Asymmetrical Hierarchical Networks with Attentive Interactions for Interpretable Review-Based Recommendation

Xin Dong, Jingchao Ni, Wei Cheng, Zhengzhang Chen, Bo Zong, Dongjin Song, Yanchi Liu, Haifeng Chen, and Gerard de Melo

1 Rutgers University, 2 NEC Laboratories America

1 {xd48@rutgers.edu, gdm@demelo.org}

2 {jni, weicheng, zchen, bzong, dsong, yanchi, haifeng}@nec-labs.com

Abstract

Recently, recommender systems have been able to emit substantially improved recommendations by leveraging user-provided reviews. Existing methods typically merge all reviews of a given user or item into a long document, and then process user and item documents in the same manner. In practice, however, these two sets of reviews are notably different: users’ reviews reflect a variety of items that they have bought and are hence very heterogeneous in their topics, while an item’s reviews pertain only to that single item and are thus topically homogeneous. In this work, we develop a novel neural network model that properly accounts for this important difference by means of asymmetric attentive modules. The user module learns to attend to only those signals that are relevant with respect to the target item, whereas the item module learns to extract the most salient contents with regard to properties of the item. Our multi-hierarchical paradigm accounts for the fact that neither are all reviews equally useful, nor are all sentences within each review equally pertinent. Extensive experimental results on a variety of real datasets demonstrate the effectiveness of our method.

1 Introduction

The rapid shift from traditional retail and services to online transactions has brought forth a large volume of review data in areas such as e-commerce, dining, tourism, among many others. While such reviews are routinely consulted directly by consumers and affect their decision-making, recent work has shown that they can also be exploited by intelligent algorithms. The detailed semantic cues that they harbor not only reveal different aspects (e.g., quality, material, color, etc.) of an item, but also reflect the sentiment of users towards these aspects. Such fine-grained signals are extremely valuable to a recommender system and significantly complement the sparse rating and click-through data, based on which many traditional collaborative filtering methods (Koren, Bell, and Volinsky 2009) have been developed. Thus, there has been a series of studies seeking to harness the potential of reviews in improving the recommendation quality.

Figure 1: An example of reviews by a user and for an item.

User u’s reviews

I am happy that this is a good lab vitamin and not sourced from genetically modified corn. I take two in the morning and after every workout. I do not trust that they are capsules but I just lick my hand down so they go to the back of my throat making it easy to swallow. Great quality and great price for quality.

Item v’s reviews

They smell strongly of olive oil in the bottle but really have no taste and are easy to swallow. I am not a fan of over packaging and these are definitely over packaged. They serve the purpose, since my doctor said I needed to take vitamin D.

This has helped burnout with the inflammation in my left knee joint. It’s only been a few weeks so I will see if it works even better after taking this second bottle. So far they’re working, are easy to read and go down after taste is gone.

First of all it does not smell of lavender at all. Not that it’s a bad scent or anything but it last for just one night. It’s not noticeable even when you stick your nose up to it. I think face needs to do a little more to make it a worthwhile product.

They solved my problem very well and I have ordered from them at least 5 times now. The package is secure and the ingredients are carefully considered, and they still manage to keep their prices low. This is a small, used to realize gel, and I recommend it if you are considering buying a d supplement.

I prefer the capsule form of vitamin d, especially when they are in 1000 iu increments. These soft gels are great.

User u rated a 5.0 score on item v after purchasing it.

These studies have shown that leveraging reviews can indeed boost the recommendation effectiveness quite remarkably. Typically, they associate users with the respective sets of reviews they have written, while associating each item with the set of all reviews that have been written for it. To predict the rating for an unseen user–item pair, in a first step, the embeddings of that user and item are inferred from the respective sets of reviews via a neural network. Then, the two embeddings are matched to predict a numeric rating between them. For example, DeepCoNN (Zheng, Noroozi, and Yu 2017) relies on convolutional neural networks to learn user (item) embeddings, and on a factorization machine (Rendle 2010) to predict ratings. D-ATT (Seo et al. 2017) uses dual-attention based networks to learn embeddings, and a simple dot product to predict ratings.

