Representation Learning of Users and Items for Review Rating Prediction Using Attention-based Convolutional Neural Network

Sungyong Seo* Jing Huang† Hao Yang† Yan Liu*

Abstract

It is common nowadays for e-commerce websites to encourage their users to rate shopping items and write review text. This review text information has been proven to be very useful in understanding user preferences and item properties, and thus enhances the capability of these websites to make personalized recommendations. In this paper, we propose to model user preferences and item properties using a convolutional neural network (CNN) with attention, motivated by the huge success of CNN for many natural language processing tasks. By using aggregated review text from users and items, we aim to build vector representations of user and item using attention-based CNNs. These vector representations are then used to predict rating values for a user on an item. We train these user and item networks jointly, which enables the interaction between users and items in a way similar to the matrix factorization technique. In addition, the visualization of the attention layer gives us insight on when words are selected by the models that highlight a user's preferences or an item's properties. We validate the proposed models on popular review datasets, *Yelp* and *Amazon*, and compare results with matrix factorization (MF), and hidden factor and topical (HFT) models. Our experiments show improvement over HFT, which proves the effectiveness of these representations learned from our networks on review text for rating prediction.

1 Introduction

Recommender systems are very common today with online shopping websites such as Amazon and Netflix. Ever since the famous Netflix Prize competition started ten years ago, collaborative filtering (CF) techniques have become successful and dominant approaches for recommender systems. Many of the CF approaches are based on matrix factorization (MF) [8], which decomposes into two latent feature matrices corresponding to latent features of users and items, and important weights of these latent factors. The dot product between a user and an item feature vector is used to predict the rating that the user would assign to the item.

Collaborative filtering has its own limitations and drawbacks, however. First, it is difficult for CF to give recommendations to users with few ratings or to recommend items with few ratings (the well-known cold start problem). Popular items tend to get more recommendations, while new items are left with no chance. In addition, it is also hard for CF techniques to recommend items to a user with unique preferences, because these rating numbers alone are inadequate to learn the user's preferences.

Another drawback of CF/MF techniques is their poor interpretability, making understanding of users preferences impossible. For example, it is difficult to associate latent features from MF with the understanding of users and items. We only know that a user might like an item due to a particular latent feature because there is a large positive (or negative) weight on that feature. But we have no clue what this feature means. In fact, it is possible that each feature corresponds to a combination of human interpretable aspects, for example, a restaraunt's cuisine style and its average price for a meal.

Using review text is one of the approaches to alleviate the above issues. Most shopping websites encourage their users to rate shopping items and write review text. Review text complements the rating numbers by providing rich information of items and implicit preferences of users. Review text explains why a user assigns such a rating to an item. A set of all reviews from this user allows us to derive this users' preferences; similarly a set of reviews on an item describes prominent properties of the item rated by many users. These preferences of a user and properties of an item can then be leveraged to alleviate the cold-start problem when the ratings are few. Recently using review text information has shown to improve rating prediction accuracy, especially for users and items with few ratings [11, 10].

Both the HFT model in [11] and the RMR model in [10] used interpretable latent topic models like LDA [3] on item review text: an item document consists of all

*Computer Science Department, University of Southern California, {sungyons, liu.cs}@usc.edu

†Visa Research, Visa Inc. {jinhuang, haoyang}@visa.com

reviews of an item; and the latent topic distribution is derived from these item documents. The topics discovered from item documents may not be suitable for users, however. We believe the aggregated review text of items cannot cover the same sentimental expressions in each individual user. For example, the same word "nice" might indicate different sentimental meaning from different users, while one user links "nice" to a rating of 4, another may link it to a rating of 5 using the same word. Therefore we should not just discard user-specific meanings of words and ignore their effect on the ratings by just assuming the topic distribution is the same as those in item review. Each user has their own preferences or tastes that can be discovered using all review text written by this user.

