# Aspect-aware Asymmetric Representation Learning Network for Review-based Recommendation


1st Hao Liu
*School of Computer Science and Technology*
*Chongqing University of Posts and Telecommunications*
Chongqing, China
liuhao@cigit.ac.cn

2nd Hezhe Qiao
*Chongqing Institute of Green and Intelligent Technology*
*Chinese Academy of Sciences*
Chongqing, China
qiaohezhe@cigit.ac.cn

3rd Xiaoyu Shi*
*Chongqing Institute of Green and Intelligent Technology*
*Chinese Academy of Sciences*
Chongqing, China
xiaoyushi@cigit.ac.cn

4th Mingsheng Shang
*Chongqing Institute of Green and Intelligent Technology*
*Chinese Academy of Sciences*
Chongqing, China
msshang@cigit.ac.cn



2022 International Joint Conference on Neural Networks (IJCNN) | 978-1-7281-8671-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IJCNN55064.2022.9892559



***Abstract*—Recently, user-provided reviews have been identified as an essential resource to improve user and item representation in recommender systems. Previous methods focus on the review-based recommender typically leverages symmetric networks to process user and item reviews. However, in reality, these two sets of reviews are markedly different: a user's reviews reflect the experience of buying diverse items and show their heterogeneous interests. In contrast, an item's reviews emphasize the quality of the specific item. Thus an item's reviews are usually homogeneous. This paper seeks to explore the aspect of review difference in the review-based recommendation framework. We propose a novel asymmetric neural network model that accurately learns the user and item representation by identifying this critical difference. We focus on capturing the dynamic change of user interest for the user-aspect reviews via modeling the temporal information into the conventional neural network(CNN). On the other side, we try to identify a specific item's essential yet essential features by utilizing the self-attention neural network. Finally, a factorization machine (FM) is adopted to finish the rating prediction task, where the user and item IDs are encoded as supplementary review embedding. We conduct comprehensive experiments on four Amazon datasets, and the experimental results show that our proposed model consistently outperforms several state-of-the-art methods.**



***Index Terms*—review-based recommendation, asymmetric network, sequential recommendation, rating prediction**


## I. Introduction

With the advent of the 5G era, everyone can easily become a producer of information content through the Internet, especially in the field of new media, which leads to the problem of information overload. The use of recommender systems is to solve the problem of information overload and reduce the time for users to retrieve information by recommending content that users are interested in [1].

The ID of the user (item) and rating are used from a rating matrix in Traditional recommender systems, but the rating matrix is sparse in real life. There is rich semantic information in users and items reviews, and the rational use of comments is beneficial to solving the problem of data sparsity. For example, Kim *et al.* [2] using text information as auxiliary side information alleviates the sparsity problem of the scoring matrix. Dong *et al.* [3] solves the difficulty posed by the sparsity of the collaborative filtering (CF) method by incorporating a review. Therefore, there has been a series of studies trying to use the potential reviews to improve the performance of recommendations. While people frequently consult such reviews, and reviews can influence their decisions, new research shows that intelligent algorithms can also leverage review information to make more accurate recommendations [4]. If a large amount of semantic information can be used rationally, the detailed information (color, size, material, etc.) in the reviews can be mined more abundantly. The capturing of this

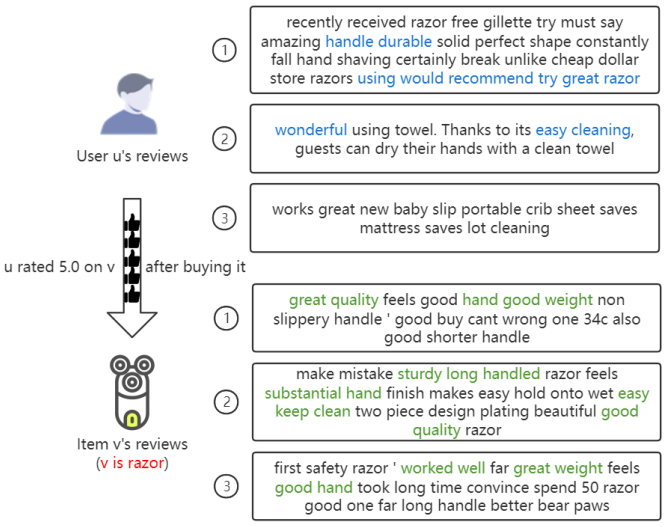


Fig. 1. An example of user's and item's historical reivews. After purchasing item v, user u gave it a 5.0 rating.


* Xiaoyu Shi is corresponding author.




978-1-7281-8671-9/22/$31.00 ©2022 IEEE

fine-grained information is conducive to improving the quality of recommendations [5–8].