Despite the encouraging progress, existing methods all regard the set of reviews by a user and the set of reviews for an item as the same type of documents, and invoke the same model (or even a shared model) to process them in parallel. In reality, however, the set of reviews for a user is fundamentally different from the set of reviews for an item. In particular, reviews for users correspond to a diverse set of items that they have rated, resulting in notably heterogeneous textual contents with a variety of topics for different items. In contrast, each item’s reviews are only about itself, and the

\textcopyright 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
We identify the asymmetric attention problem for review-based recommendation, which is important but neglected by existing approaches.

2. We propose AHN, a novel deep learning architecture that not only captures both of the asymmetric and hierarchical characteristics of the review data, while also enabling interpretability of the results.

3. We conduct experiments on 10 real datasets. The results demonstrate that AHN consistently outperforms the state-of-the-art methods by a large margin, while providing good interpretations of the predictions.

2 Related Work

Exploiting reviews has proven considerably useful in recent work on recommendation. Many methods primarily focus on topic modeling based on the review texts. For example, HFT (McAuley and Leskovec 2013) employs LDA to discover the latent aspects of users and items from reviews. RMR (Ling, Lyu, and King 2014) extracts topics from reviews to enhance the user and item embeddings obtained by factorizing the rating matrix. TopicMF (Bao, Fang, and Zhang 2014) jointly factorizes a rating matrix and bag-of-words representations of reviews to infer user and item embeddings. Despite the improvements achieved, these methods only focus on topical cues in reviews, but neglect the rich semantic contents. Moreover, they typically represent reviews as bag-of-words, and thus remain oblivious of the order and contexts of words and sentences in reviews, which are essential for modeling the characteristics of users and items (Zheng, Noroozi, and Yu 2017).

Inspired by the astonishing advances of recent deep NLP techniques in various applications (Santos et al. 2016; Wang et al. 2018; Peters et al. 2018; Dong and De Melo 2018; Devlin et al. 2018; Yang et al. 2019), there has been increasing interest in studying deep learning models. DeepCoNN (Zheng, Noroozi, and Yu 2017) employs CNNs as an automatic feature extractor to encode each user and item into a low-dimensional vector by assessing the relevant set of historical reviews. TransNet (Catherine and Cohen 2017) extends DeepCoNN with a multi-task learning scheme to regularize the user and item embeddings towards the target review. These methods, however, lack interpretability (Xian et al. 2019) in their results.

To better understand the predictions, several attention-based methods have been developed. D-ATT (Seo et al. 2017) incorporates two kinds of attention mechanisms on the words of reviews to find informative words. NARRE (Chen et al. 2018) invokes review-level attention weights to aggregate review embeddings to form user (item) embeddings. HUITA (Wu et al. 2019) is equipped with a symmetric hierarchical structure, where, at each level (e.g., word level), a regular attention mechanism is employed to infer the representation of the subsequent level (e.g., sentence level). MPCN (Jie, Luu, and Hui 2018) models the interactions between a user’s reviews and an item’s reviews via co-attention based pointers that are learned with the Gumbel-Softmax trick (Jang, Gu, and Poole 2016). However, all these methods just learn user and item embeddings in parallel and fail to consider the important differences between the two. As discussed before, this leads to suboptimal predictions.

Unlike the aforementioned methods, our method learns several hierarchical aggregators to infer user (item) embeddings. The aggregators are asymmetric to flexibly pay varying levels of attention to a user’s (item’s) reviews, so as to enhance the prediction accuracy and model interpretability.
Sentence Encoding

The sentence encoding layer (omitted in Fig. 2) aims to transform each sentence (in each review) from a sequence of discrete word tokens to a continuous vector embedding. We use a word embedding model to lay the foundation of this layer. Suppose the sentence \( s \) has \( l \) words. By employing a word embedding matrix \( \mathbf{E} \in \mathbb{R}^{d \times |V|} \), \( s \) can be represented by a sequence \( \{e_1, ..., e_l\} \), where \( e_i \) is the embedding of the \( i \)-th word in \( s \), \( d \) is the dimensionality of the word embedding, and \( |V| \) is the whole vocabulary of words. The matrix \( \mathbf{E} \) can be initialized using word embeddings such as word2vec (Mikolov et al. 2013) and GloVe (Pennington, Socher, and Manning 2014), which are widely used in NLP. To refine the word embeddings, \( \mathbf{E} \) is fine-tuned during model training.

