Combining Local Convolution with Global Self-Attention For Rating Prediction

Hansi Zeng (u1265961)
Department of Computer Science, University of Utah

Yuching Hsu (u1264420)
Department of Computer Science, University of Utah

ABSTRACT

Using reviews to learn user and item representations is important for recommender system. Current review based methods can be divided into two categories:(1) CNN based models that extract n-gram feature from user/item reviews;(2) RNN based models that learns review global contextual representation for users and items. However, both of the CNN and RNN based models suffer from their own drawbacks. The CNN based model are weak in modeling long-dependency relation in text and RNN based models are slow in training and inference due to unable to extract feature in parallel way. To alleviate the problem, we propose a new text encoder module which consists exclusively of convolution layer and self-attention layer, where convolution layer models the local interactions and self-attention layer models the global interactions of text. Furthermore, different words, sentences, reviews have different importance for modeling user and item representations. To capture the characteristic, we model our system hierarchically in three level: sentence-level, review-level, user/item level. In sentence-level, we learn a sentence representations from its words by the following two procedures: (1) encoding word contextual embeddings using our new encoder module; (2) select important words using attention netowrk. And in review-level, we use the same procedure above but learn a review representation from its sentences. In user/item level, we use the attention network to select reviews with more informativeness. The experiment in four datasets from Amazon Product Dataset shows that our model can achieve improvemtns compared with the state-of-art review based models.

1 INTRODUCTION

Utilizing reviews to model user and item representation can get a better results compared with CF-based model. For example, DeepCoNN[1] use two convolution nerual networks(CNN) to learn user and item representation from reviews, and NARRE[3] extends the model by applying attention mechanism over review level to select more "useful" review from user or item. Besides the CNN based models, other works like HANN[4] use gated recurrent unit(GRU) to model the user and item embeddings from reviews. However, both of the CNN based models and RNN based models have their own weakness. Although the CNN based models are good at capturing local interactions from text (like capturing key phrases), they are weak in learning global interactions from text. The RNN based models are good at learning long range dependencies from reviews but suffer from slow training due to unable to extract feature in parallel way.

To overcome the shortcomings of CNN and RNN based models, we build a new module to encode text information by exclusively using convolutions and self-attention[5] layer. The key intuition behind the design is that: we use the convolution layer to capture local interaction between words like some key-phrases, and use the self-attention layer to learn global interaction from text. Capturing a long dependency relation of text is really important to improve our model performance. For example, a sentence "this version is not classic like its predecessor, but its pleasures are still plentiful." contains a local strong indicator for one sentiment polarity, like "is not" above, but the successful classification relies on the comprehension of the whole sentence. Also, the module can be fully trained parallelly, which can be faster in training and inference compared with RNN based models.

Actually, different reviews have different usefulness to influence user's purchasing behavior. For example, a review written by a user with detail experience and evaluation on a certain item is more useful than that only contains a few words like "I like the cell phone". Also, different sentences in the same review might provide different informativeness. For instance, the sentence "the cellphone has very long battery life." is more informative than "I bought the cell phone yesterday". Besides, in the same sentence, the words like "great", "convenient" that reflect the user's preferences or item's properties might be more important than other words.

In our paper, we propose a new model HCSA to learn user and item representations in hierarchical way. More precisely, we construct three encoder modules to capture the review information hierarchically from word-level, sentence-level, review-level to user/item-level. In sentence-level, we encode a sentence from its words by stacking the following three layers: (1) a word-level convolution layer to extract the n-gram features from the words; (2) a self-attention layer to capture the long-dependencies relation of words; (3) a aggregation layer using attention network to select words with more informativeness from the sentence. Similarly, in sentence level, we learn a review representation from its sentences by stacking the same three layers of review level, which can capture the local and global interactions between sentences and select more important sentences. Finally, in user/item level we learn user and item representations from their reviews using attention network to select reviews containing more informativeness.

2 RELATED WORKS

The matrix factorization (MF)[14] has become the most popular collaborative filtering (CF) technique in recommendation system, which uses historical records such as clicks, ratings and consumption[13]. The original MF model was designed to find the user-item relationship by mapping users and items preference into a latent factor space. Although the MF based model has shown a good result, it will degrade when the rating matrix becomes very sparse. Furthermore, it cannot give a explanations of why the item is worth recommending or not.

