https://github.com/almightyGOSU/ANR

ANR: Aspect-based Neural Recommender

Jin Yao Chin
Nanyang Technological University, Singapore.
s160005@e.ntu.edu.sg

Kaiqi Zhao
Nanyang Technological University, Singapore.
kqzhao@ntu.edu.sg

Shafiq Joty
Nanyang Technological University, Singapore.
srjoty@ntu.edu.sg

Gao Cong
Nanyang Technological University, Singapore.
gaocong@ntu.edu.sg

ABSTRACT

Textual reviews, which are readily available on many e-commerce and review websites such as *Amazon* and *Yelp*, serve as an invaluable source of information for recommender systems. However, not all parts of the reviews are equally important, and the same choice of words may reflect a different meaning based on its context. In this paper, we propose a novel end-to-end ***A**spect-based **N**eural **R**ecommender* (**ANR**) to perform aspect-based representation learning for both users and items via an attention-based component. Furthermore, we model the multi-faceted process behind how users rate items by estimating the aspect-level user and item importance based on the neural co-attention mechanism. Our proposed model concurrently address several shortcomings of existing recommender systems, and a thorough experimental study on 25 benchmark datasets from *Amazon* and *Yelp* shows that ANR significantly outperforms recently proposed state-of-the-art baselines such as DeepCoNN, D-Attn and ALFM.

CCS CONCEPTS

• **Information systems** → **Recommender systems**;

KEYWORDS

Recommender Systems; Aspect-based Recommendation; Neural Attention; Co-Attention

ACM Reference Format:
Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based Neural Recommender. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271810

1 INTRODUCTION

With the shift towards an increasingly digital lifestyle, recommender systems play a critical role in helping consumers to find the best product or service amongst a variety of options. Some of the most widely used and successful recommendation systems rely on the Collaborative Filtering (**CF**) technique, which utilizes past interaction data such as ratings, purchase logs, or viewing history, to model user preferences and item features [22]. However, a major limitation of CF techniques such as Matrix Factorization (**MF**) is its inability to provide reliable recommendations to users with few ratings, or recommend items with limited ratings, i.e. the well-known cold start problem in real-world recommendation systems.

无法向评级低的用户推荐可靠的信息

Recent recommender systems have considered another valuable source of information which is readily available in many e-commerce and review websites such as *Amazon* and *Yelp*: free-text reviews. More often than not, users provide an accompanying review to *explain* why they liked or disliked that particular product or service, i.e., the reasons behind the overall numerical rating. For example, a review may include the user's opinions on the various *aspects* of an item, such as its price, performance, quality, etc. In fact, reviews provide more than just an avenue for modeling the implicit user preferences or item properties. The rich semantic information in these reviews can be useful in helping us understand the multi-faceted process behind how users tend to rate items, i.e. the *key factors* which influence a user to prefer one item over the other.

隐式反馈：用户评论，这里给出了一个关于重要性的议论。

Owing to its superior representation learning capabilities, deep learning techniques have been widely used in recent state-of-the-art recommendation systems to construct latent user and item representations using the review contents. This includes models such as DeepCoNN [44], D-Attn [33], and TransNets [7], all of which are based on using Convolutional Neural Networks (**CNNs**) [20] to encode the user (and item) reviews into their corresponding latent embeddings. While these proposed methods have been shown to provide good predictive performance, their approach of simply inferring a single low-dimensional latent representation for each user (and item) would inherently be limited by its inability to capture the finer-grained interactions between users and items.

批评了一下现存深度学习方法的不足——无法捕捉细粒度的用户交互特征。

Intuitively, not all parts of a review are equally important. For example, some parts of the review may be describing the plot of a movie, or even the storyline in a book, and such *'details'* may not be correlated with the overall user satisfaction. A common observation is that each part of the review tends to focus on a different facet of the user's overall experience, such as the location of a restaurant, the attitude of its service staff, or even the taste of the dishes served in that restaurant. By focusing on these salient factors, we can better infer both the preferences of a specific user (E.g. User X prefers a restaurant with outdoor seating) and the properties of an item (E.g. Restaurant Y is famous for its seafood dishes).

However, to model the rich semantics of review contents, it is imperative to move beyond the surface-level word representations. Consider the following two sentences which contain the word *'long'*: **(1)** "This laptop has a long battery life", and **(2)** "The laptop

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271810

requires a long startup time". It is evident that the word *'long'* bears a positive sentiment towards the target aspect (or item property) in the first sentence, while the same word indicates a negative sentiment for the exact same item in the second sentence. Consequently, a flexible word representation scheme which is able to take into consideration such contextual information w.r.t. any given aspect would be desirable.

Additionally, different users may emphasize more on different aspects throughout their interactions with these items. For example, some user may like a particular ***restaurant*** for its *food*, while another user frequents the same restaurant due to its cozy *ambiance*. Similarly, a user may prioritize the *storyline* when choosing a ***horror movie***, but pays more attention to the *cast* when evaluating an ***action movie***. Understandably, the importance of each aspect largely depends on both the user and item in question, and being able to capture such dynamic and fine-grained interactions between users and items would be invaluable in determining why some user may prefer an item over the other. In this paper, we aim to model this crucial information for recommendation.

The key contributions of this paper are summarized as follows:

- We propose a novel aspect-based neural recommender system which performs aspect-based representation learning for users and items by designing an attention mechanism to focus on the relevant parts of these reviews while learning the representation of aspects on the task. Furthermore, we estimate aspect-level user and item importance in a joint manner using the idea of co-attention, which allows us to model the finer-grained interactions between users and items. To the best of our knowledge, this is the first paper to propose an end-to-end neural aspect-based recommender system which concurrently addresses the above-mentioned requirements.
- Extensive experiments have been conducted on 25 benchmark datasets from *Amazon* and *Yelp* to evaluate our proposed model against several state-of-the-art baselines such as DeepCoNN [44], D-Attn [33], and ALFM [10].
- We investigate how the different components in our proposed model contribute to its effectiveness. In particular, we include an qualitative analysis of the aspects which are learned automatically by our model without any external supervision.

2 RELATED WORK

Recent work [1, 3, 4, 7, 24, 26, 33, 35, 40, 44] have all shown the importance of using reviews to improve the performance and reliability of recommender systems. As such, we focus on several key areas that are highly relevant to our work: **(1)** Deep Learning-based Recommender Systems, **(2)** Aspect-based Recommendation Systems, as well as **(3)** Neural Attention & Co-Attention.

2.1 Deep Learning-based Recommender Systems

In recent years, deep neural networks have been successfully applied to a large variety of tasks, such as natural language processing, computer vision, and speech recognition [14], often achieving state-of-the-art performance in these domains. Many recently proposed recommender systems have also turned to various deep learning techniques in order to work with the textual information, such as the use of Denoising Autoencoders in [23, 39], Recurrent Neural Networks (**RNNs**) in [1, 3], and most notably, the use of Convolutional Neural Networks (**CNNs**) [20] in [7, 8, 19, 33, 44] due to its great success in many other natural language processing tasks [11, 18]

Generally, these methods try to capitalize on the strong representation learning capabilities of neural networks to learn latent feature representations from the reviews for both users and items. However, attempting to *'compress'* all the available reviews for a user (or item) into a single latent representation may not be ideal. Besides the potential loss of useful information (due to the pooling techniques used in such models), there is an inherent risk of including the irrelevant parts of these reviews, resulting in a noisy and possibly inaccurate representation of the user (or item). Furthermore, the only interaction between users and items occurs at the final prediction layer, where the learned user and item embeddings are used for the overall rating estimation using methods such as Factorization Machine (**FM**) [31] in [7, 44], Feedforward neural networks in [8], or simply via the inner product as in [33]. In these models, it can be difficult to provide a convincing insight as to why the user rated an item in that particular manner.