To learn an embedding for \( s \), we employ a bi-directional LSTM (Peters et al. 2018) on its constituent word embeddings, and apply max-pooling on the hidden states to preserve the most informative information. That is

\[
s = \max(\{e_1, ..., e_l\}),
\]

where \( s \) is the embedding of \( s \) and

\[
\tilde{e}_i = \text{BiLSTM}(\hat{e}_{i-1}, e_i) \quad (1 \leq i \leq l),
\]

where \( \hat{e}_0 \) is initialized by an all-zero vector \( \mathbf{0} \).

Suppose a review has \( k \) sentences. We can then represent this review by a sequence \( \{s_1, ..., s_k\} \), where \( s_i \) is the embedding of the \( i \)-th sentence in the review, as inferred by Eq. (1). However, using Eq. (1), each \( s_i \) only encodes its own semantic meaning, but remains oblivious of any contextual cues from its surrounding sentences in the same review. To further refine the sentence encoding, we introduce a context-encoding layer by employing another bi-directional LSTM on top of the previous layer to model the temporal interactions between sentences, i.e.,

\[
\mathbf{s}_i = \text{BiLSTM}(\tilde{s}_{i-1}, s_i) \quad (1 \leq i \leq k),
\]

where \( \mathbf{s}_i \) is the final embedding of the \( i \)-th sentence in the review and \( \mathbf{s}_0 \) is initialized as \( \mathbf{0} \).

Sentence-Level Aggregation

Next, we develop sentence-level aggregators to embed each review into a compact vector from its constituent sentences. As discussed before, an ideal method should learn review embeddings in an asymmetric style. Thus, we design AHN to learn different attentive aggregators for users and items, respectively, as highlighted in Fig. 2.

Sentence Aggregator for Items. Given an item, we are interested in sentences that contain other users’ sentiments on different aspects of the item, which are the key factors to determine its overall rating. To build an informative embedding for each review upon such sentences, we use a sentence-level attention network to aggregate the sentence embeddings \( \{\mathbf{s}_1, ..., \mathbf{s}_k\} \) as follows, where the superscript \( v \) is used to distinguish an item’s notation from a user’s notation.

\[
r^v = \sum_{i=1}^{k} \alpha^v_i \mathbf{s}_i^v,
\]

Here, \( \sum_{i=1}^{k} \alpha^v_i = 1 \) and \( \alpha^v_i \) is the attention weight assigned to sentence \( \mathbf{s}_i^v \). It quantifies the informativeness of sentence \( \mathbf{s}_i^v \) with respect to \( v \)’s overall rating, compared to other sentences. The weights \( \alpha^v_i \) are computed by our attentive module, taking the sentence embeddings as the input, as

\[
\alpha^v_i = \frac{\exp (v^v_i (\tanh (\mathbf{W}_s^v \mathbf{s}_i^v) \odot \sigma(\mathbf{W}_h^v \mathbf{s}_i^v))))}{\sum_{j=1}^{k} \exp (v^v_i (\tanh (\mathbf{W}_s^v \mathbf{s}_j^v) \odot \sigma(\mathbf{W}_h^v \mathbf{s}_j^v))))}
\]

Here, \( v^v_i \in \mathbb{R}^{h \times 1} \), \( \mathbf{W}_s^v \in \mathbb{R}^{d \times h} \), and \( \mathbf{W}_h^v \in \mathbb{R}^{h \times h} \) are parameters, \( \odot \) is the element-wise product, and \( \sigma(\cdot) \) is the sigmoid function. As suggested by Ilse, Tomczak, and Welling (2018), the approximate linearity of tanh(\cdot) in [-1, 1] could limit the expressiveness of the model, which can be alleviated by introducing a non-linear gating mechanism. Thus, in Eq. (5), a gate function \( \sigma(\mathbf{W}_h^v \mathbf{s}_i^v) \) is incorporated, which is indeed found effective in our experiments.