Therefore, in this paper we propose to model the user and the item separately: the aggregated review text from a user is used to build a user-specific model and the aggregated review text on an item is used to build an item-specific model. We use a convolutional neural network (CNN) to extract embedded representations of users and items from their corresponding review text. CNN has shown great success in many natural language processing tasks, such as text classification, sentiment analysis, neural language model and information retrieval ([15, 20, 14, 7]). For simplicity we use the same network structure for the user model and the item model as shows in Figure 1.

With word embedding input to the first CNN layer, many of the learned filters in the first CNN layer capture features quite similar to n-grams. After a few more CNN layers we would not know what features come out of the network, thus making feature interpretation difficult. To train an interpretable model we put an attention layer after the word embedding layer and before the CNN layer. This attention layer learns what to attend to from a local window of words by learning weights for these words, and therefore selects informative keywords from this local window before the words are passed to the CNN layer. Attention gives us the ability to interpret and visualize what the model is doing: for example, words with higher weights are highlighted in Table 4.

The main contributions of this paper are summarized as follows:

- Attention-based CNNs (Attn+CNN) are used to learn user and item representations from corresponding user and item reviews; these representations are used to predict ratings of a user on an item just like the MF technique.

- The attention layer is used before the CNN layers to select keywords from a local window that contributes to the rating. The visualization of the attention layer gives us insight on the words that are selected by the models that highlight a user's preferences or an item's properties.

- Our Attn+CNN model obtains a 6% relative improvement over the baseline MF and 2% relative improvement over HFT on the *Yelp* Challenge dataset 2013; and slightly better than those from MF/HFT models on 12 *Amazon* datasets.

The rest of this paper is organized as follows: Related works are first reviewed in Section 2. Section 3 describes in detail the components of our networks. Experimental setup and result analysis are presented in Section 4 and Section 5; and finally Section 6 concludes with future work.

2 Related Work

There are two lines of research related to our work: the first uses review text for recommendation, and the second is recent research on sentiment analysis using deep learning. We present brief reviews for these two research areas in the following.

Recent research on using review text for recommendation focused mostly on topic modeling for items from review text, such as hidden factors as topics (HFT) [11] and ratings meet reviews (RMR) [10]. HFT employed a LDA-like topic model on review text for items, and a matrix factorization (MF) model to fit the ratings. The two models were combined in an objective function that used the likelihood of the review text modeling by the topic distribution as a regularization term for the latent user-item parameters. This approach was shown to improve significantly over the baselines that use ratings or reviews alone, and it also works with items with only a few reviews. RMR [10] shared the same LDA-like topic modeling on item review text as HFT, except that RMR used Gaussian mixtures to model the rating instead of MF-like techniques.

TopicMF [2], as the name suggests, jointly modeled user ratings with MF and review text with non-negative matrix factorization (NMF) to derive topics from the review text. HFT learned the topics for each item, while topicMF learned the topics for each review. Their experimental results showed topicMF improved upon HFT. However, the exponential function used as the transformation function might fixate the relationship between latent factors in MF and the topic distribution with limited flexibility. As pointed out in [2], the authors hoped to model the user's preferences directly to their rating behavior, which is what our model is designed to do

Recently deep learning techniques have been ap-

plied to recommender systems with review content. In [1] a model was proposed that consists of a MF model for learning the latent factors and a recurrent neural network (RNN) for modeling the likelihood of the review using an item's latent factors. The RNN model is combined with MF via a regularization term, just like the approach used in HFT. In [18] a hierarchical Bayesian model called collaborative deep learning (CDL) was proposed to take advantage of review content information. CDL used stacked denoising autoencoders (SDAE) to learn feature representations for the items. This network together with collaborative filtering with a rating matrix were jointly trained. However, the content information was only extracted from bag-of-words representations, which did not take into account word orders and context that are important for extracting semantic meanings. All the above work failed to link user's preferences and sentiment in their review text to their ratings.