2.2 Aspect-based Recommendation Systems

Prior to the surge in utilizing deep learning techniques for recommendation, a popular line of research focuses on either extracting or learning aspects from these textual reviews.

The first type of aspect-based recommendation systems such as EFM [43], TriRank [16], LRPPM [9], and the recently proposed SULM [5], relies on external Sentiment Analysis (**SA**) tools [30] to analyze the review contents and uncover the mentioned aspects together with their opinions and/or sentiments. Besides the fact that they are not self-contained, the performance of such models largely depends on the quality of these SA tools, i.e. how well they are able to extract such information from these textual reviews.

An alternative type of aspect-based systems [10, 12, 40] automatically learn these aspects from the review contents, typically through the use of generative statistical models such as Latent Dirichlet Allocation (**LDA**) [6, 42]. JMARS [12] and FLAME [40] are both integrated probabilistic frameworks which represent each aspect as a distribution over the words in the vocabulary. The newly proposed ALFM [10] includes an Aspect-aware Topic Model (**ATM**) which models each aspect as a multinomial distribution over the same set of K latent topics, each of which is defined as a multinomial distribution over the vocabulary. The output from ATM, i.e. the aspect-level user preferences and item characteristics, is subsequently used as part of their latent factor model for estimating the overall rating via the MF approach.

A key advantage to these aspect-based methods is that they are generally more transparent and intuitive, as most of them are capable of providing explanation in order to support their recommendations. However, existing aspect-based systems either (1) depend on external tools or input, or (2) does not emphasize on how different parts of the review may contribute differently to the overall satisfaction. Additionally, they fail to consider the varied aspect-level importance for both users and items while taking into account the target user and item in question (as and when necessary).

Table 1: Notations and their definitions[1]

Notation	Definition
$\mathcal{D}$	Corpus with Ratings & Reviews
$(u, i, r_{u,i}, d_{u,i})$	Complete User-Item Interaction
$r_{u,i}$	Rating from User u for Item i
$d_{u,i}$	Review from User u for Item i
D_u	User Document (Set of Reviews from User u)
D_i	Item Document (Set of Reviews for Item i)
$\mathcal{A}$	Set of K Aspects
$\mathbf{v}_a$	Embedding Vector for Aspect $a \in \mathcal{A}$
$\mathbf{W}_a$	Word Projection Matrix for Aspect $a \in \mathcal{A}$
$\mathbf{p}_{u,a}$	Latent Representation of User u for Aspect a
$\mathbf{q}_{i,a}$	Latent Representation of Item i for Aspect a
$\boldsymbol{\beta}_{u,a}$	Importance of Aspect a for User u
$\boldsymbol{\beta}_{i,a}$	Importance of Aspect a for Item i

2.3 Neural Attention & Co-Attention

Loosely based on the idea of visual attention in humans, the neural attention mechanism is one of the most exciting developments in the field of deep learning, and has been successfully applied to a multitude of machine learning tasks such as machine translation and abstractive summarization [2, 32, 37]. More recently, it has also been utilized in various recommender systems [8, 33, 36]. In essence, it equips neural networks with the ability to focus on selective parts of the input, such as a certain region in an image or even specific words/sentences in a textual document.

For example, if we are trying to determine the suitability of some restaurant based on its *price*, not all words in its set of user reviews would be equally important. Almost instinctively, we would turn our attention to a subset of informative words in these reviews, such as *expensive*, *cheap*, *costly*, *affordable*, etc. This is the central idea behind how our proposed model is able to automatically derive the aspect-level representations from the corresponding textual contents using a fully data-driven approach. Basically, the model learns to identify a subset of vocabulary words which are highly relevant given some target aspect via the neural attention mechanism.

A closely related technique is neural co-attention [25, 41], which can be roughly described as a form of pairwise neural attention. In certain scenarios, it can be beneficial to jointly reason about the attention for a pair of related entities, such as between the image and question for the task of Visual Question Answering in [25]. The basic idea behind the neural co-attention mechanism is that the attention for one entity (e.g. image) is learned w.r.t. the representation(s) of the other entity (e.g. question), and vice versa.

For our model, we extend this particular idea of a two-way neural attention for the estimation of the aspect-level user and item importance, enhancing it with the ability to be aware of the current user-item pair. The *aspect-level* ***item*** *representations* are used as the context to influence the learning of *aspect-level* ***user*** *importance*, and conversely, the *aspect-level* ***item*** *importance* are conditioned on the *aspect-level* ***user*** *representations*. In other words, our proposed model takes into consideration the target item when inferring the importance of each aspect for the user, and vice versa.

3 PROPOSED MODEL

In this section, we present our proposed **A**spect-based **N**eural **R**ecommender (**ANR**), a neural recommendation system which aims to capture the finer-grained interactions between users and items at an aspect-level. First, we specify the problem setting and key notations used, and present an overview of our architecture along with the motivations behind some of the key components. Following which, we describe in detail our attention-based module for learning the aspect-level user (and item) representations. Next, we will show our co-attention-based module for dynamically inferring the aspect-level importance for any given user-item pair, as well as how the aspect-level representations and importance can be combined effectively to infer the overall rating. Lastly, we will go through the model optimization details for ANR.

3.1 Problem Setting

Considering a corpus of ratings and reviews $\mathcal{D}$, for a set of items $\mathcal{I}$ and a set of users $\mathcal{U}$, each user-item interaction can be represented as a tuple $(u, i, r_{u,i}, d_{u,i})$ where $r_{u,i}$ is a numerical rating denoting user u's overall satisfaction towards item i, and $d_{u,i}$ is the accompanying textual review. The primary objective is to estimate the rating $\hat{r}_{u,i}$ for any unseen user-item pair, i.e. the unknown rating of a given user u towards an item i that he/she has not interacted with before. Table 1 summarizes the key notations used throughout the rest of this paper.

3.2 Overview of ANR

Figure 1 shows the overall architecture of our proposed model. Similar to [33, 44], we feed the user document D_u and item document D_i, i.e. the set of reviews written by the user u and the set of reviews written for item i, respectively, as the inputs to the network. Since the modeling process for users and items are identical, we focus on illustrating the process for a given user. It should be noted that the construction of user and item documents is constrained to the set of reviews from the training split, i.e. they do **not** include any review from the validation or testing split.

Embedding Layer. First, the user document D_u is transformed into a matrix $\mathbf{M}_u \in \mathbb{R}^{n \times d}$ via an embedding layer, where n is the number of words in D_u, and d is the number of dimensions for each word embedding vector. Basically, the embedding layer performs a look-up operation in a shared embedding matrix $f : \mathcal{V} \rightarrow \mathbb{R}^d$ which maps each word in the vocabulary $\mathcal{V}$ to its corresponding d-dimensional vector. The embedding matrix can be initialized using word vectors that have been pre-trained on large corpora, such as *word2vec*[2] [28] or *GloVe*[3] [29], which facilitates a better semantic representation of the user (and item) documents. Unlike topic modeling-based methods which rely on the *bag-of-words* assumption, the order and context of words is preserved in the embedded document.