Sentence Aggregator for Users. Next, we develop an interaction-based sentence aggregator for users. Given a user–item pair, we aim to select a homogeneous subset of sentences from each of the user’s reviews such that the selected sentences are relevant to the item to be recommended, i.e., the target item. In the following, we introduce a co-attentive network that uses the target item’s sentences to guide the search of user’s sentences.

Figure 2: The overall architecture of AHN.
Here, $M_r \in \mathbb{R}^{d_r \times d_r}$ is a learnable parameter, $\phi(\cdot)$ is an activation function such as ReLU, and $f(\cdot)$ is a mapping function such as a multi-layer perceptron (MLP). If $f(\cdot)$ is the identity mapping, Eq. (6) becomes a bilinear mapping. Here, the $(p, q)$-th entry of $G_i$ represents the affinity between the $p$-th sentence of $R_i^v$ and the $q$-th sentence of $[R_{r1}^u; ...; R_{rm}^u]$.

To measure how relevant the $p$-th sentence of the user’s review $R_i^u$ is to the target item, we use the maximum value in the $p$-th row of $G_i$. The intuition is that, if a user’s sentence (i.e., a row of $G_i$) has a large affinity to at least one sentence of the target item (i.e., a column of $G_i$) – in other words, the maximal affinity of this row is large – then this user’s sentence is relevant to the target item.

However, not all sentences of the target item are useful for searching relevant sentences from the user. For instance, in Fig. 1 the first sentence of the item’s review 2, “I received it three days ago,” conveys little information about the target item, and hence cannot aid in identifying relevant sentences from the user, and indeed may introduce noise into the affinity matrix. To solve this problem, recall that $\alpha_v^i$ in Eq. (5) represents how informative an item’s sentence is. Thus, we concatenate $\alpha_v^i$’s of all sentences of the target item to form $\alpha_v^i \in \mathbb{R}^{1 \times nk}$. Subsequently, we compute an element-wise product between each row of $G_i$ and the vector $\alpha_v$, i.e., $G_i \odot_{row} \alpha_v$. In this manner, the $(p, q)$-th entry, $(G_i \odot_{row} \alpha_v)_{pq}$, is high only if the $p$-th sentence of the user is similar to the $q$-th sentence of the target item and the $q$-th sentence of the target item is non-trivial.

By summarizing the above insights, we learn attention weights for the sentences in $R_i^u$ for each $i \in [1, n]$ by

$$\alpha_v^i = \text{softmax}(\max_{\text{row}}(G_i \odot_{row} \alpha_v)), \quad (7)$$

where $\max_{\text{row}}$ refers to row-wise max-pooling for obtaining the maximum affinity. Intuitively, $(\alpha_v^i)_j$ is large if the $j$-th sentence in the $i$-th review of the user describes some aspects of some item that is highly similar to the target item. This serves our purpose for selecting a homogeneous subset of sentences from the user.

Next, we use $\alpha_v^i$ to aggregate the sentences in $R_i^u$ to infer an embedding of the $i$-th review for the user:

$$r_i^u = \sum_{j=1}^{k} (\alpha_v^i)_j (R_i^u)_{sj}, \quad (8)$$

where $(R_i^u)_{sj}$ is the $j$-th column of $R_i^u$. Recall that $R_i^u = [s_{i1}^u; ...; s_{im}^u]$, where each column of $R_i^u$ is a sentence embedding. Note that our method iterates over $i$ for $i \in [1, n]$ to calculate all review embeddings $r_i^1, ..., r_i^m$.

**Remark.** Our co-attentive mechanism employs the idea of sequence pair modeling but notably differs from the conventional co-attention used in QA systems [Santos et al. 2016; Xiong, Zhong, and Socher 2017; Zhang et al. 2017]. First, we only consider one side of the affinity matrix, i.e., the user. Second, our affinity matrix is adapted by row-wise multiplication of $\alpha_v$ to quantify the utility of the item’s sentences. Thus, our method is specifically designed for learning asymmetric attentions from user-item interactions.