Another line of research closely related to recommendation on review text is sentiment analysis and text classification. Recently there are many works on sentiment analysis and text classification using deep learning techniques that acheived impressive results as shown in [6, 17, 14, 16, 9, 20, 5, 19]. Various deep neural network configurations were used for these tasks: CNN [6, 17, 14, 20, 5], recursive neural tensor network, recurrent neural network (RNN) [16, 9] and LSTM [19]. Most networks used word embeddings [17, 14, 16, 9, 19] as the input layer, while [6, 20, 5] used character embeddings, especially [5] used a very deep (29 layers) CNN on character embeddings and showed impressive results on text classification, including the Amazon reviews.

The idea of attention in neural networks is loosely based on the visual attention mechanism found in humans. Human visual attention is able to focus on a certain region of an image with "high resolution" while perceiving the surrounding image in "low resolution", and adjusting the focal point over time. A big advantage of attention is that it gives us the ability to interpret and visualize what the model is doing. While attention-based deep learning models rely on RNNs and encoder-decoders for tasks such as machine translation and image caption generation, the attention module in our model is designed to work with CNN. It allows us to infer from training the informative keywords that link directly to the user's rating. We put the attention layer before the CNN so that we could visualize those keywords at the end of model training and help us interpret the results (as shown in Table 4).

3 Proposed Model

In this section, we describe our Attn+CNN model for learning latent representations from review texts. Figure 1 shows the overall architecture. We use the same network structure for the user and item network. So, we describe the user network in detail: the left part is the attention-based module that learns representations of informative keywords. The right part is the CNN module that learns representations from the original review word sequences. These two representations are then combined through a CNN layer and a fully-connected layer as the final user/item representations for rating predictions.

We describe each of these modules in Figure 1. We denote scalars with italic lower-cases (x, y), vectors with bold lower-cases $(\mathbf{x}, \mathbf{z})$, and matrices with bold upper-cases $(\mathbf{X}, \mathbf{W})$.

Embedding Layer We use word embedding for input review document D_u: a set of review from user u. An embedding layer can be simply regarded as a look-up operation that reads an one-hot vector, $\mathbf{e}_t \in \mathbb{R}^{|\mathcal{V}|}$, for a word as an input, and map it to a dense vector, $\mathbf{x}_t = (x_1, x_2, \cdots, x_d)$ and $\mathbf{x}_t \in \mathbb{R}^d$. The weight of the embedding layer is $\mathbf{W}_e \in \mathbb{R}^{d \times |\mathcal{V}|}$:

$$\mathbf{x}_t = \mathbf{W}_e \mathbf{e}_t.$$

and $\mathcal{V}$ is a set of words. $|\mathcal{V}|$ is the size of the vocabulary: the most frequent 20,000 words.

Attention Module on the left An attention mechanism is motivated by human visual attention. When we read text or see images, we focus on certain part of the input to understand or recognize them more efficiently. In our model the attention module is used to learn which words are more informative in a given window.

Let D_u be represented as a length T word embeddings $(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T)$. Then, we apply the attention through sliding windows to this sequence. Let $\mathbf{x}_i$ be the center word and w be the window width. We compute weighting scores for each word in the window with a $\mathbf{W}^1_{att} \in \mathbb{R}^{w \times d}$ parameter matrix and a bias b^1_{att} as follows:

$$\mathbf{X}_{att,i} = (\mathbf{x}_{i+\frac{-w+1}{2}}, \mathbf{x}_{i+\frac{-w+3}{2}}, \cdots, \mathbf{x}_i, \cdots, \mathbf{x}_{i+\frac{w-1}{2}})^\top,$$
$$s_i = g(\mathbf{X}_{att,i} * \mathbf{W}^1_{att} + b^1_{att}).$$

s_i is the score showing how much the i-th word is informative. The score can be directly used as a weight for i-th word embedding or we can apply a threshold to remove "trivial" words and only consider informative