To overcome these problems, the topic model was integrated into the original MF based model to generate the latent factors for users and items incorporating review texts, such as HFT[15], RMR[16],

EFM[17], TriRank[18], RBLT[19], and sCVR[20]. These modules first extract explicit product features and user opinions by phrase-level sentiment analysis on user reviews, then produce the feature-level explanations according to user reviews. However, the phrase-level sentiment analysis in these methods is simply extracting words or phrases from the review text, which will change the integrity of reviews. Furthermore, the textual similarity in above methods is solely based on lexical similarity, which means the semantic meaning will be ignored.

Hence, to overcome this problem, many review based neural network models have been proposed[1]. The neural network model analyze the semantic meaning of the review text and then provide the explanation for user-item relationship. However, the methods regards each review as a long sentence, and cannot distinguish informative sentences and words from less informative one.

Recently many recommendation models combine neural network with review data to learn user and item representations like DeepCoNN[1], TransNets[2], D-Att[7], NARRE[3], HUITA[8], HANN[4]. These models can be roughly divided into two categories:(1) CNN based model; (2) RNN based model. For example, the CNN based model DeepCoNN use two convolution neural networks to learn user and item representations from reviews. And D-Att extends the DeepCoNN by using attention neural network over word-level to select important words. Also, NARRE uses attention network over review-level to select reviews with more informativeness. Besides, the RNN based model like HANN encodes reviews using GRU and uses attention networks in word-level and review-level. However, the shortcomings of the CNN based models is that they are weak in capturing long dependency relation in reviews, and RNN based models cannot train models in a parallel way, which results in the slow training and inference of the model.

To overcome the problems of RNN and CNN models, we propose a text encoder module by stacking convolution and self-attention layer, where convolution layer captures text local feature and self-attention layer learns text global feature. Also, we use the encoder module hierarchically to flexible pay varying attentions to reviews for user/item representations.

3 PROPOSED METHOD

To learn user and item representations, our model hierarchically encode information from reviews, see Figure 1. There are three major modules to encode information: a sentence encoder that learns the embedding of a sentence from its words; a review encoder which learns the representation of a review from its sentences; a user/item encoder which learns user/item representation from its reviews.

3.1 Sentence Encoder

The sentence encoder is a module that learns a sentence representation from words, and there are three layers in the module.

The first layer is a word embedding layer that maps a sequence of words from a sentence into corresponding dense vectors. Mathematically, assume a sentence s_i consist of a sequence of word $\{w_{i,1}, \dots w_{i,t}\}$ with size of t. The embedding layer transforms the word sequence into dense vectors $\{e_{i,1}, \dots e_{i,t}\}$, where $e_{i,*} \in \mathbb{R}^d$.

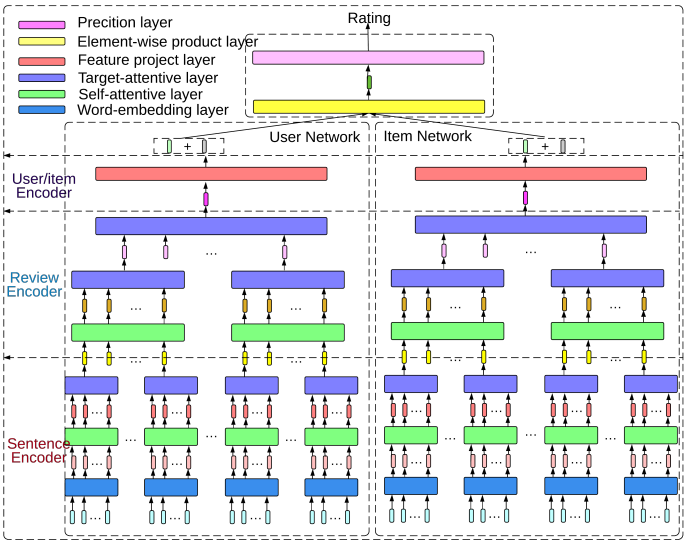

Figure 1: Architecture of the proposed method

We denote the word embedding layer as $\mathbf{E} \in \mathbb{R}^{V \times d}$, where V is the word vocabulary size, d is the dense vector dimension.