**Review-Level Aggregation**

From Eq. (4), we obtain review embeddings for an item, $r_1^v, ..., r_m^v$. From Eq. (5), we obtain review embeddings for a user, $r_1^u, ..., r_n^u$. As shown in Fig. 2 based on these review embeddings, we develop review-level aggregators to infer an embedding for each user and item, respectively.

As discussed before, different reviews exhibit different degrees of informativeness in modeling users and items. In particular, an item’s reviews are homogeneous. Thus, we are interested in reviews with rich descriptions regarding its relevant aspects and corresponding sentiments, such as the reviews 1–3 of $v$ in Fig. 4 compared with the less informative review 4 of $v$. To attend to such reviews, similar to Eq. (4), we aggregate the review embeddings to represent an item by

$$\tilde{v} = \sum_{i=1}^{m} \beta_i^v r_i^v, \quad (9)$$

where $\sum_{i=1}^{m} \beta_i^v = 1$, and $\beta_i^v$ is the attention weight assigned to review $r_i^v$. It quantifies the informativeness of the review $r_i^v$ with respect to $v$’s overall rating. $\beta_i^v$ is produced by an attentive module with gating mechanism as follows:

$$\beta_i^v = \frac{\exp(v_i^\top (\tanh(W_r r_i^v) \otimes \sigma(\tilde{W}_r r_i^v)))}{\sum_{j=1}^{m} \exp(v_j^\top (\tanh(W_r r_j^v) \otimes \sigma(\tilde{W}_r r_j^v)))}, \quad (10)$$

where $v_r \in \mathbb{R}^{h \times 1}$, $W_r \in \mathbb{R}^{h \times d}$, and $\tilde{W}_r \in \mathbb{R}^{h \times d}$ are model parameters.

At the same time, a user’s reviews are heterogeneous concerning a variety of items that the user has purchased, and not all reviews are relevant to the target item. Thus, similar to Eq. (6) and Eq. (7), given a user–item pair, a review-level co-attentive network is designed to select reviews from the user as guided by the reviews of the item.

Specifically, an affinity matrix at the review level

$$G = \phi(f([r_1^u; ...; r_m^u])^\top M_r f([r_1^v; ...; r_m^v])), \quad (11)$$

is computed, where $M_r \in \mathbb{R}^{d_r \times d_r}$ is a learnable parameter. Here, the $(p, q)$-th entry of $G$ represents the affinity between the $p$-th review of the user and the $q$-th review of the item.

Then, attention weights for the reviews of the user

$$\beta_u^v = \text{softmax}(\max_{\text{row}}(G \odot_{row} \beta^v)), \quad (12)$$

are obtained, where $\beta^v = [\beta_1^v, ..., \beta_m^v]$ was obtained by Eq. (10) for the item. It is introduced to adapt $G$ to encode important reviews of the item. Finally, we aggregate the review embeddings to represent a user by the following weighted sum.

$$\tilde{u} = \sum_{i=1}^{n} \beta_i^u r_i^u \quad (13)$$

**Encoding Latent Rating Patterns.** Although the embeddings $\tilde{u}$ and $\tilde{v}$ contain rich semantic information from reviews, there are some latent characteristics of users (items) that are not encoded by their reviews, but can be inferred from the rating patterns. For instance, a picky user might tend to uniformly pick lower ratings than a more easygoing user. To encode such personalized preferences, as inspired by Koren, Bell, and Volinsky (2009), we embed a
one-hot representation of the ID of each user (item) using an MLP, and obtain an embedding vector \( \tilde{u} \) for the user (item). This vector directly correlates with the ratings of a user (item), and is thus able to capture the latent rating patterns. Then, as illustrated in Fig. 2, we concatenate \( \tilde{u} \) and \( \tilde{u} \) to obtain the final embedding of a user, i.e., \( u = [\tilde{u}; \tilde{u}] \), and concatenate \( v \) and \( v \) to obtain the final embedding of an item, i.e., \( v = [\tilde{v}; \tilde{v}] \).