Aspect-based Representation Learning. For example, considering the domain of *restaurants*, the aspect set $\mathcal{A}$ could include aspects such as *price*, *quality*, *service*, *location*, etc. In other words,

[1] Unless stated otherwise, we denote vectors with bold lower-cases, and bold upper-cases are reserved for matrices or high dimensional tensors.

[2] https://code.google.com/archive/p/word2vec/

[3] https://nlp.stanford.edu/projects/glove/

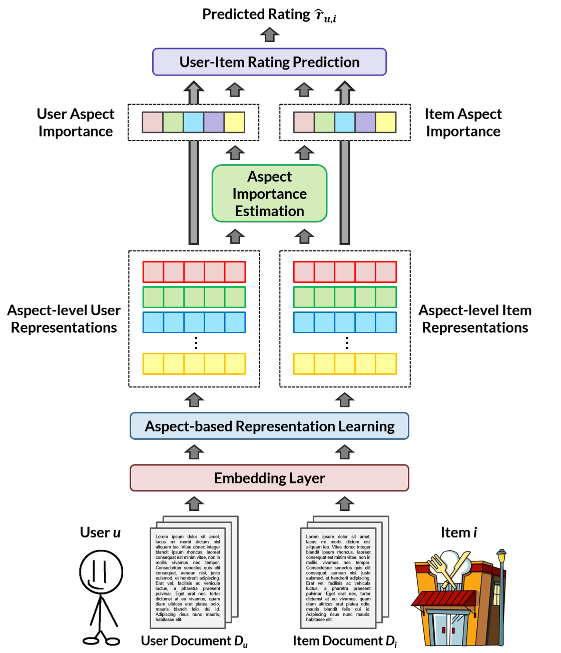

Predicted Rating $\hat{r}_{u,i}$
User-Item Rating Prediction
User Aspect Importance
Item Aspect Importance
Aspect Importance Estimation
Aspect-level User Representations
Aspect-level Item Representations
Aspect-based Representation Learning
Embedding Layer
User u
Item i
User Document D_u
Item Document D_i

Figure 1: Overall architecture of the proposed model

an *aspect* can be defined as a high-level semantic concept encompassing a specific facet of item properties for a given domain. For *restaurants*, the aspect *service* can encompass properties such as *{staff, waiting time, reservation, valet parking, . . . }*.

Given the embedded user document representation $\mathbf{M}_u$, our goal here is to derive a set of aspect-level user representations $\mathbf{P}_u = \{\mathbf{p}_{u,a} \mid a \in \mathcal{A}\}$ w.r.t. a set of K domain-dependent aspects, $\mathcal{A}$. Intuitively, the review $d_{u,i}$ describes user u's opinions towards item i based on this set (or possibly, subset) of aspects. Consequently, the user document D_u covers user u's opinions towards $\mathcal{A}$ aggregated across all the items that he/she has previously interacted with. Similarly, the item document D_i describes the properties of item i w.r.t. $\mathcal{A}$ aggregated across all the users that have reviewed it.

Our hypothesis is that given sufficient data, we can learn this set of aspects $\mathcal{A}$, as well as the aspect-level user (or item) representation for each aspect $a \in \mathcal{A}$, by learning to attend to a subset of aspect-related words within each user (or item) document. In this paper, we propose a novel aspect-aware attention-based component for learning these aspect-level representations, and the details are presented in Section 3.3.

Aspect Importance. It is not uncommon for different users to have varied preferences for the different aspects of an item. Furthermore, for a given user, his/her aspect preferences may change depending on the target item. For instance, a user may focus on the *price* and *aesthetics* when choosing a ***mobile phone***, while he/she may be more concerned about the *performance* and *portability* when purchasing a ***laptop***. Likewise, the same item may appeal differently to two different users. To illustrate, some user may like a particular ***restaurant*** for its *food*, while another user frequents the same restaurant due to its cozy *ambiance*.

Additionally, these aspects are often not evaluated separately. For example, a user may be willing to overlook the steep *price* of a ***mobile phone*** if its *quality* and *performance* exceeds his/her expectations, even though the historical data may indicate that this user would generally prefer *cheaper* ***mobile phones***.

As such, rather than having static user and item aspect importance, our new idea is to model the rich and complex interactions between users and items at the aspect level by dynamically estimating the user and item aspect importance for each user-item pair. In this paper, we propose a novel co-attention-based component which is able to consider these crucial observations for recommendation, and the details are presented in Section 3.4.

3.3 Aspect-based Representation Learning

Before delving into the specific details of our proposed aspect-based representation learning approach, we highlight some vital intuitions which we aim to capture through this component.

Intuition 1: Not all words in a review (or document) are equally important, and the importance of each document word varies w.r.t. the aspect being considered. Generally, reviews tend to include opinions towards multiple aspects of the target item, and we should be able to focus on specific subparts of the review (or document) when learning the aspect-level representation for a given aspect.

Intuition 2: The sentimental polarity of the same word could be completely different for two different aspects in the same domain. For example, the word '*high*' in the sentences "This phone has a high storage capacity" and "This camera captures high quality images" carries a positive sentiment towards the target aspect (or item property). On the other hand, considering the sentences "The price is way too high" and "This computer has extremely high power consumption", the same word actually reflects a negative sentiment. In fact, many of these sentiment-bearing words tend to indicate a different polarity based on the aspect being considered, and this should be captured in the aspect-level representations.

Intuition 3: It has been well-established that aspect-related words (e.g. *price*, *taste*, *ambiance*) and their sentiment-bearing words (e.g. *expensive*, *delicious*, *amazing*) are often in close proximity [17]. This implies that we can better infer the importance of a word within the document by looking at its surrounding words, i.e. by considering a local context window.

Now, we describe how the aspect-level user representation, i.e. $\mathbf{p}_{u,a}$, can be obtained for user u and a given aspect $a \in \mathcal{A}$. Since all words in the vocabulary $\mathcal{V}$ share the same d-dimensional vector across the K aspects, we use an aspect-specific word projection matrix[4] $\mathbf{W}_a \in \mathbb{R}^{d\,\text{x}\,h_1}$ to allow variations in the word representations w.r.t. the target aspect a (**Intuition 2**). More formally,

$$\mathbf{M}_{u,a}[i] = \mathbf{M}_u[i]\,\mathbf{W}_a \tag{1}$$

where $\mathbf{M}_u[i]$ is the **original** d-dimensional word embedding for the i-th word in $\mathbf{M}_u$, $\mathbf{M}_{u,a}[i]$ is the **aspect-specific** word representation, and $\mathbf{M}_{u,a} \in \mathbb{R}^{n\,\text{x}\,h_1}$ is the **aspect-specific** document embedding for user u and aspect a. The result of this projection is a tensor in $\mathbb{R}^{K\,\text{x}\,n\,\text{x}\,h_1}$ for K different aspects.

