8803**	0.6442	0.7826	0.5526
NARRE DNN	0.8805	**0.5523**	0.7836	0.5507
NARRE Attention	0.8961	0.605	**0.7554**	**0.5214**

[1] https://nijianmo.github.io/amazon/index.html

5.3 Sensitivity Analysis

We've tested how sensitive the models RMSE to changes to the size of the latent factors the models learn during training. The analysis was made on the Toys & Games dataset only because of limited computational resources. Examining Figure 4 we can see correlation with observations describes in Section 5.2.

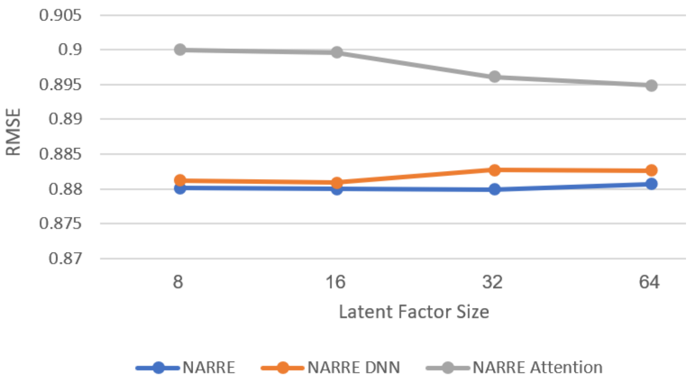

Figure 4: Sensitivity Analysis of the latent factors size to RMSE on the Toys % Games dateset.

The attention mechanism require more data to perform as well as the original NARRE. We can see it's performance is the worse throughout all the latent factors size we've tested. When the attention model has more latent factors to learn it start to compensate on the lack of data, and improve it's performance. Note that both NARRE and the NARRE DNN results are deteriorating with more latent factors, while the attention version is improving. This observation support our theory that the attention mechanism in the prediction layer can in fact improve the model by enabling it to learn more.

6 CONCLUSION

Post-purchase reviews play a very import role for user's purchasing behavior. However, it is hard for users to find useful information from an immense amount of reviews. In this paper, we propose a neural attentional extension model to the NARRE which simultaneously predicts precise user ratings to the item, as proposed by [1]. Our experiments conducted on 2 categories from the Amazon 5-Core dataset show the attention mechanism added in the predictive layer can be beneficial for the model, But needs to be used in cases when there is large amount of data to learn from. The sensitivity analysis made on the size of the latent factors shows that higher latent factors can help compensate for the shortage of data.

REFERENCES

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

[2] Jiawei Chen, Can Wang, Qihao Shi, Yan Feng, and Chun Chen. 2019. Social recommendation based on users' attention and preference. *Neurocomputing* 341 (2019), 1–9.

[3] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. 2018. Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proceedings of the 2018 world wide web conference*. 639–648.

[4] Ruoran Huang, Nian Wang, Chuanqi Han, Fang Yu, and Li Cui. 2020. TNAM: A tag-aware neural attention model for Top-N recommendation. *Neurocomputing* 385 (2020), 1–12.

[5] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* (2016).

[6] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

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

[8] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[9] Xiaojiang Lei, Xueming Qian, and Guoshuai Zhao. 2016. Rating prediction based on social sentiment from textual reviews. *IEEE transactions on multimedia* 18, 9 (2016), 1910–1921.

[10] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*. 165–172.

[11] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[12] Andriy Mnih and Russ R Salakhutdinov. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*. 1257–1264.

[13] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 188–197.

[14] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[15] Lin Qiu, Sheng Gao, Wenlong Cheng, and Jun Guo. 2016. Aspect-based latent factor model by integrating ratings and reviews for recommender system. *Knowledge-Based Systems* 110 (2016), 233–243.

[16] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, Lexing Xie, and Darius Braziunas. 2017. Low-rank linear cold-start recommendation from social data. In *Thirty-first AAAI conference on artificial intelligence*.

[17] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).

[18] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-boosted latent topics: Understanding users and items with ratings and reviews.. In *IJCAI*, Vol. 16. 2640–2646.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[20] Xianchen Wang, Hongtao Liu, Peiyi Wang, Fangzhao Wu, Hongyan Xu, Wenjun Wang, and Xing Xie. 2019. Neural Review Rating Prediction with Hierarchical Attentions and Latent Factors. In *International Conference on Database Systems for Advanced Applications*. Springer, 363–367.

[21] Zhenghua Xu, Thomas Lukasiewicz, Cheng Chen, Yishu Miao, and Xiangwu Meng. 2017. Tag-aware personalized recommendation using a hybrid deep model. AAAI Press/International Joint Conferences on Artificial Intelligence.

[22] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. 2018. Aesthetic-based clothing recommendation. In *Proceedings of the 2018 World Wide Web Conference*. 649–658.

[23] Sheng Zhang, Weihong Wang, James Ford, and Fillia Makedon. 2006. Learning from incomplete ratings using non-negative matrix factorization. In *Proceedings of the 2006 SIAM international conference on data mining*. SIAM, 549–553.

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