### Prediction Layer

As shown by the top part of Fig. 2, the prediction layer receives \( u \) and \( v \), and concatenates them to \( [u; v] \), which is then fed into a function \( g(\cdot) \) to predict the rating. In this work, we realize \( g(\cdot) \) as a parameterized factorization machine (FM) (Rendle 2010), which is effective to model the pairwise interactions between the input features for improving recommendation performance. Given an input \( x \in \mathbb{R}^{d \times 1} \), \( g(\cdot) \) is defined as

\[
g(x) = b + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} \langle z_i, z_j \rangle x_i x_j, \tag{14}
\]

where \( b \) is a bias term, \( w \) is a parameter for linear regression, \( \{z_i\}_{i=1}^{d} \) are the factorized parameter for modeling the pairwise interactions between \( x_i \) and \( x_j \), \( \langle \cdot, \cdot \rangle \) denotes the inner product, and the output of \( g(x) \) is the predicted rating.

To learn model parameters, we minimize the difference between the true ratings and the predicted ratings, as measured by the mean squared error

\[
\ell = \frac{1}{c} \sum_{i=1}^{c} (y_i - g([u; v]))^2, \tag{15}
\]

where \( c \) is the total number of user–item pairs in the training data, and \( y_i \) is the true rating of the \( i \)-th user–item pair. The \( \ell \) in Eq. (15) serves as our loss function for model training.

### 4 Experiments

In this section, we evaluate our AHN model on several real datasets and compare it with state-of-the-art approaches.

#### Datasets

We conducted experiments on 10 different datasets, including 9 Amazon product review datasets for 9 different domains, and the large-scale Yelp challenge dataset on restaurant reviews. Table 1 summarizes the domains and statistics for these datasets. Across all datasets, we follow the existing work (Seo et al. 2017; Tay, Luu, and Hui 2018) to perform preprocessing to ensure they are in a t-core fashion, i.e., the datasets only include users and items that have at least \( t \) reviews. In our experiments, we evaluate the two cases of \( t = 5 \) and \( t = 10 \). For the Yelp dataset, we follow Seo et al. (2017) to focus on restaurants in the AZ metropolitan area. For each dataset, we randomly split the user–item pairs into 80% training set, 10% validation set, and 10% testing set. When learning the representations for users and items, we only use their reviews from the training set, and none from the validation and testing sets. This ensures a practical scenario where we cannot include any future reviews into a user’s (item’s) history for model training.

#### Compared Methods

We compare our model with both conventional approaches and state-of-the-art approaches, including Factorization Machines (FM) (Rendle 2010), SVD (Koren, Bell, and Volinsky 2009), Probabilistic Matrix Factorization (PMF) (Mnih and Salakhutdinov 2008), Nonnegative Matrix Factorization (NMF) (Lee and Seung 2001), DeepCoNN (Zheng, Norouzi, and Yu 2017), D-ATT (Seo et al. 2017), MPCN (Tay, Luu, and Hui 2018), and HUITA (Wu et al. 2019).

Among these methods, FM, SVD, PMF, and NMF are rating-based collaborative filtering methods. DeepCoNN, D-ATT, MPCN, and HUITA are state-of-the-art methods that leverage the semantic information in reviews for improved performance. Specifically, DeepCoNN uses the same CNN module to learn user and item embeddings based on their reviews for recommendation. D-ATT extends DeepCoNN by adding a dual-attention layer at word-level before convolution. MPCN attends to informative reviews by several pointers. HUITA uses a symmetric hierarchical structure to infer user (item) embeddings using regular attention mechanisms. It is worth noting that all of the above review-based methods regard user reviews and item reviews as the same type of documents and process them in an identical way.

Finally, to gain further insights on some of the design choices of our AHN model, we compare AHN with its variants, which will be discussed later in the ablation analysis.

#### Experimental Settings

The parameters of the compared methods are selected based on their performance on the validation set. Specifically, for FM, the dimensionality of the factorized parameters is 10. For SVD, PMF, and NMF, the number of factors is set to 50. DeepCoNN uses 100 convolutional kernels with window size 3. D-ATT uses 200 filters and window size 5 for local attention; 100 filters and window sizes [2, 3, 4] for global attention. MPCN uses 3 pointers, and hidden dimensionality of 300 for inferring affinity matrix. HUITA uses 200 filters in the word-level CNN with window size 3, and 100 filters in the sentence-level CNN with window size 3.