Recent studies have shown that reviews have a significant impact on improving the quality of recommender systems. Most studies predict the rating of a user-item pair that has not appeared before by organizing each user's reviews into a review set while associating each item with other users' reviews on the item. For example, Zheng *et al*. [5] proposed DeepCoNN, which relies on factorization machines (FM) to predict ratings and uses convolutional neural networks to learn user (item) embeddings. Seo *et al*. [7] proposed D-ATT, which uses a dual attention network to learn to embed and predict the score after the user embedding and the item embedding are multiplied by a dot multiplication operation. The set of reviews was used in these methods as the same long document and processed in parallel using the same network model. However, in real life, a user review set is a collection of user reviews on purchased items, and the items purchased by each user are often different. The item review set is a collection of reviews on the same item purchased by different users. Since they are reviews on the same item, the contents of the reviews are roughly the same. Reviews depend on the properties of the item, and items with different themes usually have different properties. A variety of products purchased by users will correspond to their themes, and different themes will lead to different perspectives of user reviews. On the contrary, the theme of the same item is determined, and thus the reviews of different users on the item are similar. Therefore, user-specific review sets and project-specific review sets are fundamentally different.

Fig. 1. shows several reviews from the Amazon Health and Personal space. The reviews of user $u$ are made on three different items, and the reviews of item $v$ are made by three different users. To more intuitively show the difference between the user review set and the item review set, we use different colors to mark in the figure. For example, $u$'s review 1 and review 2 are all about the similar and $v$ household items, $u$ mentioned aspects such as "handle durable", "easy to clean", which shows that users tend to choose good quality, easy to clean items. At the same time, $v$'s reviews mentioned the "great quality", "handle good weight", "handle long handled" and "easy to clean", these other expressions are in line with the user's preference. Thus, user $u$ may be interested in $v$ and also gave a 5.0 score after purchasing.

To solve the above problems, we proposed an asymmetric aspect-aware network (ATN), a deep learning network based on the aspect-aware recommendation of heterogeneous networks. ATN is characterized by using the attention mechanism to focus on content related to item attributes and using the CNN network to learn user embedding representations. However, the review information contains rich semantic information, the problem that some latent features of users (items) are not encoded by their reviews is often overlooked. We solve this problem by one-hot encoding the user (item) ID. We ran numerous experiments to assess the model's performance. The results of our experiments reveal that our model outperforms even the most advanced methods. Our contribution is as follows:

- We propose an aspect-aware asymmetric neural network for the review-based recommendation, which can learn the user and item representation more accurately.
- We show how to capture user preference by integrating the time information. For the latent features of some user (item) reviews that cannot be encoded, we use MLP to embed the one-hot representation of each user (item) ID.
- We have put ATN to work on Amazon's four data sets. The results of our experiments reveal that our model outperforms even the most advanced methods.

## II. RELATED WORKS

### A. Review-Based in Recommender System

User reviews are used to address the data sparsity problem caused by rating methods. McAuley *et al*. [9] used LDA to discover potential aspects of users and projects from reviews. Ling *et al*. [10] first extracted topics from the reviews and decomposed the score matrix to enhance user and item embedding. Bao *et al*. [11] jointly decomposed the scoring matrix and the packet representation of reviews to infer user and item embeddings. Although these methods use reviews to alleviate the data sparsity brought by only using ratings, they only focus on topic clues in reviews, thus ignoring the semantic content possessed by texts. It helps to model user and project characteristics by considering the order and context of words and sentences in reviews.

As the use of deep learning in the field of NLP has made great progress [12–14], more and more researchers have begun to focus on deep learning methods. For example, Zheng *et al*. [5] used LSTM to extract long-term and short-term dependence information, then used CNN as feature extraction, and finally used dot multiplication and FM to predict the score. Catherine *et al*. [6] the CNN module in DeepCoNN is improved by using a multi-task learning method to learn user and project embedded representation better. Seo *et al*. [7] used the attention mechanism at the word level to obtain information-rich vocabulary. Tay *et al*. [15] fit the interaction between user reviews and item reviews by using a pointer based on common attention, and Wu *et al.* [16] use an attention mechanism at the word level to infer sentence-level representations by constructing a symmetrical hierarchy. However, all of these methods use parallel learning of user and item embeddings without considering a huge difference between user review sets and item review sets, which leads to a decline in the recommendation effect.

### B. Time-aware in Recomender System

The purchase behavior in the past in the user's purchase history cannot truly reflect the current interest of the target user. Based on this, many methods are proposed to use time information to improve the accuracy of the recommendation system. Zimdars *et al*. [17]. mapped a recommendation problem to a time series prediction problem by considering temporal information. Koren *et al*. [18]. propose TimeSVD++,

which incorporates the item feature vector into the biased SVD recommendation model, with the time impact being the same hidden factor vector as the user dimension. However, matrix factorization is limited to the three-dimensional space of the user, score, and time. Recently, Song *et al*. [19] explored the influence of the time window on the training set and used some recent scores as the training set, which is beneficial to improve the accuracy and diversity. Although these methods effectively improve recommendation accuracy, they are uncertain for avoiding outdated items.