Each aspect $a \in \mathcal{A}$ is represented as an embedding vector $\mathbf{v}_a \in \mathbb{R}^{(c\,\text{x}\,h_1)}$ with length $c \times h_1$, where c is a hyperparameter which determines the width (in terms of the number of words) of the local

[4] Note that h_1 is a hyperparameter which allows the number of latent factors used for the aspect-level representations to be defined, without being constrained by the size of the original word embeddings.

context window (**Intuition 3**). To compute the importance of the i-th document word in this **aspect-specific** embedding subspace, 2 we consider a local context window with it as the center word:

$$\mathbf{z}_{u,a,i} = (\mathbf{M}_{u,a}[i - c/2]; \ldots; \mathbf{M}_{u,a}[i]; \ldots; \mathbf{M}_{u,a}[i + c/2]) \quad (2)$$

where $(\cdot\,;\,\cdot)$ is the concatenation operator. We calculate the attention 3 score for the i-th word by taking the inner product followed by the softmax function:

$$\mathbf{attn}_{u,a}[i] = softmax(\mathbf{v}_a\,(\mathbf{z}_{u,a,i})^\top) \quad (3)$$

where $softmax(w_i) = exp(w_i)\,/\,\sum_j exp(w_j)$, and $\mathbf{attn}_{u,a}$ is the soft attention vector (i.e. a probability distribution) defined over the document words for user u w.r.t. aspect a. Intrinsically, the importance of the i-th word in the document depends on both the word itself and its surrounding words (**Intuition 3**). Taking into consideration the learned importance of each word in the document (**Intuition 1**), the aspect-level user representation can be derived 4 based on the following weighted sum:

$$\mathbf{p}_{u,a} = \sum_{i=1}^{n} (\mathbf{attn}_{u,a}[i]\,\mathbf{M}_{u,a}[i]) \quad (4)$$

The aspect-level item representation $\mathbf{q}_{i,a}$ for item i and aspect a can be obtained in a similar manner, following Equations (1) to (4). Additionally, for each aspect $a \in \mathcal{A}$, we share the aspect embedding vector $\mathbf{v}_a$ and aspect-specific word projection matrix $\mathbf{W}_a$ for users and items (i.e. the aspect-level user and item representations for each aspect a reside in the same aspect-specific feature space). Sharing aspect-specific parameters allows us to learn better mapping between aspects in user and item documents while reducing the number of trainable parameters in the model. We denote the set of parameters for the *Aspect-based **R**epresentation **L**earning* layer as $\Theta_{ARL} = \{\mathbf{v}_a, \mathbf{W}_a \mid a \in \mathcal{A}\}$[5].

3.4 Aspect Importance Estimation

A straightforward solution is that we could attempt to estimate the user and item aspect importance separately. However, this would result in 'static' user and item aspect importance, whereby the user aspect importance does not actually take into consideration the actual item of interest, and vice versa. In other words, the user (item) aspect importance remains fixed across all possible items (users), and would be suboptimal for any given user-item pair since it is not derived specifically for the user and item in question.

To this end, we propose learning the user and item aspect importance in a joint manner. The aspect-level item representations are used as the context when learning the user aspect importance, and similarly, the user aspect-level user representations can be used as the context when learning the item aspect importance. The output of this layer would be a K-dimensional vector indicating the importance of each aspect for the user, and a corresponding K-dimensional vector for the item.

In order to incorporate the aspect-level item representations when calculating the user aspect importance (and vice versa), we need to know how the target user and item matches at an aspect-level. First, using the aspect-level user representation $\mathbf{P}_u \in \mathbb{R}^{K \times h_1}$

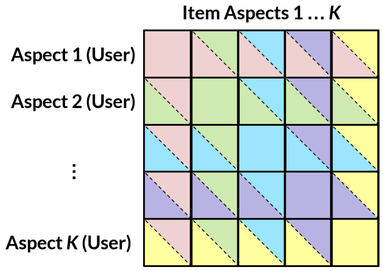


Figure 2: Aspect-level Affinity Matrix (*Best viewed in color*)

and item representation $\mathbf{Q}_i \in \mathbb{R}^{K \times h_1}$, we can obtain an aspect-level affinity matrix $\mathbf{S}$ as follows:

$$\mathbf{S} = \phi(\mathbf{P}_u\,\mathbf{W}_s\,\mathbf{Q}_i^\top) \quad (5)$$

where $\mathbf{W}_s \in \mathbb{R}^{h_1 \times h_1}$ is a learnable weight matrix, $\phi(x) = max(0, x)$ is the ReLU function, and each entry in the affinity matrix $\mathbf{S} \in \mathbb{R}^{K \times K}$ denotes the *affinity* (or *shared similarity*) between the corresponding pair of aspect-level user and item representations. We provide the illustration of an affinity matrix with K aspects in Figure 2.

Next, following [25], we use the affinity matrix $\mathbf{S}$ as a *feature* to estimate the aspect-level user and item importance:

$$\mathbf{H}_u = \phi(\mathbf{P}_u\mathbf{W}_x + \mathbf{S}^\top(\mathbf{Q}_i\mathbf{W}_y)),\ \boldsymbol{\beta}_u = softmax(\mathbf{H}_u\mathbf{v}_x) \quad (6)$$

$$\mathbf{H}_i = \phi(\mathbf{Q}_i\mathbf{W}_y + \mathbf{S}(\mathbf{P}_u\mathbf{W}_x)),\ \boldsymbol{\beta}_i = softmax(\mathbf{H}_i\mathbf{v}_y) \quad (7)$$

where $\mathbf{W}_x, \mathbf{W}_y \in \mathbb{R}^{h_1 \times h_2}$ and $\mathbf{v}_x, \mathbf{v}_y \in \mathbb{R}^{h_2}$ are the learnable parameters. $\boldsymbol{\beta}_u \in \mathbb{R}^K$ and $\boldsymbol{\beta}_i \in \mathbb{R}^K$ are the estimated aspect importance over the set of K aspects $\mathcal{A}$ for user u and item i, respectively. Basically, we consider both the user representation $\mathbf{P}_u$ and the item representation $\mathbf{Q}_i$ when computing $\mathbf{H}_u$ and $\mathbf{H}_i$. Considering the fact that the aspect-level representations may be substantially different between (1) users and items, (2) two different users, and (3) two different items, we find that these additional hidden layers improve the model performance by allowing it to better estimate the pairwise aspect-level importance for any given user-item pair.

As highlighted previously, we specifically designed this component to take into consideration the target user and item, enabling such estimation of the aspect-level importance to be personalized for both the user and item. We denote the set of parameters for the *Aspect Importance Estimation* layer as $\Theta_{AIE} = \{\mathbf{W}_s, \mathbf{W}_x, \mathbf{W}_y, \mathbf{v}_x, \mathbf{v}_y\}$.

3.5 Model Inference and Optimization

We now describe the *User-Item Rating Prediction* component shown in Figure 1. By combining the user and item aspect-level representations $\mathbf{P}_u, \mathbf{Q}_i$ with the aspect importance $\boldsymbol{\beta}_u, \boldsymbol{\beta}_i$, the overall rating for any user-item pair can be inferred as follows:

$$\hat{r}_{u,i} = \sum_{a \in \mathcal{A}} \overbrace{(\boldsymbol{\beta}_{u,a} \cdot \boldsymbol{\beta}_{i,a}}^{\textbf{Aspect Importance}} \cdot \underbrace{(\mathbf{p}_{u,a}\,(\mathbf{q}_{i,a})^\top)}_{\textbf{Aspect-based Representations}} + b_u + b_i + b_0 \quad (8)$$

where b_u, b_i, b_0 are the user, item, and global bias (as in traditional latent factor models), respectively. The model optimization process can be viewed as a regression problem and the complete set of model

[5] For each aspect a, both the aspect embedding vector $\mathbf{v}_a$ and the aspect-specific projection matrix $\mathbf{W}_a$ are initialized randomly using a uniform distribution $\mathcal{U}(-0.01, 0.01)$. Essentially, the aspect-based representation learning is performed in a data-driven manner without any external supervision.

parameters $\Theta = \{\Theta_{ARL}, \Theta_{AIE}, \mathbf{b}_u, \mathbf{b}_i, b_0\}$ can be learned using the backpropagation technique with the standard Mean Squared Error (**MSE**) as the loss function.