The second layer is a stack of the basic building blocks: [convolution layer, multihead self attention layer, feed-forward layer]. The input of the layer is word embedding sequence $\{e_1, \dots e_t\}$ of t elements where $e_i \in \mathbb{R}^d$, and the output is $\{z_1^w, \dots z_t^w\}$ with the same size as input where $z_i \in \mathbb{R}^d$. We apply convolution layer first to learn n-gram feature of the input sequence. Mathematically, for a sequence of word embedding $\{e_1, \dots e_t\}$, we denote $c_i \in \mathbb{R}^{dn}$ as a concatenation of word embeddings $e_{i-\lfloor \frac{n-1}{2} \rfloor}, \dots, e_{i+\lfloor \frac{n-1}{2} \rfloor}$, where n is the width, and d is embedding dimension. Then we can learn the n-gram feature of q_i^w, k_i^w, v_i^w from $e_{i-\lfloor \frac{n-1}{2} \rfloor}, \dots, e_{i+\lfloor \frac{n-1}{2} \rfloor}$ by using the convolution weights $\{W_Q^w, W_K^w, W_V^w\} \in \mathbb{R}^{d \times nd}$ and biases $\{b_Q^w, b_K^w, b_V^w\} \in \mathbb{R}^d$ respectively:

$$q_i^w = Relu(W_Q^w c_i + b_Q^w) \tag{1}$$

$$k_i^w = Relu(W_K^w c_i + b_K^w) \tag{2}$$

$$v_i^w = Relu(W_V^w c_i + b_V^w) \tag{3}$$

Then we apply the multihead self attention layer in the outputs of the convolution layer. We use self attention since it can capture the long-range dependencies of the sequence, which can offers complementary information with CNN. Formally, we split $(q_i^w, k_i^w, v_i^w)_{i=1}^t$ into H heads, denoted as $(q_i^{w,h}, k_i^{w,h}, v_i^{w,h})_{i=1}^t$ for each head, where $\{q_i^{w,h}, k_i^{w,h}, v_i^{w,h}\} \in \mathbb{R}^{d_h}$. We apply the same operation to each head. Furthermore we use relative position[6] to consider the pairwise relation between input sequences. each head's output $z_i^{w,h}$ is the weighted sum of transformed inputs with a learnable vector $r_{ij}^V \in R^{d_h}$ encoding the relation between word i and word j,

$$z_i^{w,h} = \sum_{j=1}^{t} a_{ij}^{w,h} (v_j^{w,h} + r_{ij}^V) \tag{4}$$

each weighted coefficient is computed using a softmax function,

$$a_{ij}^{w,h} = \frac{\exp(e_{ij}^{w,h})}{\sum_{k=1}^{t} \exp(e_{ik}^{w,h})} \tag{5}$$

And the $e_{ij}^{w,h}$ is computed as the inner product between two elements,

$$e_{ij}^{w,h} = \frac{< q_i^{w,h}, k_j^{w,h} + r_{i,j}^K >}{\sqrt{d_h}} \tag{6}$$

where $r_{i,j}^K \in \mathbb{R}_h^d$ is a learnable parameter. Noted the $r_{i,j}^K$ and r_{ij}^V are the same for all heads. After that, we concatenate each head's output $z_i^{w,h}$, and get z_i^w. For each z_i^w we apply a 1 layer feed-forward function with weights W_f^w, bias b_f

$$z_i^w = \text{Relu}(W_f^w z_i^w + b_f^w) \tag{7}$$

Then output of the second layer is $\{z_1^w, \dots z_t^w\}$

The third layer is the word-level attention network. Different words have different importance even in the same sentence. For example, a sentence "I am really impressed with the cellphone's design". The word "impressed" is more important than "I", as it contains the user evaluation on the item. To capture the characteristic, we apply the attention network to select words based on their importance in the sentence. Formally, the $i-th$ word importance score is computed as,