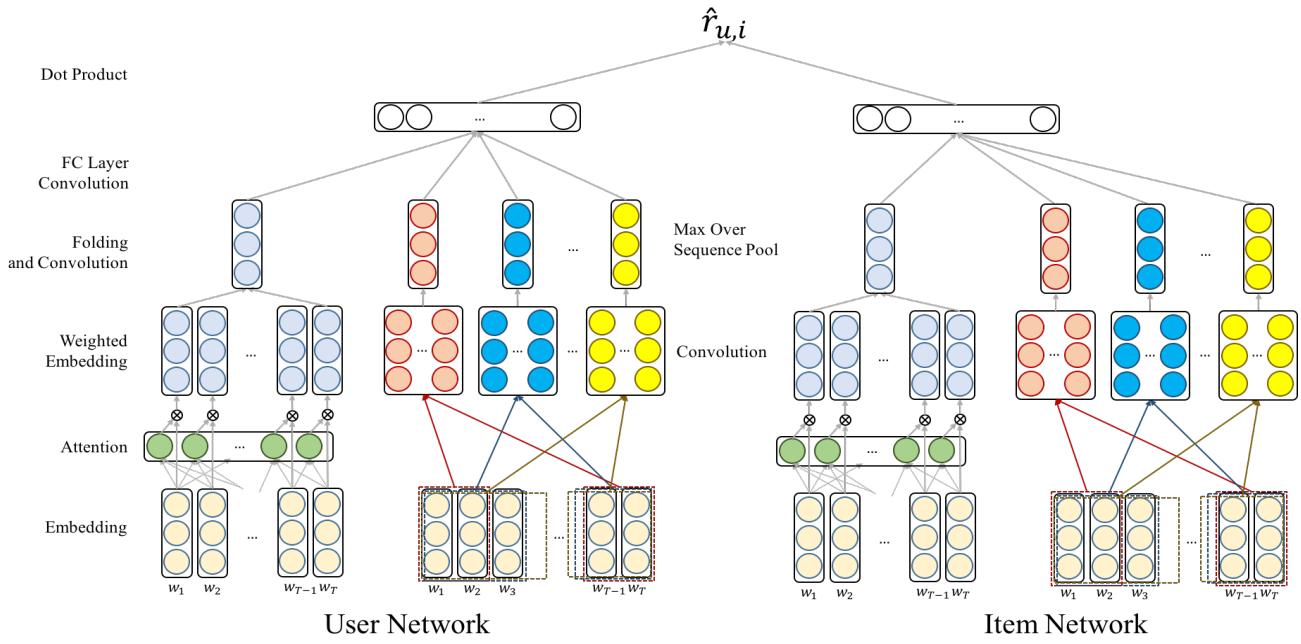

Figure 1: Attention-based CNNs to extract latent representations of users or items. A user document D_u and an item document D_i are fed into (Left) the user network and (Right) the item network respectively.

attention words. In this work, we use scores as weights. We use *sigmoid* for the activation function g.

$$\hat{\mathbf{x}}_t = s_t \mathbf{x}_t.$$

$\hat{\mathbf{x}}_t$ where $t \in [1, T]$ is a weighted sequence of word embeddings.

The folding layer summarizes attention words and outputs the *attention representation* of given text. The attention words $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \cdots, \hat{\mathbf{x}}_T)$ are folded through the sum operation along the sequential order, $\mathbf{y} = \sum_t \hat{\mathbf{x}}_t$. Finally, the attention representation is obtained through a convolution operation with a matrix $\mathbf{W}^2_{att} \in \mathbb{R}^{d \times n_{att}}$ and a bias $\mathbf{b}^2_{att} \in \mathbb{R}^{n_{att}}$:

$$\mathbf{z}_{att}(i) = g(\mathbf{y} * \mathbf{W}^2_{att}(:, i) + \mathbf{b}^2_{att}(i)), \quad i \in [1, n_{att}].$$

n_{att} is the number of filters and g is a *tanh* function.