Pre-training. It has been shown in [13] that the performance of neural networks can be rather sensitive to how the parameters are initialized. For our proposed model, the *Aspect Importance Estimation* component is fully based on the outputs $\mathbf{P}_u$, $\mathbf{Q}_i$ from the previous *Aspect-based Representation Learning* layer, i.e. it implicitly relies on the set of parameters $\Theta_{ARL} = \{\mathbf{v}_a, \mathbf{W}_a \mid a \in \mathcal{A}\}$. As such, we employed a *pre-training phase* using a simplified model to obtain a good initialization for Θ_{ARL}. We replaced the *Aspect Importance Estimation* component with two feed-forward neural networks, for users and items, respectively. The user (item) network takes as input the concatenation of the aspect-level user (item) representations, and produces an abstract user (item) representation. These abstract user and item representations are then concatenated and used for predicting the overall rating $\hat{r}_{u,i}$ via another feed-forward layer. This simplified model does not consider the aspect-level interactions between users and items, and it is trained in a similar fashion using the backpropagation method with the MSE loss function.

Generalization. Many existing work have found that deep learning models tend to suffer from overfitting. In order to improve the generalization performance, we adopt the dropout technique [34], which is widely used in existing neural models for recommendation [7, 8, 33, 44]. For each aspect-level representation, which is a h_1-dimensional vector of latent factors, ρ percent of this vector is dropped out at random during the training phase. Additionally, we apply L_2 regularization to the user and item biases in Equation (8).

4 EXPERIMENTS

We evaluate our proposed model against several state-of-the-art baseline methods using publicly available datasets from *Yelp* and *Amazon*. In this section, we describe the datasets used, introduce the baseline methods, elaborate on the experimental setup, and present the experimental results.

4.1 Datasets

For *Yelp*, we use the latest version (*Round 11*) for the Yelp Dataset Challenge[6], which contains ratings and reviews for local businesses across 4 countries. As for *Amazon*, we use the Amazon Product Reviews[7] from [15, 27] which has already been organized into **24** individual product categories.

For the Yelp dataset and 3 of the larger datasets from Amazon (i.e. Books, Electronics, and Clothing, Shoes & Jewelry), we randomly sub-sampled 5, 000, 000 user-item interactions for the experiments. Following which, similar to [7, 8, 33, 44], we randomly partitioned each of these **25** datasets into training, validation, and testing sets using the ratio 80:10:10. Following [7, 26, 44], we directly use these datasets as they are. Specifically, we have chosen not to adopt the *'5-core setting'* used in [8, 10, 33], whereby there are at least 5 ratings/reviews for each user and item, as it trivializes the problem of data sparsity which is inevitable in real-world recommendation systems. Table 2 shows the statistics of the datasets used.

Table 2: Statistical details for the datasets[8]

Dataset	Users	Items	Ratings
Amazon Instant Video	348,665	22,083	499,667
Apps for Android	1,135,316	56,841	2,424,812
Automotive	710,163	279,269	1,193,219
Baby	448,895	59,005	823,549
Beauty	1,014,152	224,878	1,794,288
Books	2,370,327	1,068,230	5,000,000
CDs & Vinyl	1,338,741	445,885	3,454,125
Cell Phones & Accessories	1,886,723	284,794	3,014,598
Clothing, Shoes & Jewelry	2,417,497	926,060	5,000,000
Digital Music	399,571	225,461	725,103
Electronics	2,586,767	362,819	5,000,000
Grocery & Gourmet Food	642,408	150,567	1,151,829
Health & Personal Care	1,546,374	229,078	2,638,255
Home & Kitchen	2,118,130	368,247	3,800,692
Kindle Store	1,189,641	394,742	2,944,055
Movies & TV	1,765,998	187,426	4,241,131
Musical Instruments	280,758	74,731	433,834
Office Products	749,514	116,666	1,069,322
Patio, Lawn And Garden	588,559	95,824	853,064
Pet Supplies	624,250	93,917	1,103,110
Sports And Outdoors	1,667,978	425,034	2,887,105
Tools & Home Improvement	1,012,104	232,744	1,693,910
Toys & Games	1,127,969	294,840	1,998,854
Video Games	689,357	47,562	1,177,239
Yelp (2018)	1,144,046	174,013	5,000,000

4.2 Baseline Methods

We compare our proposed method against 3 state-of-the-art baseline methods which utilize review information to improve the overall recommendation performance.

(1) Deep Cooperative Neural Networks (**DeepCoNN**) [44]: This is a state-of-the-art neural recommendation model which derives latent user and item representations from their corresponding reviews using a convolutional architecture. The user and item representations are concatenated and used as the input to a Factorization Machine (**FM**) [31] for the overall rating prediction. It has been shown from extensive empirical evaluations that DeepCoNN far outperforms *classic* recommendation methods such as Matrix Factorization (**MF**) [22], Latent Dirichlet Allocation (**LDA**) [6], and Hidden Factors as Topics (**HFT**) [26].

(2) Dual Attention-based Model (**D-Attn**) [33]: Similar to DeepCoNN, D-Attn relies on Convolutional Neural Networks (**CNNs**) to learn the user and item representations. The key difference is that prior to the convolutional layer, D-Attn incorporates local and global attention-based modules for selecting locally and globally informative words from the reviews, respectively. However, instead of a FM, D-Attn simply uses the inner product of the user and item representations for the rating prediction.

(3) Aspect-aware Latent Factor Model (**ALFM**) [10]: ALFM is a state-of-the-art aspect-based recommendation system which does **not** rely on external sentiment analysis tools. The authors designed

[6] https://www.yelp.com/dataset/challenge

[7] http://jmcauley.ucsd.edu/data/amazon/

[8] The average sparsity of these datasets is 99.9985%, and the average number of ratings/reviews per user and item (across all datasets) are 1.91 and 12.12, respectively.

an Aspect-aware Topic Model (**ATM**) to represent each aspect $a \in \mathcal{A}$ as a distribution over latent topics based on the review contents. The output from ATM is then combined with ALFM, which associates latent factors with the same set of aspects $\mathcal{A}$ by using the MF approach on the ratings.

It should be noted that all three baseline methods have been proposed *very recently*, and amongst them, have been shown to outperform many other highly competitive recommendation methods [7, 24, 35, 38, 39].

4.3 Experimental Setup

First, all reviews are tokenized using NLTK[9] and we retain the 50,000 most frequent words to be used as the vocabulary $\mathcal{V}$ for each dataset.

For ALFM, we use the code provided by the authors, and follow the hyperpameter settings and optimization method as reported in the paper. Both the number of aspects and latent topics used in ALFM are set to 5. Although [10] only uses 5 latent factors for their model comparison, their hyperparameter study found that more latent factors generally leads to better performance. As such, we used the validation set to select the optimal number of latent factors amongst $\{5, 10, 15, 20, 25\}$ for each dataset.