$$a_i^w = h_w^T \text{Relu}(W_p^w z_i^w + b_p^w) \tag{8}$$

$$a_i^w = \frac{\exp(a_w^i)}{\sum_{j=1}^{t} \exp(a_j^s)} \tag{9}$$

where $W_p^w \in \mathbb{R}^{d \times d_p}$, $b_p^w \in \mathbb{R}^{d_p}$, $h_w \in \mathbb{R}^{d_p}$ are learnable parameters, and in our model we set $d_p = \frac{d}{4}$. Then the sentence representation s is the weighted sum of contextual embedding of each word,

$$s = \sum_{i=1}^{t} a_i^w z_i^w \tag{10}$$

3.2 Review Encoder

The review encoder is a module that learns a review representation from its sentences, which consist of two major layers. The first layer learns a contextual representation for each sentence by consider local and global relationships between sentences. And the second layer aggregates these contextual embeddings into a single vector as a representation of the review.

The first layer is a stack of the three building blocks [convolution layer, multihead self attention layer, feed-forward layer]. For each review, the input of the layer is the sequence of sentence embeddings $[s_1, \dots s_n]$ with the size of n where $s_i \in \mathbb{R}^d$, and the output is the sentence contextual vectors $[z_1^s, \dots z_n^s]$ where $z_i \in \mathbb{R}^d$. In this layer, the convolution is used first to capture the local interactions between sentences. Formally, $c_i \in \mathbb{R}^{dn}$ as a concatenation of sentence embeddings $s_{i-\lfloor \frac{n-1}{2} \rfloor}, \dots, s_{i+\lfloor \frac{n-1}{2} \rfloor}$, where d is the dimension and n is the local window size. Then for $i-th$ sentence, we can learn its local contextual representation q_i^s, k_i^s, v_i^s by using the convolution weights $\{W_Q^s, W_K^s, W_V^s\} \in \mathbb{R}^{d \times nd}$ and biases $\{b_Q^s, b_K^s, b_V^s\} \in \mathbb{R}^d$ respectively,

$$q_i^s = Relu(W_Q^s c_i + b_Q^s) \tag{11}$$

$$k_i^s = Relu(W_K^s c_i + b_K^s) \tag{12}$$

$$v_i^s = Relu(W_V^s c_i + b_V^s) \tag{13}$$

To capture the global interactions between sentences among the same review, we use the multihead self attention layer at the outputs of the convolution layer. Mathematically, the sequences $(q_i^s, k_i^s, v_i^s)_{i=1}^n$ are split into H heads, denoted as $(q_i^{s,h}, k_i^{s,h}, v_i^{s,h})_{i=1}^n$, where $\{q_i^{s,h}, k_i^{s,h}, v_i^{s,h}\} \in \mathbb{R}^{d_h}$. The $i-th$ head's output is the weighted sum of learned contextual sentences inputs with a learnable parameters $l_{ij}^V \in \mathbb{R}^{=}$ which is the pairwise relation encoding between sentence i and j,

$$z_i^{s,h} = \sum_{j=1}^{t} a_{ij}^{s,h} (v_j^{s,h} + r_{ij}^V) \tag{14}$$

each weighted coefficient is computed using a softmax function

$$a_{ij}^{s,h} = \frac{\exp(e_{ij}^{s,h})}{\sum_{k=1}^{t} \exp(e_{ik}^{s,h})} \tag{15}$$

And the $e_{ij}^{s,h}$ is computed by the inner product between two input elements,

$$e_{ij}^{s,h} = \frac{< q_i^{s,h}, k_j^{s,h} + l_{i,j}^K >}{\sqrt{d_h}} \tag{16}$$

where $l_{i,j}^K$ is also a learn vector similar to $l_{i,j}^V \in \mathbb{R}^{d_h}$ encodes the sentence i, j's pairwise relation. For $i-th$ sentence, we concatenate its output of each head $z_i^{s,h}$, and get z_i^s. To better learn the representation of z_i^s, we use it as a input into 1 layer feed-forward network parameterized with weights W_f^s, bias b_f^s,

$$z_i^s = Relu(W_f^s z_i^s + b_f^s) \tag{17}$$

Then the $\{z_1^s \dots z_n^s\}$ is the final contextual sentences embeddings.