---

1https://www.yelp.com/dataset/challenge
Table 2: MSE results of the compared methods on different 5-core datasets

| Dataset            | FM    | PMF   | NMF   | SVD   | DeepCoNN | D-ATT  | MPCN   | HUITA  | AHN    |
|--------------------|-------|-------|-------|-------|----------|--------|--------|--------|--------|
| Digital_Music (DM) | 0.8498| 0.8788| 1.0491| 1.0843| 0.8754   | 0.8506 | 0.8396 | 0.8719 | 0.8172 |
| Office_Products (OP)| 0.7291| 0.7807| 0.9285| 0.7906| 0.7253   | 0.7124 | 0.7084 | 0.7082 | 0.6825 |
| Health (HE)        | 1.1825| 1.2076| 1.4317| 1.1508| 1.0862   | 1.0915 | 1.0817 | 1.1207 | 1.0743 |
| Toys_and_Games (TG)| 0.8639| 0.9192| 1.1105| 0.9188| 0.8391   | 0.8364 | 0.8577 | 0.6544 | 0.6270 |
| Kindle_Store (KS)  | 0.6469| 0.6695| 0.8032| 0.7370| 0.6814   | 0.6832 | 0.7032 | 0.6704 | 0.6282 |
| Pets_Supplies (PS) | 1.3303| 1.4344| 1.6806| 1.3191| 1.2598   | 1.2730 | 1.2566 | 1.3038 | 1.2515 |
| Tools_and_Home (TH)| 1.0229| 1.1182| 1.3580| 1.0373| 1.0817   | 1.0815 | 1.0747 | 1.1172 | 1.1138 |
| Videos_Games (VG) | 1.1849| 1.2473| 1.4357| 1.1575| 1.1548   | 1.1448 | 1.1747 | 1.1772 | 1.1138 |
| Automotive (AM)    | 0.8189| 0.9187| 1.2074| 0.8140| 0.7809   | 0.7654 | 0.7643 | 0.7676 | 0.7314 |
| Yelp               | 1.6094| 1.8207| 1.8389| 1.6615| 1.5957   | 1.5959 | 1.6195 | 1.6105 | 1.5735 |

Table 3: MSE results of the compared methods on different 10-core datasets

| Dataset            | FM    | PMF   | NMF   | SVD   | DeepCoNN | D-ATT  | MPCN   | HUITA  | AHN    |
|--------------------|-------|-------|-------|-------|----------|--------|--------|--------|--------|
| Digital_Music (DM) | 0.8611| 0.8641| 0.9491| 0.8503| 0.8734   | 0.8429 | 0.8629 | 0.7880 | 0.7880 |
| Office_Products (OP)| 0.6291| 0.6695| 0.7346| 0.6757| 0.6016   | 0.5914 | 0.6120 | 0.6009 | 0.5717 |
| Health (HE)        | 0.8166| 0.9138| 0.9200| 0.8275| 0.8328   | 0.8019 | 0.8020 | 0.8177 | 0.7802 |
| Toys_and_Games (TG)| 0.6904| 0.6233| 0.7575| 0.6331| 0.6331   | 0.6292 | 0.6412 | 0.6303 | 0.5964 |
| Kindle_Store (KS)  | 0.5954| 0.6035| 0.6305| 0.6483| 0.5325   | 0.5275 | 0.5124 | 0.3312 | 0.5092 |
| Pets_Supplies (PS) | 1.2236| 1.5239| 1.2536| 0.9950| 0.9927   | 0.9616 | 1.0722 | 1.0168 | 0.9421 |
| Tools_and_Home (TH)| 0.8746| 0.7688| 0.9032| 0.7391| 0.6632   | 0.6297 | 0.6507 | 0.6445 | 0.5948 |
| Videos_Games (VG) | 1.0611| 1.0718| 1.2435| 1.0318| 1.0743   | 1.0365 | 1.0730 | 1.0697 | 0.9927 |
| Yelp               | 1.5432| 1.4734| 1.5735| 1.4025| 1.3961   | 1.4018 | 1.4033 | 1.4040 | 1.3671 |

Figure 3: The relative improvements of AHN over (a) DeepCoNN, (b) D-ATT, (c) MPCN, and (d) HUITA, on different datasets. The abbreviations of the datasets can be found in Table 1-3. Here, blue refers to the improvement on the 10-core datasets, orange refers to the improvement on the 5-core datasets, brown is the overlapped area between blue and orange.