We implemented the neural recommendation models, i.e. DeepCoNN, D-Attn, as well as our proposed method, using PyTorch[10]. We set the length of input user and item documents, i.e. $|D_u|$ and $|D_i|$, to 500. Our model and DeepCoNN use 300-*d* word embeddings trained on Google News [28], while D-Attn uses 100-*d* word embeddings trained on Wikipedia using *GloVe* [29] (We tried using the same 300-*d* embeddings for D-Attn, but it consistently degrades its performance across multiple datasets). We reuse the settings reported in [33, 44] for hyperpameters such as the number and size of convolutional filters, the number of factors used for the fully connected layers, and the activation functions. For DeepCoNN, we set the dropout rate to 0.5 and the number of factors used in the FM as 10, based on a grid search using the validation sets as these values were not specified in the paper. For fair comparison with ALFM, we use the same number of aspects in our model, i.e. $|\mathcal{A}| = K = 5$. Other hyperparameters for ANR, such as the width of the local context window c, number of latent factors h_1, h_2, and dropout rate ρ are set as 3, 10, 50, and 0.5, respectively. All 3 neural models are trained using Adam [21], using an initial learning rate of 0.002, a batch size of 128 and the MSE loss.

Following [33, 44], we use the standard Mean Squared Error (**MSE**) as the evaluation metric. All the experiments are repeated 5 times, and we report the (average) test MSE obtained when the validation MSE is the lowest.

4.4 Results and Discussion

Table 3 shows the results from our experiments on all 25 datasets. We observe that ANR achieves statistically significant improvement over all 3 state-of-the-art baseline methods, based on the paired sample t-test using results from 5 separate runs for each model.

Next, we note that aspect-aware recommendation methods such as ALFM and ANR consistently outperforms DeepCoNN and D-Attn. We believe that this can be attributed to the fact that DeepCoNN and D-Attn *'compresses'* the user (and item) documents into a single representation (i.e. vector), and consequently, the only *'interaction'* between users and items occurs at the prediction layer, i.e. when using the user and item representations for predicting the overall rating. In other words, they are unable to capture the multi-faceted decision making process involved in these user-item interactions. Both DeepCoNN and D-Attn have a similar model architecture due to their use of the convolutional layer as the encoder, and it may seem like D-Attn would perform better with its additional local and global attention-based modules. However, D-Attn was previously evaluated using the much denser 5-core setting [33], and it seems to underperform due to the data sparsity which is evident in our experimental setup.

Finally, although ALFM attempts to utilize the review contents in their framework, they do so using a topic modeling approach. One major drawback is that the proposed Aspect-aware Topic Model (**ATM**) of ALFM does not consider the rating information when inferring the user and item preferences from the reviews; and the review contents are not utilized when ALFM learns the latent user and item representations using the MF approach. Put differently, unlike our proposed method, ALFM uses the review contents and rating information ***separately***.

5 MODEL ANALYSIS

In this section, we examine the effects of key hyperparameters on the model performance. Furthermore, we provide a glimpse of the inner workings of our model via a qualitative analysis of the learned aspects and an ablation study.

5.1 Parameter Sensitivity

5.1.1 Number of Aspects. Figure 3 illustrates the effect of varying the number of aspects between 2 to 8 for our model across multiple datasets. We notice that the optimal number of aspects varies across the different datasets, and most likely depends on the characteristics of the review contents for any given dataset. In general, we observe that reasonably good performance can be obtained using around 4 to 6 aspects. Additionally, we hypothesize that changing the total number of aspects only affects the granularity of each individual aspect, i.e. numerous *fine-grained aspects* versus a handful of *broader aspects*. As such, varying the number of aspects (within a reasonable range) would have rather little impact on the overall model performance.

5.1.2 Number of Factors for h_1 and h_2. We investigate the model's sensitivity to the number of factors used for h_1 and h_2. The 3-D figures in Figure 4 shows the performance of our model by varying h_1 from 5 to 50 and h_2 from 10 to 100, for different datasets. Recall that h_1 determines the number of latent factors used for the aspect-level user and item representations (i.e. $|\mathbf{p}_{u,a}|$ & $|\mathbf{q}_{i,a}|$), while h_2 defines the size of the hidden layers used for estimating the user and aspect importance (i.e. $\boldsymbol{\beta}_u$ & $\boldsymbol{\beta}_i$) based on the affinity matrix $\mathbf{S}$.

First, it does not require a large number of latent factors to encode the user and item representations at an aspect-level and the model performance does not improve when h_1 is greater than 15. However, as shown in Figure 4(a), the performance may degrade if insufficient latent factors are used for $\mathbf{p}_{u,a}$ and $\mathbf{q}_{i,a}$. Next, we find that the number of hidden factors used for estimating the aspect

[9] https://www.nltk.org/
[10] https://pytorch.org/

Table 3: Comparison with state-of-the-art baseline methods in terms of the Mean Squared Error (The best result for each dataset is indicated in bold). All reported improvements over baseline methods are statistically significant with *p-value* < 0.01 based on the paired sample t-test.

Dataset	D-Attn	DeepCoNN	ALFM	ANR	Improvement (%)		
	(a)	(b)	(c)	(d)	(d) vs. (a)	(d) vs. (b)	(d) vs. (c)
Amazon Instant Video	1.213	1.178	1.075	**1.009**	16.83	14.36	6.13
Apps for Android	1.637	1.593	1.555	**1.412**	13.73	11.34	9.14
Automotive	1.411	1.349	1.257	**1.188**	15.76	11.91	5.43
Baby	1.507	1.442	1.359	**1.258**	16.51	12.73	7.44
Beauty	1.609	1.566	1.466	**1.386**	13.89	11.48	5.46
Books	1.122	1.089	1.055	**0.976**	12.94	10.30	7.43
CDs & Vinyl	1.014	0.980	0.956	**0.914**	9.93	6.81	4.46
Cell Phones & Accessories	2.083	2.040	1.787	**1.689**	18.92	17.23	5.50
Clothing, Shoes & Jewelry	1.491	1.430	1.316	**1.266**	15.09	11.48	3.78
Digital Music	0.775	0.749	0.725	**0.688**	11.22	8.12	5.07
Electronics	1.744	1.659	1.563	**1.445**	17.10	12.89	7.50
Grocery & Gourmet Food	1.386	1.345	1.284	**1.187**	14.42	11.76	7.57
Health & Personal Care	1.612	1.545	1.466	**1.356**	15.91	12.23	7.49
Home & Kitchen	1.575	1.508	1.443	**1.317**	16.38	12.69	8.76
Kindle Store	0.949	0.905	0.870	**0.834**	12.08	7.81	4.10
Movies & TV	1.246	1.207	1.193	**1.112**	10.75	7.88	6.80
Musical Instruments	1.224	1.160	1.072	**1.034**	15.51	10.81	3.49
Office Products	1.650	1.569	1.474	**1.337**	18.98	14.79	9.30
Patio, Lawn & Garden	1.696	1.622	1.510	**1.403**	17.30	13.51	7.09
Pet Supplies	1.628	1.565	1.485	**1.377**	15.41	12.05	7.28
Sports & Outdoors	1.354	1.300	1.221	**1.137**	16.04	12.55	6.86
Tools & Home Improvement	1.474	1.429	1.348	**1.230**	16.51	13.93	8.74
Toys & Games	1.298	1.227	1.131	**1.075**	17.16	12.34	4.88
Video Games	1.533	1.498	1.383	**1.292**	15.72	13.72	6.57
Yelp (2018)	1.691	1.669	1.614	**1.527**	9.68	8.49	5.42
Average	1.437	1.385	1.304	**1.218**	14.95	11.73	6.47

importance has a much lesser impact on the overall performance, and our choice of setting h_2 to 50 should suffice for most datasets.