The second layer is the sentence-level attention layer. The intuition behind the layer is that different sentences have different informativeness in a review. For instance, there are two consecutive sentences in a cellphone review: "I bought a cellphone yesterday. I am really impressed by its design.". We can conclude that the second might be more informative since it contains a phrase describing a cellphone's aspect "its design" with evaluation "impressed". To select more informative sentence from a review, we apply a attention network to compute each sentence informativeness score. Mathematically, the $i-th$ sentence informativeness score is computed as,

$$a_i^s = h_w^T \text{Relu}(W_p^s z_i^s + b_p^s) \tag{18}$$

$$a_i^s = \frac{\exp(a_s^i)}{\sum_{j=1}^{t} \exp(a_j^s)} \tag{19}$$

where $W_p^s \in \mathbb{R}^{d \times d_p}$, $b_p^s \in \mathbb{R}^{d_p}$, $h_s \in \mathbb{R}^{d_p}$ are learnable parameters, and in our model we set $d_p = \frac{d}{4}$. Then the review representation is a weighted sum of contextual sentence representations,

$$r = \sum_{i=1}^{n} a_i^s z_i^s \tag{20}$$

3.3 User/Item Encoder

We learn user/item representations from their reviews. Motivated by the model NARRE[3], different reviews have different informativeness. For example a review written by a experience user with detail description of different aspects of a certain item might provide more information than those reviews only consist of a few words, like "I bought the cellphone yesterday, and it is good". To select more useful reviews of a user, we use attention network that can assign more weights for those reviews with more informativeness. Formally, we assume a user $\mathbf{P}^u$ writes m reviews $r_1, \dots r_m$, then for the $i-th$ review r_i, its important score is computed as

$$a_i^r = h_r^T \text{Relu}(W_p^r z_i^r + b_p^r) \tag{21}$$

$$a_i^r = \frac{\exp(a_i^r)}{\sum_{j=1}^{t} \exp(a_j^r)} \tag{22}$$

where $W_p^r \in \mathbb{R}^{d \times d_p}$, $b_p^r \in \mathbb{R}^{d_p}$, $h_r \in \mathbb{R}^{d_p}$ are learnable parameters, and in our model we set $d_p = \frac{d}{4}$. The final representation for user u can be computed as the weighted sum of their review embeddings,

$$u^r = \sum_{i=1}^{m} a_i^r r_i \tag{23}$$

3.4 Prediction Layer

we first project user representation u^r learned from reviews into the latent space using one layer feed-forward network,

$$\overline{u^r} = \text{Relu}(W_u u^r + b_u) \tag{24}$$

where $W_u \in \mathbb{R}^{d_h \times d_l}$, $b_u \in \mathbb{R}^{d_l}$. Similarly using the equation above, we can get the projected item representation $\overline{v^r}$. Also, we introduce the user and item ID embeddings $\overline{u^d}$, $\overline{v^d}$ where $\overline{u^d} \in \mathbb{R}^{d_l}$, $\overline{v^d} \in \mathbb{R}^{d_l}$, then the interaction between user and item can be modeled as,

$$h = (\overline{u^d} + \overline{u^r}) \odot (\overline{v^d} + \overline{v^r}) \tag{25}$$

Then we can get the final rating prediction by using the following equation,

$$\hat{R}_{uv} = W_f^T h + b_u + b_v + b_g \tag{26}$$

where $W_f \in \mathbb{R}^{d_h}$, $b_u, b_v, b_g \in \mathbb{R}$. The b_u represents user bias, b_v is item bias, and b_g is the global bias.

When training model, we minimize the loss between ground truth rating R_{uv} with predicted rating $\hat{R}_{uv}$. We use the mean square error as the loss function,

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} (R_{uv} - \hat{R}_{uv})^2 \tag{27}$$

where N is the number of user item pairs in the training dataset.

4 EXPERIMENT

4.1 Datasets

In our experiment, we use four datasets from different categories of Amazon product dataset[12]. The four categories are 5-core , **Toys and Games**,**Digital Music**, **Movies and TV** and **Office Products**. The statistic of these four dataset are shown in the Table 1.