Convolutional module on the right The word sequence (with its original order) from D_u is input to the CNN module to learn a global semantic representation for u. For a convolutional layer, we set the length of a filter as w_f, which means the filter operates on w_f words. If the number of filters is n_{conv}, the convolution filters $\mathbf{W}_{conv} \in \mathbb{R}^{w_f \times d \times n_{conv}}$ are applied to a sequence of w_f word embeddings, $\mathbf{X}_{conv,i} \in \mathbb{R}^{w_f \times d}$, and output features $\mathbf{Z} \in \mathbb{R}^{(T - w_f + 1) \times n_{conv}}$:

$$\mathbf{X}_{conv,i} = (\mathbf{x}_i, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_{i+w_f-1})^\top,$$
$$\mathbf{Z}(i, j) = g(\mathbf{X}_{conv,i} * \mathbf{W}_{conv}(:, :, j) + \mathbf{b}_{conv}(j)),$$
$$i \in [1, T - w_f + 1], \quad j \in [1, n_{conv}].$$

g is a nonlinear activation function and $\mathbf{b}_{conv}$ is a bias vector. In the pooling layer, a max pooling is applied over the sequence: $\mathbf{z}_{conv}(j) = \textsc{Max}(\mathbf{Z}(:, j))$. We can obtain as many $\mathbf{z}_{conv}$ as different filter length w_f. In this work $\#n_{conv}$ different filter lengths are used.

Final layers The output of the attention module and the CNN module are concatenated, and run through an additional convolutional layer $\mathbf{W}_{out}$ and FC layer $\mathbf{W}_{FC}$:

$$\mathbf{z}_o = \mathbf{z}_{att} \oplus \mathbf{z}^1_{conv} \oplus \cdots \oplus \mathbf{z}^{\#n_{conv}},$$
$$\mathbf{z}_{out}(i) = g(\mathbf{z}_o * \mathbf{W}_{out}(:, i) + \mathbf{b}_{out}(i)), \quad i \in [1, n_{out}],$$
$$\gamma_u = \mathbf{W}_{FC} \cdot \mathbf{z}_{out}.$$

$\oplus$ is a concatenation operator and n_{out} is the number of filters applied to $\mathbf{z}_o$.

Training of the network Two different channels (attention module and CNN) are used to learn two different latent representations, *attention representation* and *semantic representation*, and are merged into one

attention-based semantic representation. Let γ_u be the attention-based semantic vector from the users' network and γ_i be the corresponding vector from the items' network. As in CF/MF techniques, the latent representations (γ_u, γ_i) are mapped into same vector space $(\mathbb{R}^K)$ and ratings can be estimated by the inner product.

$$\hat{r}_{u,i} = \gamma_u^\top \gamma_i$$

The estimation can be considered as a regression problem and all parameters in two networks (user network and item network) are trained jointly through the back-propagation technique. Mean Squared Error (MSE) is used as an error function. In training time, D_u and D_i are fed into two networks respectively, and $\hat{r}_{u,i}$ is the target value. At test time, a pair of a user and an item (u, i) along with their corresponding D_u and D_i are fed through the user/item networks, and the inner product of γ_u, γ_i is the estimated rating $r_{u,i}$.

4 Experimental Setup

We evaluate our proposed models using open datasets from *Yelp* and *Amazon*. In this section, we describe these datasets as well as our experimental setup.

4.1 Datasets We used two publicly available datasets that provide user reviews and rating information. The first dataset is from *Yelp Business Rating Prediction Challenge* 2013[1], which includes reviews on restaurants in Phoenix, AZ metropolitan area. The second dataset is *Amazon Product Data*[2], which contains millions of product reviews and metadata from *Amazon*. This dataset has been investigated by many researchers [11, 12, 13]. In this paper, we focused on the *5-core* subset, with at least 5 reviews for each user or item. There are 24 product categories and 12 categories are used in this work. The key characteristics of these two datasets are summarized in Table 1.

Dataset	*Yelp*	*Amazon*
# of Users	45,980	268,788
# of Items	11,537	187,203
# of Reviews	229,900	3,165,314
Avg. # of Words per review	130	128

Table 1: Statistics of Dataset

We randomly divided each dataset into training, validation and test sets (80%, 10%, 10% respectively).