Table 4: Top 10 words for each aspect in the *Video Games* dataset. The 'aspect labels' are added based on our interpretation of that aspect.

Price	Family	Negative	Gameplay	Graphics
works	son	bad	lot	bought
recommend	new	little	hours	pretty
well	highly	horrible	bit	still
buy	story	waste	couple	graphics
bought	favorite	hard	characters	much
awesome	part	boring	stars	think
price	character	terrible	course	work
loves	daughter	frustrating	minutes	recommend
worth	controller	difficult	side	cool
purchase	characters	disappointed	fan	nice

5.2 Qualitative Analysis of Learned Aspects

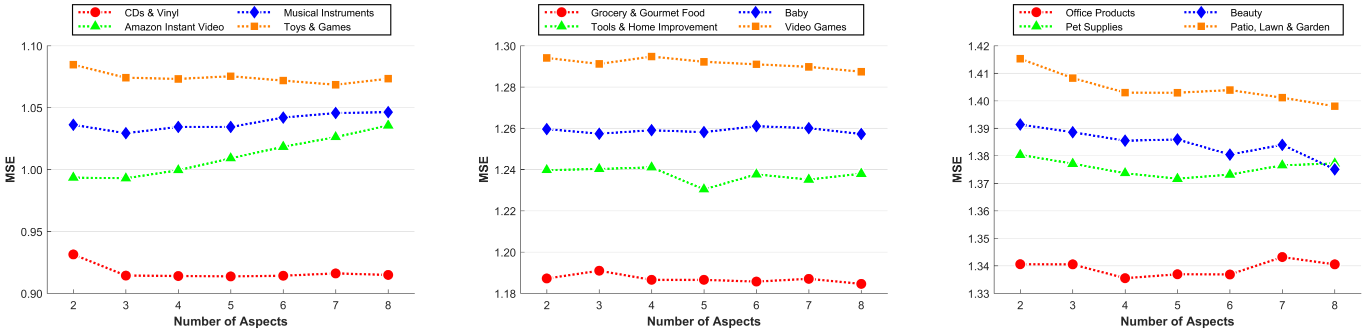
In Section 3.3, we described the process of obtaining the aspect-level representations by learning to attend to a subset of aspect-related words within the corresponding document. The soft attention vector $\mathbf{attn}_{u,a}$ in Equation (3) can also be viewed as a probability distribution over the vocabulary $\mathcal{V}$, for a user $u \in \mathcal{U}$ and an aspect $a \in \mathcal{A}$. As such, we can calculate the *importance* of each word $z \in \mathcal{V}$ w.r.t. the user u and an aspect a as follows:

$$\psi_{z,u,a} = \sum_{i=1}^{|D_u|} \mathbf{attn}_{u,a}[i] \ \text{ if } \ (D_u[i] = z) \tag{9}$$

where $D_u[i]$ refers to the i-th word in the user document. The *importance* of word z for aspect a can then be computed as:

$$\psi_{z,a} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \psi_{z,u,a} \ + \ \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \psi_{z,i,a} \tag{10}$$

Following [26], the 'background' distribution of a word z is defined as $b_z = (\sum_{a \in \mathcal{A}} \psi_{z,a}) \,/\, |\mathcal{A}|$, and we semantically represent each aspect a using its top words based on $(\psi_{z,a} - b_z)$. The aspects learned by our model for the ***Video Games*** dataset are shown in Table 4. We find that each aspect does cover a rather specific and meaningful facet of item properties for the particular domain, and reflects the *different factors* that contribute to the overall rating of these user-item interactions. Given that the optimal number of aspects can be rather different for each dataset, the quality of these learned aspects could potentially improve if we consider a different number of aspects.

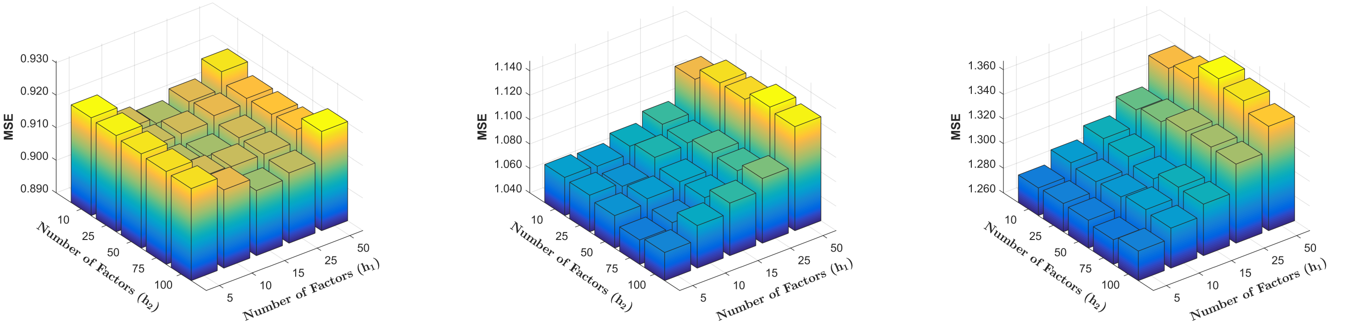


(a) CDs & Vinyl; Amazon Instant Video; Musical Instruments; Toys & Games

(b) Grocery & Gourmet Food; Tools & Home Improvement; Baby; Video Games

(c) Office Products; Pet Supplies; Beauty; Patio, Lawn & Garden

Figure 3: Effect of the Number of Aspects



(a) CDs & Vinyl

(b) Toys & Games

(c) Video Games

Figure 4: Effect of the Number of Factors for h_1 and h_2

5.3 Ablation Study

We perform an ablation study to analyze how different components in our proposed model contribute to the overall performance and hopefully, justify some of our architectural decisions. The *'baseline'* of this discussion refers to the complete model as described in Section 3, using the hyperparameter settings stated in Section 4.3, and we compare it with its five variants:

- **Simplified Model:** Instead of modeling the aspect-level interactions, the aspect-level user (and item) representations are concatenated and put through a hidden layer to obtain the final user (and item) representation. Similar to the baseline methods DeepCoNN and D-Attn, the only interaction between users and items occurs at the final prediction layer, i.e. when using the user and item embeddings to derive the overall rating.
- **No Pre-training:** We forgo the pre-training phase for Θ_{ARL}, i.e. the set of parameters for *Aspect-based Representation Learning* layer, to validate its effectiveness.
- **Shared Projection Layer:** Rather than having aspect-specific projection matrices, we constrain the model by having only a single projection matrix which is shared across all aspects. Basically, each word has the exact same representation across all aspects, and we make use of this model variant to verify **Intuition 2**.
- **Uniform Aspect Importance:** We replace $\boldsymbol{\beta}_{u,a} \cdot \boldsymbol{\beta}_{i,a}$ in Equation (8) with $1/K$, i.e. all aspects are assumed to be *equally important.*
- **Main Diagonal:** The main diagonal of the affinity matrix S obtained using Equation (5) is used to replace $\boldsymbol{\beta}_{u,a} \cdot \boldsymbol{\beta}_{i,a}$. Note that the main diagonal only captures the relationship between corresponding pairs of aspects, i.e. aspect a (User) and aspect a (Item) for each $a \in \mathcal{A}$.