## III. OUR APPROACH

In this section, we will take a bottom-up approach to introduce our ATH model. The architecture of ATH is depicted in Fig. 2.

### *A. The Embedding Layer*

In our model, we use reviews with time information as a powerful addition to understanding user behavior. We type all reviews from a user or item into the embedding layer. Let $R = \{r_1, r_2, ..., r_n\}$ denote the set of $p$ reviews. In the embedding layer, each review is transformed from the pre-trained word matrix $W_e$ ($W_e \in \mathbb{R}^{d\times|v|}$) into a word vector matrix by a table lookup operation and generate an index $e_j$ ($e_j \in \mathbb{R}^{|v|}$), where $d$ represents the dimension of the word vector and $|v|$ represents the number of words in the pre-train words data set. We can use a low-dimensional dense vector $w_j$ ($w_j \in \mathbb{R}^d$) to represent the $j_{th}$ word, where d represents the dimension of the word vector. As a result, $w_j$ can be expressed as:

$$w_j = e_j \times W_e \tag{1}$$

Finally, a whole review such as $i_{th}$ review $r_i \in \mathbb{R}^{q\times d}$ can be represented as:

$$r_i = w_1 \oplus w_2 \oplus \cdots \oplus w_q, \tag{2}$$

where $\oplus$ is the concatenation operator. As a result, the review set $R$ is encoded as $R = [r_1, r_2, \ldots, r_n]$.

### *B. The Semantic Extracting Layer*

In the extraction layer, in order to better learn user preferences and item attributes, we input the user review set $R_u$ and item review set $R_i$ respectively into two different branches of an asymmetric parallel neural network. we treat the words in each review as sequence data and use LSTM to efficiently process this textual information. Benefit from the tremendous progress LSTM has made in processing sequence data, especially Bi-directional LSTM (BiLSTM). It can effectively capture the forward and backward information in the sequence.

The LSTM can change the flow of information through some gates:forgetting gate $z^f$, input gate $z^i$, and output gate $z^o$, The formula of LSTM is as follows:

$$z^i = \sigma(W_i \times [h_{k-1}, x_k] + b_i), \tag{3}$$

$$z^f = \sigma(W_f \times [h_{k-1}, x_k] + b_f), \tag{4}$$

$$C_k = z^f \odot C_{k-1} + z^i \odot \phi(W_c \times [h_{k-1}, x_k] + b_c), \tag{5}$$

where $C_k$ is the $k_{th}$ LSTM unit's cell status, $x_k$ and $h_{k-1}$ denote current input and the prior hidden state. Then, depending on the content preserved by the cell state $C_k$ , the output content $z^o$ is identified, and the material stored in the cell state us is selectively output:

$$z^o = \sigma(W_o \times [h_{k-1}, x_k] + b_o), \tag{6}$$

The hidden state of the time step $k_{th}$ step after the LSTM unit's internal update is:

$$h_k = z^o \odot \phi(C_k) \tag{7}$$

where $W_o, W_i, W_f \in \mathbb{R}^{D\times D}$ are trainable parameters, and D is the dimension of hidden layer and input embedding in LSTM. In the formulae above, $\phi$ and $\sigma$ represent the activation function that is tanh function and sigmoid function, and $\odot$ denotes the elementwise product operation.

The low-level LSTM states represent syntactic characteristics of words, while high-level states capture context-dependent aspects of word meaning. The forward layer's forward computation and the backward layer's inverse calculation are combined to produce $z_k^o$:

$$\overrightarrow{h}_k = f(w_1 x_k + w_2 \overrightarrow{h}_{k-1}), \tag{8}$$

$$\overleftarrow{h}_k = f(w_3 x_k + w_4 \overleftarrow{h}_{k-1}), \tag{9}$$

$$z_k^o = g(w_5 \overrightarrow{h}_k + w_6 \overleftarrow{h}_k), \tag{10}$$

where $\overleftarrow{h}_k$ and $\overrightarrow{h}_k$ denote the backward and forward LSTM hidden states at the $k_{th}$ step. In the BiLSTM, the hidden state of the time step $k$ is updated as:

$$h_k = \left[\overrightarrow{h}_k, \overleftarrow{h}_k\right]. \tag{11}$$

A large amount of review information in ***R*** is transformed into rich semantic information after being processed by BiLSTM, denoted as $Y = [y_1, y_2, ..., y_n], Y \in \mathbb{R}^{p\times m}$, where $m$ is the size of a review embedding, $p$ is the length of the input review sequence from a user or item. Then, We use a CNN composed of a convolution layer and a pooling layer to further extract user preferences from Y, and use the attention mechanism to extract content related to item attributes from Y.