For our AHN model, the dimensionality of the hidden states of the BiLSTM is set to 150. The dimensionality of the user and item ID embeddings are set to 300. The dimensionality of $M_s$ ($M_r$) in Eq. (6) (Eq. (11)) is 300. We apply dropout (Srivastava et al. 2014) with rate 0.5 after the fully connected layer to alleviate the overfitting problem. The loss function is optimized by Adam (Kingma and Ba 2014), with a learning rate of 0.0002 and a maximum of 10 epochs.

For the methods DeepCoNN, D-ATT, and HUITA, the pre-trained GloVe (Pennington, Socher, and Manning 2014) are used to initialize the word embeddings. For MPCN and our AHN, the word embeddings are learned from scratch since using pre-trained embeddings generally degrades their performance. For all methods, the dimensionality of the word embedding is set to 300. We independently repeat each experiment 5 times, and use the averaged mean square error (MSE) (Zheng, Noroozi, and Yu 2017) to quantitatively evaluate the performance.

**Experimental Results**

Table 2 summarizes the results of the compared approaches on the 5-core datasets. We have several observations from the results. First, review-based methods generally outperform rating-based methods. This validates the usefulness of reviews in providing fine-grained information for refining user and item embeddings for improving the accuracy of rating prediction. Second, methods that distinguish reviews, such as D-ATT and MPCN, often outperform DeepCoNN, which suggests that different reviews exhibit different degrees of importance for modeling users and items. We also observe that HUITA does not show superiority over DeepCoNN. This may stem from its symmetric style of attention.
5 Conclusions

We next investigate the interpretability of AHN. Fig. 4(a) shows the interpretation of AHN on the top three sentences on both the user and item sides. The item is a sleep aid medicine. The vertical bars represent weights. Darker colors indicate higher weights.

In this work, we highlight the asymmetric attention problem for review-based recommendation, which has been ignored by existing approaches. To address it, we propose a flexible and hard selection mechanism to extract fine-grained signals from sentences and reviews. Extensive experimental results on datasets from different domains demonstrate the effectiveness and interpretability of our method.

Table 4: Ablation analysis

| Model          | (a) Item aggregation | (b) User aggregators | (c) Adapted affinity | (d) FM | (e) Gating |
|----------------|----------------------|----------------------|----------------------|--------|-----------|
| VG             | 0.8172               | 0.8172               | 0.7467               | 0.7348 | 0.7091    |
| DM             | 0.8172               | 0.8172               | 0.7467               | 0.7348 | 0.7091    |
| AM             | 0.8172               | 0.8172               | 0.7467               | 0.7348 | 0.7091    |
| OP             | 0.8172               | 0.8172               | 0.7467               | 0.7348 | 0.7091    |

Figure 4: The visualization of attention weights on (a) user’s reviews, (b) item’s reviews, (c) user’s sentences, (d) item’s sentences. The model architecture is replaced by a dot product. (a) The user’s attention modules are replaced by average-pooling. In (b), the item’s attention modules are replaced by average-pooling. In (c), the user co-attention modules are replaced by average-pooling. In (d), the item co-attention modules are replaced by average-pooling. In (e), the attention-adapted affinity matrix in Eqs. (7) and (12) is used. Comparing with the baseline methods. In (a), the attention weights in Eqs. (7) and (12) are removed. From Table 4, we observe that different variants of AHN perform better than the baseline methods. In (c), the user’s attention modules are replaced by average-pooling. In (d), the item’s attention modules are replaced by average-pooling. In (e), the attention-adapted affinity matrix in Eqs. (7) and (12) is used.
reviews for recommendation. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, 425–434. ACM.