[1] https://www.kaggle.com/c/yelp-recsys-2013
[2] http://jmcauley.ucsd.edu/data/amazon/

4.2 Data preprocessing The first step in our data processing pipeline is to concatenate all reviews from the same user, say user u, into a document D_u. Similarly, we concatenate all reviews on the same item, say item i, into a document D_i. As expected, the length of these concatenated review documents follows a long-tailed distribution and we set the length of each document to conserve at least 70% of the words. The resulting documents are fed into the embedding layer as described in Section 3.

4.3 Baselines We implemented several baselines as comparisons to our proposed model. The first one is Matrix Factorization (MF)[3][8] that characterizes both users and items by vectors of factors inferred from item rating patterns. The second baseline is Hidden Factors as Topics (HFT) [11]. We set the number of hidden topics K to 5 which is reported in [11]. In addition, results from CNN-only are also compared to show the effectiveness of the attention module. Finally, we show a naive method, *Offset*, which simply uses the average rating (μ) as the prediction. This baseline shows the upper bounds of rating estimates for each dataset.

4.4 Parameter Setting We use pre-trained word embedding *fastText* [4] based on the skip-gram model, where each word is represented as a bag of n characters. The dimension of embedding is $d = 100$. In the attention module, we use $w = 5$ window size for the local attention layer with *sigmoid* and $n_{att} = 400$ in $\mathbf{W}_{att}^2$. In the CNN module, we use four different filter lengths $w_f \in [2, 3, 4, 5]$ and $n_{conv} = 100$ for each $\mathbf{W}_{conv}$. To combine attention representation and semantic representation, we set $n_{out} = 256$ for $\mathbf{W}_{out}$ and finally, the number of hidden factors is $K = 250$ in $\mathbf{W}_{FC}$ with 25% dropout probability. ReLU is used as an activation function for the last convolutional layer and fully-connected layers.

Our implementation uses Theano to train and optimize the neural networks, and we use GPU cards (Nvidia GeForce GTX TITAN X) to speed up the model training process.

5 Results and Discussion

5.1 Rating Estimation The Mean Squared Error (MSE) of rating estimation is shown in Table 2 for Attn+CNN as well as for different baselines. We can see that Attn+CNN outperforms other models on the *Yelp* dataset and on the *Amazon* dataset on average. This clearly confirms the effectiveness of our proposed method.

[3] We use MyMediaLite package. http://www.mymedialite.net

Dataset	Offset	MF	HFT	CNN-only	Attn+CNN
Yelp	1.484	1.295	1.243	1.269	1.212*
Amazon	1.102	0.935	0.934	0.935	0.928*

Table 2: MSE of rating estimation for different models (the best results are starred)

Dataset	Offset	MF	HFT	CNN-only	Attn+CNN
Amazon instant video	1.273	0.946	0.925*	0.944	0.936
Automotive	0.939	0.875*	0.975	0.890	0.881
Baby	1.315	1.167	1.151*	1.178	1.176
Cds and vinyl	1.150	0.885	0.861*	0.875	0.866
Grocery and gourmet food	1.202	1.017	1.004	1.010	1.004*
Health and personal care	1.233	1.068	1.053*	1.060	1.054
Kindle store	0.916	0.623	0.609*	0.621	0.617
Musical instruments	0.742	0.689*	0.726	0.710	0.703
Office products	0.867	0.724*	0.726	0.728	0.726
Patio lawn and garden	1.153	1.029	1.025	1.017	0.999*
Pet supplies	1.384	1.253	1.232*	1.241	1.236
Tools and home improvement	1.053	0.941	0.927*	0.945	0.938

Table 3: Breakdown of *Amazon* dataset evaluation by product category

We further break down the results for the *Amazon* dataset by product category, as shown in Table 3. While review-based models (Attn+CNN and HFT) are generally better than MF, there are several categories (e.g., *Automotive, Musical instruments*, and *Office products*) where MF performs the best and HFT performs the worst.