### *C. The Dynamic Feature Learning Layer*

Considering that user preferences will shift over time, we have introduced a time dimension so that our model can better solve this problem. The set of review time in ***R*** is denoted by $T = \{t_1, t_2, ..., t_n\}$. In addition, the time period between the $i_{th}$ and $(i+1)_{th}$ reviews was standardized. As a result, the $i_{th}$ review's time information is encoded as follows:

$$t_i = \frac{t_i - \min(T)}{\max(T) - \min(T)}, \tag{12}$$

where $T$ is the collection of intervals between two adjacent interactions. As a result, the $y_i$ with time information can be written as:

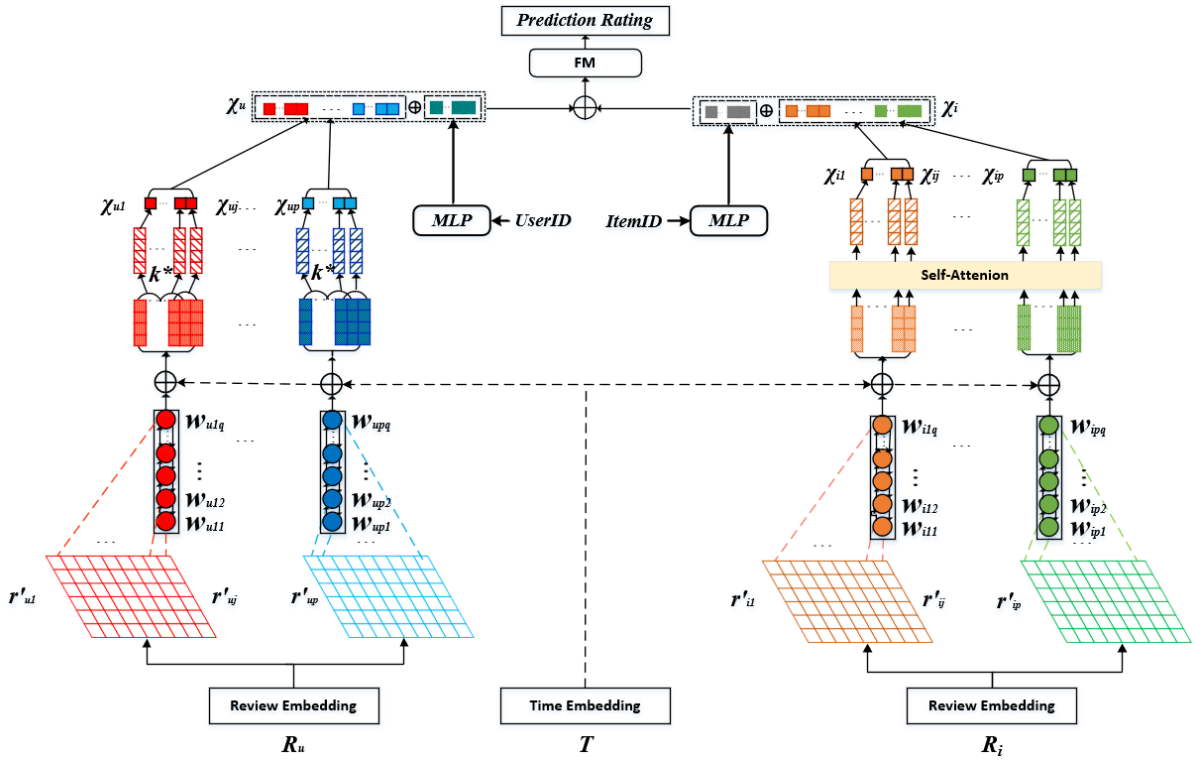


Fig. 2. illustrates the architecture of ATN.

$$y_i^{'} = y_i \oplus t_i, \tag{13}$$

where the $\oplus$ is the concatenation operator.

*1) CNN:* A review sequence of a user or item from the result of the semantic extracting layer can be as: $Y^{'} = \left[y_1^{'}, y_2^{'}, ..., y_n^{'}\right], Y^{'} \in \mathbb{R}^{p \times m}$, where $p$ is the length of the input review sequence, $m$ is the size of a review embedding and time information after concatenation operator. $X_{1:n}^{i}$ represents the $i_{th}$ review containing the $q$ words. To acquire numerous features, we employ different convolution filters $k$. For the $i_{th}$ review, the outcome of the $j_{th}$ convolution filter operation is as follows:

$$l_i = \phi(K_i * X_{1:n}^{i} + b_i), \tag{14}$$

where $K_i$ is a convolution filter,$*$ represents the convolution operation, $b_i$ is a bias term, and $\phi$ is an activation function.The more significant feature can be deduced as follows:

$$z_j = \max(L_j), \tag{15}$$

$$Z_i = [z_1, z_2, ..., z_h], \tag{16}$$

where $z_j$ is the result of max pooling from $L_j$, $Z_i$ is the set of one review features, and h is the number of convolutional kernels. Multiple review vectors are connected together and expressed as:

$$\Gamma = Z_1 \oplus Z_2 \oplus ... \oplus Z_p. \tag{17}$$

Finally, we input the result of the max-pooling layer to a fully connected layer.

$$\chi = relu(W \times \Gamma + b), \tag{18}$$

where $W$ is a weight matrix, $b$ is a bias term, and $X \in \mathbb{R}^{o \times 1}$ is a one-dimensional vector. A user representation is represented as $\widetilde{\chi}_u$.