For data preprocessing, We randomly split each dataset into 80% training set, 10% validation set, and 10% testing set. And it should be noted that the reviews from validation set and testing set are not included in the training set. And we use NLTK[21] to tokenize our sentences and words. Since the distribution of length and the number of reviews, we only keep the number of reviews cover p_1 and length of reviews cover p_2 users and items. In **Toys and Games**,**Digital Music**, **Office Products** we set $p_1 = 0.9$, and in **Movies and TV** dataset we set $p_1 = 0.85$. and $p_2 = 0.7$ since we find the setting is good for our model both at efficiency and performance.

Table 1: The statistics of two datasets

	Toys and Games	**Digital Music**	**Office Products**	**Movies and TV**
users	19,412	5,541	4,905	123,960
items	11,924	3,568	2,420	50,052
rating & reviews	167,597	64,706	53,258	119,441

4.2 Evaluation Metric

we use Root Mean Square Error(RMSE) to evaluate our recommender system. The lower the score is, the better our system is. Formally, assume there are N user item rating pairs, and we denote the predicted rating of pair user u, item v is $\hat{R}_{uv}$, ground truth rating is R_{uv}. Then the RMSE is computed as,

$$MSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\hat{R}_{uv} - R_{uv})^2}$$

4.3 Compared Methods

we compare our model with state-of-art review based methods, including DeepCoNN[1], NARRE[3], D-ATT[7]. specially, the DeepCoNN use two CNN to learn the user and item representations from their reviews, and feed the representations into Factorization Machines(FM)[9] to get the final rating prediction. The NARRE extends the DeepCoNN by using the attention mechanism over review level to select reviews with more informativeness. And D-Att [7] also improves the DeepCoNN by adding dual-attention layer at the word-level before convolution.

4.4 Experimental Settings

We use pretrained word embeddings from Google News[10] for all models with 300 word embedding size. Also for all models, we use Adam[11] as our optimizer to minimize the loss function, and the learning rate is 0.0001. We select the parameters of compared models based on their performance on the validation set. For DeepCoNN, we use 100 convolution kernels with window size 3. And NARRE uses the same size of convolution kernels, and uses 32 latent dimension in self-attention layer to compute the usefulness scores of reviews. And D-ATT uses 200 filters with window size 5 in local attention and 100 filters with window size [2,3,4] for global attention.

For our model, we set the CNN filter size is 300 with kernel size is 3, and hidden dimension of self-attention-layer is 300 with 1 head in word-level and sentence-level. We use dropout[22] to prevent overfitting. Dropout is used in the output of embedding layer, input of the feed-forward layer and the input of convolution layer.

We trained each model for at most 20 epochs with early stops if its performance not increase in 5 epochs. We independently repeat each experiment 5 times and use the root mean square error to quantify their perforamnce.

4.5 Experimental Results

The RMSE results of our model and other baseline models are shown in Table 2. From the results, we have several observations can be made:

Firstly, the CNN based models with attention network like NARRE and D-ATT can outperforms DeepCoNN which not use attention netowrk. The reason might because using attention network can distill important information from reviews, which can improve the final performance.

Secondly, we find our model is better than other baseline models in all dataset. It might because our model is consist of convolution layer with self attention layer, which can both extract the local and global information of reviews. Also, the hierarchical model architecture helps select information more robust and effective.

Table 2: RMSE scores of the different methods on different 5-core datasets.

Dataset	HCSA	NARRE	D-ATT	DeepCoNN
Digital Music	**0.8817**	0.8859	0.8872	0.8889
Toys and Games	**0.8883**	0.8902	0.8914	0.8950
Kindle Store	**0.7874**	0.7881	0.7905	0.7938
Office and Product	**0.8260**	0.8331	0.8335	0.8412

4.6 Case Study

We conducted several case studies to check if our approach can select informative words, sentences and reviews. First, the visualization of the attention networks from Figure 2 shows that our word-level attention network can effectively select and attend to important words, the denser the blue highlighted on the word, the higher attention score the word have. The orange boxes to the left of the reviews indicate the attention weights from the sentence-level network. Thus, the result validate the effectiveness of our approach.