The results of the ablation study for the ***Toys & Games*** and ***Video Games*** datasets are shown in Table 5. First, we observe that the lack of aspect-level interactions in the *Simplified Model* leads to large performance degradation on both datasets. For the *Aspect-based Representation Learning* layer, we find that the pre-training phase does provide a better starting point for learning the user and item aspect importance. Additionally, allowing variations in the word representations through the aspect-specific projection layer leads to better overall performance, supporting our **Intuition 2**. Finally, the results from the last 2 model variants highlight the need for dynamically adapting the user and item aspect importance for each user-item pair.

Table 5: Comparison of the model variants for the *Toys & Games* and *Video Games* datasets

Setup	Toys & Games	Video Games
Baseline	1.069	1.278
Simplified Model	1.173	1.495
No Pre-training	1.123	1.354
Shared Projection Layer	1.122	1.349
Uniform Aspect Importance	1.106	1.310
Main Diagonal	1.108	1.315

6 CONCLUSION

We have presented a novel **A**spect-based **N**eural **R**ecommender (**ANR**), which includes an aspect-aware representation learning component and an aspect importance estimator, that are based on the ideas of neural attention and co-attention, respectively. Experimental results have shown that ANR achieves statistically significant improvement over existing state-of-the-art recommendation systems. One interesting future direction would be to extend ANR into a domain-independent framework, which will be able to handle multiple categories simultaneously, by incorporating either transfer learning or multi-task learning.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their meticulous reviews and invaluable suggestions. This work is supported in part by the MOE Tier-2 grant (MOE2016-T2-1-137) and the MOE Tier-1 grant (RG31/17) awarded by the Ministry of Education, Singapore. This work is also supported in part by the National Natural Science Foundation of China under Grant No.: 61772537.

REFERENCES

[1] Amjad Almahairi, Kyle Kastner, Kyunghyun Cho, and Aaron Courville. 2015. Learning Distributed Representations from Reviews for Collaborative Filtering. In *RecSys '15*. ACM, New York, NY, USA, 147–154.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014). http://arxiv.org/abs/1409.0473

[3] Trapit Bansal, David Belanger, and Andrew McCallum. 2016. Ask the GRU: Multi-task Learning for Deep Text Recommendations. In *RecSys '16*. ACM, 107–114.

[4] Yang Bao, Hui Fang, and Jie Zhang. 2014. TopicMF: Simultaneously Exploiting Ratings and Reviews for Recommendation. In *AAAI '14*. AAAI Press, 2–8.

[5] Konstantin Bauman, Bing Liu, and Alexander Tuzhilin. 2017. Aspect Based Recommendations: Recommending Items with the Most Valuable Aspects Based on User Reviews. In *KDD '17*. ACM, 717–725.

[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *JMLR* 3 (March 2003), 993–1022.

[7] Rose Catherine and William Cohen. 2017. TransNets: Learning to Transform for Recommendation. In *RecSys '17*. ACM, New York, NY, USA, 288–296.

[8] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *WWW '18*. 1583–1592.

[9] Xu Chen, Zheng Qin, Yongfeng Zhang, and Tao Xu. 2016. Learning to Rank Features for Recommendation over Multiple Categories. In *SIGIR '16*. ACM, New York, NY, USA, 305–314.

[10] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *WWW '18*. 639–648.

[11] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very Deep Convolutional Networks for Text Classification. In *EACL '17*. ACL, 1107–1116.

[12] Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J. Smola, Jing Jiang, and Chong Wang. 2014. Jointly Modeling Aspects, Ratings and Sentiments for Movie Recommendation (JMARS). In *KDD '14*. ACM, New York, NY, USA, 193–202.

[13] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why Does Unsupervised Pre-training Help Deep Learning? *JMLR* 11 (March 2010), 625–660.

[14] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[15] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW '16*. 507–517.

[16] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM '15*. ACM, New York, NY, USA, 1661–1670.

[17] Minqing Hu and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *KDD '04*. ACM, New York, NY, USA, 168–177.

[18] Rie Johnson and Tong Zhang. 2017. Deep Pyramid Convolutional Neural Networks for Text Categorization. In *ACL '17*. ACL, 562–570.

[19] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. 2016. Convolutional Matrix Factorization for Document Context-Aware Recommendation. In *RecSys '16*. ACM, New York, NY, USA, 233–240.

[20] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP '14*. ACL, 1746–1751.

[21] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). http://arxiv.org/abs/1412.6980

[22] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37.

[23] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder. In *CIKM '15*. ACM, New York, NY, USA, 811–820.

[24] Guang Ling, Michael R. Lyu, and Irwin King. 2014. Ratings Meet Reviews, a Combined Approach to Recommend. In *RecSys '14*. ACM, New York, NY, USA, 105–112.

[25] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical Question-image Co-attention for Visual Question Answering. In *NIPS '16*. Curran Associates Inc., USA, 289–297.

[26] Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In *RecSys '13*. ACM, New York, NY, USA, 165–172.

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR '15*. ACM, New York, NY, USA, 43–52.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *NIPS '13*. 3111–3119.

[29] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP '14*. 1532–1543.

[30] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion Word Expansion and Target Extraction Through Double Propagation. *COLING* 37, 1 (March 2011), 9–27.

[31] Steffen Rendle. 2010. Factorization Machines. In *ICDM '10*. IEEE Computer Society, Washington, DC, USA, 995–1000.

[32] Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *EMNLP '15*. ACL, 379–389.

[33] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable Convolutional Neural Networks with Dual Local and Global Attention for Review Rating Prediction. In *RecSys '17*. ACM, New York, NY, USA, 297–305.

[34] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *JMLR* 15 (2014), 1929–1958.

[35] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-boosted Latent Topics: Understanding Users and Items with Ratings and Reviews. In *IJCAI '16*. AAAI Press, 2640–2646.

[36] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. Multi-Pointer Co-Attention Networks for Recommendation. In *KDD '18*. ACM, 2309–2318.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS '17*. Curran Associates, Inc., 5998–6008.

[38] Chong Wang and David M. Blei. 2011. Collaborative Topic Modeling for Recommending Scientific Articles. In *KDD '11*. ACM, New York, NY, USA, 448–456.

[39] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *KDD '15*. ACM, New York, NY, USA, 1235–1244.

[40] Yao Wu and Martin Ester. 2015. FLAME: A Probabilistic Model Combining Aspect Based Opinion Mining and Collaborative Filtering. In *WSDM '15*. ACM, New York, NY, USA, 199–208.

[41] Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic Coattention Networks For Question Answering. *CoRR* abs/1611.01604 (2016). http://arxiv.org/abs/1611.01604

[42] Yongfeng Zhang. 2015. Incorporating Phrase-level Sentiment Analysis on Textual Reviews for Personalized Recommendation. In *WSDM '15*. ACM, New York, NY, USA, 435–440.

[43] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit Factor Models for Explainable Recommendation Based on Phrase-level Sentiment Analysis. In *SIGIR '14*. ACM, New York, NY, USA, 83–92.

[44] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In *WSDM '17*. ACM, 425–434.