*2) Self-Attention:* We use the dot product attention mechanism to filter out content related to item attributes. The dot product attention is defined as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V, \tag{19}$$

Where $Q$ represents the queries, $K$ the keys and $V$ the values. Intuitively, The weighted total of all values is calculated by the attention layer, with the weight between query $i$ and value $j$ being related to the interaction between query $i$ and key $j$. When the dimension is high, using the scale factor $\sqrt{d}$ can effectively solve the problem that the inner product value is too large.

In our instance, the embedding $Y^{'}$ is taken as input to self-attention, which is transformed into three matrices by linear projection and fed into the attention layer:

$$S = Attention(Y^{'}W^Q, Y^{'}W^K, Y^{'}W^V), \tag{20}$$

where the projection $W^Q, W^K, W^V \in \mathbb{R}^{m \times m}$. These three parameters can be trained and shared, and thanks to this, the model can have better fitting ability.

Although self-attention can aggregate the embeddings of all previous items with adaptive weights, it is still a linear model in the end. To better capture item attributes, we use a point-to-point two-layer feed-forward network applied to all $S_i$ identically:

$$F_i = FFN(S_i) = relu(S_i W^{'} + b^{'})W^{''} + b^{''}, (21)$$

where $W^{'}$, $W^{''}$ are $m \times m$ matrices and $b^{'}$,$b^{''}$ are m-dimensional vectors. Note that there is no interaction between $S_i$ and $S_j$ $(i \neq j)$, which means that we can still prevent information leaks (from back to front).

The more meaningful feature can be extracted as:

$$z_l = \max(F), \tag{22}$$

Using similar processing as before, we input the result of the max-pooling layer to a fully connected layer.

$$\chi = relu(W \times \Gamma + b), \tag{23}$$

where $W$ is a weight matrix, $b$ is a bias term, and $X \in \mathbb{R}^{o \times 1}$ is a one-dimensional vector. A item representation is represented as $\widetilde{\chi}_i$.

*3) Encoding Latent Rating Patterns:* Although the rich semantic information in the reviews is contained in the embeddings $\widetilde{\chi}_u$ and $\widetilde{\chi}_i$, some potential characteristics of users (items) are not encoded by their reviews, but can be inferred from the rating patterns. We use an MLP to embed a one-hot representation of the ID of each user(item), and acquire an embedding vector $\widehat{\chi}_u$ $(\widehat{\chi}_i)$ for the user (item). It is more effective to capture potential rating patterns by considering this vector that is directly related to the ratings of a user (item). Then, as shown in Fig. 2, we concatenate $\widetilde{\chi}_u$ and $\widehat{\chi}_u$ to acquire the final embedding of a user, $\chi_u = [\widetilde{\chi}_u; \widehat{\chi}_u]$, and concatenate $\widetilde{\chi}_i$ and $\widehat{\chi}_i$ to acquire the final embedding of an item, $\chi_i = [\widetilde{\chi}_i; \widehat{\chi}_i]$.

### D. The Output Layer

We obtain the user representation $\chi_u$ and item representation $\chi_i$ from the previous layer calculations.

$$\gamma = \chi_u \oplus \chi_i, \tag{24}$$

$$\langle v_n, v_m \rangle := \sum_{f=1}^{K} v_n, f \cdot v_m, f, \tag{25}$$

$$y := \omega_0 + \sum_{n=1}^{N} \omega_n \gamma_n + \sum_{n=1}^{N} \sum_{m+1}^{N} \langle v_n, v_m \rangle \gamma_n \gamma_m, \tag{26}$$

where $\omega_0 \in \mathbb{R}$ is the global bias, $\omega_n \in \mathbb{R}^N$ represents the $n_{th}$ variable strength, $\omega_{n,m} := \langle v_n, v_m \rangle$ represents the interaction between the $n_{th}$ and $m_{th}$ variables. The least squares error is used as the loss function of the model:

$$\varsigma = \frac{1}{2} \sum_{i=1}^{M} (\widehat{y}_i - y_i)^2, \tag{27}$$

where $\widehat{y}_i$ is the prediction rating, $y_i$ is the real rating and $M$ is the number of samples. When optimizing the loss function, we add a regularization term to avoid overfitting.

$$\ell = \varsigma + \eta \|\theta\|_2, \tag{28}$$

where $\eta$ is the penalty coefficient, $\theta$ is the set of trainable parameters.