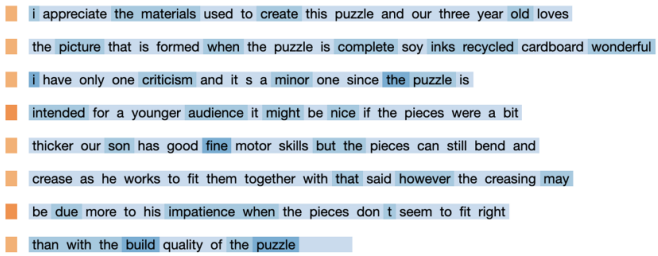

Figure 2: For a (user, item) pair, select a "most important review" (in orange), and inspect each word's attention score (in blue).

REFERENCES

[1] Lei Zheng, Vahid Noroozi, Philip S. Yu. Joint Deep Modeling of Users and Items Using Reviews for Recommendation In WSDM 2017. 425–434.

[2] Rose Catherine, William Cohen TransNets: Learning to Transform for Recommendation In RecSys '17:, Pages 288–296

[3] Chong Chen, Min Zhang, Yiqun Liu, Shaoping Ma. Neural Attentional Rating Regression with Review-level Explanations. In WWW '18. Pages 1583–1592.

[4] Dawei Cong, anyan Zhao, Bing Qin, Yu Han, Murray Zhang Hierarchical Attention based Neural Network for Explainable Recommendation In CMR '19, June 10–13, 2019, Ottawa, ON, Canada

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin Attention Is All You Need In 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

[6] Peter Shaw, Jakob Uszkoreit, Ashish Vaswani Self-Attention with Relative Position Representations In Proceedings of NAACL-HLT 2018, pages 464–468

[7] Sungyong Seo, Jing Huang, Hao Yang, Yan liu. Interpretable Convolutional Neural Networks with Dual Local and Global Attention for Review Rating Prediction In RecSys '17. 297–305.

[8] Chuhan Wu, Fangzhao Wu, Junxin Liu, and Yongfeng Huang Hierarchical User and Item Representation with Three-Tier Attention for Recommendation In Proceedings ofNAACL-HLT 2019, pages 1818–1826

[9] Steffen Rendle Factorization Machines In ICDM '10: Proceedings of the 2010 IEEE International Conference on Data Mining December 2010 Pages 995–1000

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, J. Dean Distributed representations of words and phrases and their compositionality In NIPS'13: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 December 2013 Pages 3111–3119

[11] D. Kingma, J. Ba. 2014. Adam: A Method for Stochastic Optimization In arxiv:1412.6980

[12] R. He, J. McAuley Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering WWW, 2016

[13] Yehuda Koren Factorization meets the neighborhood: a multifaceted collaborative filtering model In SIGKDD 426–434

[14] Yehuda Koren, Robert Bell, and Chris Volinsky Matrix factorization techniques for recommender systems Computer, 2009

[15] Julian McAuley and Jure Leskovec Hidden factors and hidden topics: understanding rating dimensions with review text In RecSys 165–172 2013

[16] Guang Ling, Michael R Lyu, and Irwin King Ratings meet reviews, a combined approach to recommend In RecSys 105–112 2014

[17] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma Explicit factor models for explainable recommendation based on phrase-level sentiment analysis In SIGIR 83–92 2014

[18] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen Trirank: Review-aware explainable recommendation by modeling aspects In CIKM 1661–1670 2015

[19] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma Rating-Boosted Latent Topics: Understanding Users and Items with Ratings and Reviews In IJCAI 2640–2646 2016

[20] Zhaochun Ren, Shangsong Liang, Piji Li, Shuaiqiang Wang, and Maarten de Rijke Social collaborative viewpoint regression with explainable recommendations In WSDM 485–494 2017

[21] Edward Loper , Steven Bird NLTK: The Natural Language Toolkit (2002) In ACL'2002

[22] Nitish Srivastava, imageGeoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan R Salakhutdinov Dropout: a simple way to prevent neural networks from overfitting In JMLR'14