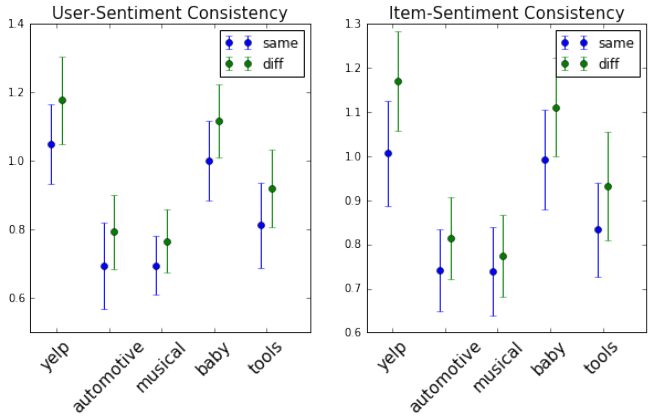

Figure 2: User-sentiment consistency and Item-sentiment consistency. Blue dots are for two reviews by same user or item, and green dots are for different users or items.

To understand these results more closely, we examine the consistency between ratings and texts, in particular, user-sentiment consistency, item-sentiment consistency, user-text consistency, and item-text consistency, as suggested in [17].

Figure 2 shows the sentiment consistency in different datasets or categories, measured by the average of absolute rating differences between two reviews. In other words, it tells to what extent the ratings by the same user, or on the same item, are consistent. This finding explains why MF works better in some categories. MF estimates ratings based on user ratings only. Thus, the more consistent these ratings are, the better performance MF can achieve. Indeed, it performs the best on *Automotive* and *Musical Instruments* categories, which have higher consistencies than the other categories.

On the other hand, text consistency captures the textual similarity between reviews written by the same user or on the same item, measured by cosine similarity between the bag-of-words of two reviews. Figure 3 shows the text consistency in different datasets or categories. We can see that Baby and Tools categories have higher degrees of text consistency than the others. This also explains why HFT works better in these categories, because it is more effective to extract topic information from these reviews and infer the user preferences.

While MF and HFT each has its own strengths, as discussed above, our Attn+CNN model can achieve

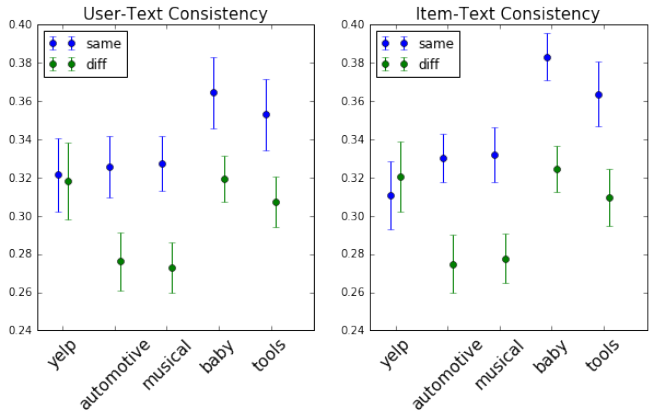

Figure 3: User-text consistency and Item-text consistency. Blue dots are for two reviews by same user or item, and green dots are for different users or items.

the best of both, which can best be seen from the results on *Yelp* dataset. First, the ratings in this dataset are inconsistent. This explains why MF leads to large MSE on this dataset. Second, the review text is also inconsistent. In other words, each user is likely to use different words for different items, and each item has reviews written by a diverse set of words. As a result, it is difficult to infer proper topics for such heterogeneous corpus. In contrast to both MF and HFT, our model is more robust to model users' preferences or items' properties because word embedding captures implicit meaning of each word, and the attention module tells which words are more meaningful for the rating estimation.

5.2 Visualize Attention Keywords To confirm our findings on the attention module, we highlight in Table 4 words that are considered as informative by the attention module. We select two review examples from *Amazon (patio lawn and garden)* and *Yelp*, and the highlighted words are obtained from high attention scores in the item or user network. We can see a few interesting patterns from these results. For example, adjective words that describes properties of the item are likely highlighted in the item's review. For example, *Yelp* reviews clearly show properties(or characteristics) of a certain restaurant by highlighting *terrible, average*, and *disrespectful*. More personalized words such as *huge fan* and *enjoy* and informative adjectives are also highlighted in a user's review. This confirms that the attention module can indeed identify the most informative words in the reviews.