## IV. Expreiments

In this section, we present our experimental setup and experimental results. Our experiments aim to answer the research question.

**RQ1:** Can the asymmetric structure be due to the symmetric structure?

**RQ2:** How much influence does the number of latent factors $\xi$ have on the ATN model?

**RQ3:** Is the ID valid in the ATN network structure? If it works, why?

### A. Datasets

The datasets in our experiments use four datasets from Amazon, while are Digital Music (Mus), Toys and Games (Toys), Musical Instruments (Ins), and Automotive (Auto). The details of the dataset are shown in Table 1.

TABLE I
THE STATISTICS OF AMAZON DATASETS

| Dataset | #users | #items | #reviews | density(%) |
|---|---|---|---|---|
| Mus | 5541 | 3568 | 64706 | 0.3273 |
| Toys | 19412 | 11942 | 167597 | 0.0725 |
| Ins | 1429 | 900 | 10261 | 0.7977 |
| Auto | 2928 | 1835 | 20473 | 0.3809 |

On the test set, mean squared error (MSE) was used to evaluate the model. MSE is defined as follows:

$$MSE = \frac{1}{N} \sum_{n=1}^{N} (\widehat{r}_n - r_n)^2, \tag{29}$$

where $\widehat{r}_n$ is the predicted rating for the $n_{th}$ piece of data, $N$ is the number of samples in the test set, $r_n$ is the true rating of the $n_{th}$ data.

### B. Baseline Models

We compare our proposed model to the following baseline models to illustrate its superiority:

- **DeepCoNN [5]**: Two parallel CNNs are used to process text data, one for user-level reviews and the other for item-level reviews.
- **MPCN [15]**: Simulate the interaction between user reviews and project reviews through pointers based on common attention.
- **TransNet [6]**: A method of combining multi-task learning based on Deepconn-based CNN architecture.
- **DSL-TR [20]**: a temporal-aware recommendation framework based on deep sentiment learning.

### C. Details in our Experiment

Each dataset is sorted by timestamp and divided into three sets: training set, validation set, and test set. In this case, the sorting order is 6:2:2 respectively. To obtain word vectors, we employ pre-trained word embeddings in Google News. The Source code can be found here [1]

### D. Recommendation Performance

Table II presents the recommendation performance of all methods on the four datasets(**RQ1**). By considering the method performance results on all datasets, the directional performance with symmetric network structure is generally worse than that of asymmetric network structure. This may be attributed to the better representation ability of the asymmetric structure for user (item) features.

Our method ATN outperforms all baselines on four different datasets, and it achieves at least 12.1% improvement compared to the DeepConn model. One possible reason is that our model adopts an asymmetric structure so that it has the ability to distinguish the difference between user reviews and item reviews. Secondly, we use MLP to perform one-hot encoding on user (item) ID, which makes up for the problem that latent features of some user reviews cannot be encoded.

### E. Experimental Results

*1) Parameter sensitive:* In order to explore the degree of influence of different parameter settings on the model, we conduct sufficient experiments on each of the four datasets. The results of setting different numbers of latent factors $\xi$ are shown in Fig. 3. From the figure, we can see that setting the latent factor to 50 achieves better results (**RQ2**). Besides, dimension of word embedding d, learning rate $\lambda$, and the batch size are set as 300, 0.002, 100, respectively.

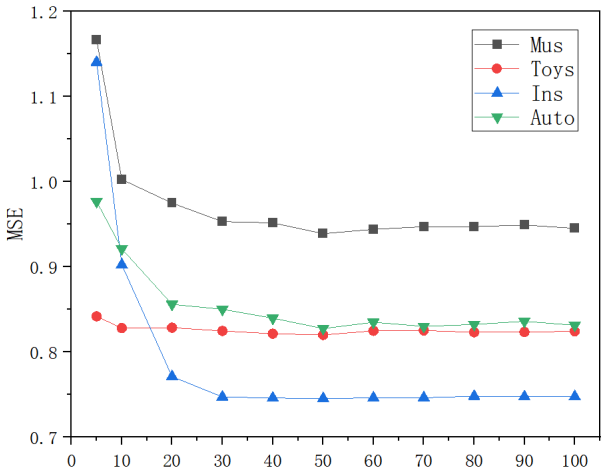


Fig. 3. The performance of ATN with varying $\xi$

*2) Effectiveness of ID:* To verify the ID effect, we conducted several experiments on four datasets from Amazon with three variants of our proposed model.

[1] $https://github.com/xiaoliuhao1211/ATN$

- **ATN-noid**: Only reviews information is entered into the network, regardless of user or item ID.
- **ATN-uid**: Based on ATN-noid, we introduce ID information for encoding latent rating patterns. This variant explores whether ID information can complement latent information in reviews that cannot be encoded.
- **ATN-iid**: Similar to ATN-uid, this model adds an ID feature for items, instead of introducing ID information for user.

The performance of ATN and its three variants in Amazon datasets are summarized in Fig. 4. (**RQ3**). It shows that ATN provides better results than variants in all four datasets. Next, ATN-uid and ATN-iid have a certain effect compared to the ATN with the ID field removed. Finally, the ATN with the id field removed shows the worst performance. The underlying reason is that rich semantic information can be obtained from the user review set and the item review set. Still, some latent properties of users (items) can be inferred from rating patterns that are not represented by their reviews.

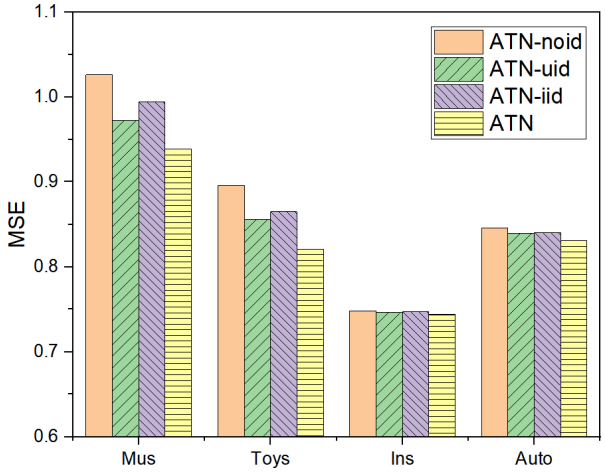


Fig. 4. Comparing variants of the ATN model about ID.

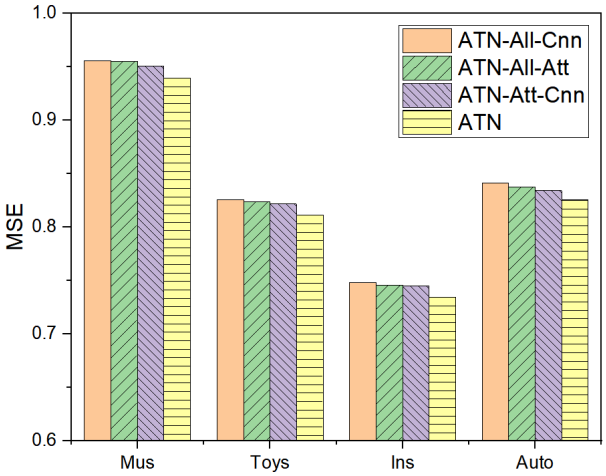


Fig. 5. Comparing variants of the ATN model structure.

TABLE II
PERFORMANCE OF MULTIPLE MODELS IN TERMS OF MSE. BOLD INDICATES THE BEST RESULTS.

| Dataset | Model | | | | | Improvement of ATN(%) |
|---|---|---|---|---|---|---|
| | DeepConn | MPCN | Transnet | DSL-TR | ATN | |
| Mus | 1.094 | 1.051 | 1.087 | 1.013 | **0.939** | 14.16 |
| Toys | 0.934 | 0.948 | 1.035 | 0.892 | **0.821** | 12.09 |
| Ins | 1.003 | 0.992 | 0.983 | 0.755 | **0.744** | 25.82 |
| Auto | 0.998 | 0.924 | 0.966 | 0.841 | **0.827** | 16.73 |

*3) Asymmetric structure:* To verify the effectiveness of the asymmetric structure, we conduct a series of experiments on Amazon's dataset by exchanging the positions of parallel asymmetric modules.

- **ATN-All-Cnn**: The parallel network part uses a symmetric CNN structure
- **ATN-All-Att**: The parallel network part uses a symmetric Attention structure
- **ATN-Att-Cnn**: The parallel network part uses an asymmetric structure, but the user network uses Attention, and the item network uses CNN

Fig. 5. summarizes the performance of ATN and its three variants on the Amazon dataset. It shows that ATN provides better results than its variants in all four datasets. From the results, adopting an asymmetric network structure is more effective than an asymmetric network structure because the reviews of a single user have significantly different themes, while the themes of the reviews of a single item are limited to a single narrow range. In view of the difference between the two, using an asymmetric network structure can more effectively capture user preferences and item attributes.

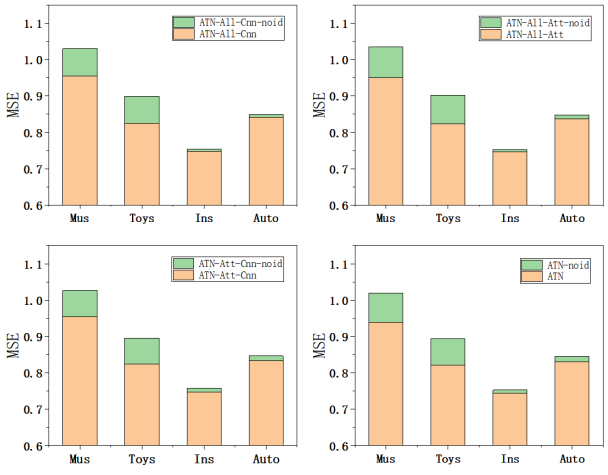


Fig. 6. Further exploration of id and structure

Fig. 6. shows the combination of three variants of ATN structure with and without ID, and the purpose of this is to further explore whether ID and structure have an impact on the model. A striking conclusion from the figure is that both ID and structure have a positive effect on the model. This is due to the addition of IDs, which is beneficial for filling in the hidden features that cannot be captured in the reviews. In addition, an asymmetric structure is added, which can better extract user preferences and item attribute features.