Table 5 shows the same text but highlighted differently by the user network and the item network. The user network and item network are trained with different sets of documents, the joint training of objective function decides which part of the text is important for the final score. Therefore, the two networks choose different attention words as expected.

category : patio lawn and garden (item)
This hose is an excellent garden tool . I bought 2 of them one 25ft for my front garden and 50ft for my rear garden. 3 4 months and still in excellent condition . I probably should have bought something a bit more flexible and less rugged since I constantly coil uncoil it for washing cars but that's my fault not a product fault .
category : Yelp (item)
Not sure how to start this review. Some parts of the dinner were terrible while the other parts were just average , just nothing special . I decided to take my mother out to dinner who was visiting all the way from Dubai based on the great things I've heard about Mastro's Ocean Club. The night started off in a terrible way when our waiter was extremely disrespectful to us. We asked for a few minutes when our waiter asked us if we wanted still, sparkling or tap water and his reply in a really condescending tone was" right now I'm asking you about the water".
category : Yelp (user)
Payton & Shantal do a great job after such a quick transition. I'm a huge fan of the farm-to-table philosophy and the frequently changing menus. It's a happy place , especially if you sit at the counter and enjoy the show .

Table 4: Visualization of review text with highlights. Colored words are considered as informative words, and green words have higher attention scores than those of yellow words.

6 Conclusion and Future Works

We have presented an attention-based CNN model that combines review text and ratings for product recommendation. It learns the vector representation of users and items from the aggregated reviews, and enables the interaction between user and item models in a way similar to matrix factorization. By leveraging the best of collaborative filtering and topic-based approaches, our model is inherently more robust to noise and inconsistency in the review and rating data, which is validated by our experiments over both *Yelp* and *Amazon* datasets. We believe this work offers a new avenue to apply representation learning in the context of recommendation systems. One future direction we are explor-

category : Yelp (user)
A disappointing meal and a very disappointing service. I won't be coming back anytime soon. If I was the manager I would demote the waiter and promote the busboy to a waiter as he was great tonight and he was the only reason I gave a 20% tip as it is unfair for him to suffer because of the waiters.
category : Yelp (item)
A disappointing meal and a very disappointing service. I won't be coming back anytime soon. If I was the manager I would demote the waiter and promote the busboy to a waiter as he was great tonight and he was the only reason I gave a 20% tip as it is unfair for him to suffer because of the waiters.

Table 5: Attention words of the same review. Each network highlights different words.

ing is to use sequence learning, e.g., Long Short-Term Memory (LSTM) network, to handle long-range dependency in the review texts.

References

[1] Amjad Almahairi, Kyle Kastner, Kyunghyun Cho, and Aaron Courville. Learning distributed representations from reviews for collaborative filtering. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 147–154. ACM, 2015.

[2] Yang Bao, Hui Fang, and Jie Zhang. Topicmf: Simultaneously exploiting ratings and reviews for recommendation. In *AAAI*, pages 2–8, 2014.

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

[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[5] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for natural language processing. *arXiv preprint arXiv:1606.01781*, 2016.

[6] Cícero Nogueira dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78, 2014.

[7] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. *arXiv preprint arXiv:1508.06615*, 2015.

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

[9] Jiwei Li, Minh-Thang Luong, Dan Jurafsky, and Eudard Hovy. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185*, 2015.

[10] Guang Ling, Michael R Lyu, and Irwin King. Ratings meet reviews, a combined approach to recommend. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 105–112. ACM, 2014.

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

[12] Julian McAuley, Rahul Pandey, and Jure Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2015.

[13] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52. ACM, 2015.

[14] Aliaksei Severyn and Alessandro Moschitti. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 959–962. ACM, 2015.

[15] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 101–110. ACM, 2014.

[16] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[17] Duyu Tang, Bing Qin, and Ting Liu. Learning semantic representations of users and products for document level sentiment classification. In *Proc. ACL*, 2015.

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

[19] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

[20] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657, 2015.