The DeepConn model uses a CNN to learn features from user and item reviews adaptively. Compared with the rating-based recommender system, its effect has a certain improvement. But it cannot capture hidden time-series information from reviews. The TransNet model is improved on the basis of DeepConn, raising the point that review information should not be used in the test set. The DSL-TR model improves on DeepConn's inability to capture hidden time-series information from reviews by explicitly encoding temporal information and merging it with reviews into a symmetric time-aware frame, but as mentioned earlier, the User review set and item review set are heterogeneous, and different models should be used for separate learning. Considering the above situation, we propose a time-aware asymmetric model, which achieves the best results in all benchmark experiments. The potential results can be summarized as follows: firstly, the reviews are sent to BiLSTM for embedded representation and then merged with time-series information, and secondly, they are sent to an asymmetric model for feature learning, specifically using CNN adaptive learning. User review feature, using attention to focus on content related to item attributes. Finally, our model can capture user and item features more effectively through the asymmetric structure. Experiments show that these features can get better results.

## V. Conclusion

In this paper, we propose an aspect-aware asymmetric neural network for review-based recommendation, which can learn the user and item representation more accurately via identifying the different aspects of reviews. The core of our method is to use an asymmetric structure, which combines the temporal information with the review information after the BiLSTM representation learning in the embedding layer, and then uses an asymmetric network structure to separately feature user reviews and item reviews. We explicitly model the temporal information into the user-aspect review network to capture the dynamic changes in user preference. A self-attention network is adopted to dynamically assign the weights to different latent factors to learn the essential features of a specific item. Moreover, user and item IDs are a powerful supplementary to improve the recommendation performance.

Extensive experiments have been conducted on real-world datasets. The results of our experiments reveal that our model outperforms even the most advanced methods.

## ACKNOWLEDGMENT

This research is partly supported by the key cooperation project of Chongqing municipal education commission (HZ2021017), in part by West Light Foundation of The Chinese Academy of Sciences, ‘in part by the independent innovation project of Scientific and technological talents of Chongqing Beibei District .

## REFERENCES

[1] Lü L, Medo M, Yeung C H, et al. Recommender systems[J]. Physics reports, 2012, 519(1): 1-49.

[2] Kim D, Park C, Oh J, et al. Convolutional matrix factorization for document context-aware recommendation[C]//Proceedings of the 10th ACM conference on recommender systems. 2016: 233-240.

[3] Dong R, O'Mahony M P, Schaal M, et al. Sentimental product recommendation[C]//Proceedings of the 7th ACM conference on Recommender systems. 2013: 411-414.

[4] Dong X, Ni J, Cheng W, et al. Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(05): 7667-7674.

[5] Zheng L, Noroozi V, Yu P S. Joint deep modeling of users and items using reviews for recommendation[C]//Proceedings of the tenth ACM international conference on web search and data mining. 2017: 425-434.

[6] Catherine R, Cohen W. Transnets: Learning to transform for recommendation[C]//Proceedings of the eleventh ACM conference on recommender systems. 2017: 288-296.

[7] Seo S, Huang J, Yang H, et al. Interpretable convolutional neural networks with dual local and global attention for review rating prediction[C]//Proceedings of the eleventh ACM conference on recommender systems. 2017: 297-305.

[8] Chen C, Zhang M, Liu Y, et al. Neural attentional rating regression with review-level explanations[C]//Proceedings of the 2018 World Wide Web Conference. 2018: 1583-1592.

[9] McAuley J, Leskovec J. Hidden factors and hidden topics: understanding rating dimensions with review text[C]//Proceedings of the 7th ACM conference on Recommender systems. 2013: 165-172.

[10] Ling G, Lyu M R, King I. Ratings meet reviews, a combined approach to recommend[C]//Proceedings of the 8th ACM Conference on Recommender systems. 2014: 105-112.

[11] Bao Y, Fang H, Zhang J. Topicmf: Simultaneously exploiting ratings and reviews for recommendation[C]//Twenty-Eighth AAAI conference on artificial intelligence. 2014.

[12] Santos C, Tan M, Xiang B, et al. Attentive pooling networks[J]. arXiv preprint arXiv:1602.03609, 2016.

[13] Wang S, Yu M, Chang S, et al. A co-matching model for multi-choice reading comprehension[J]. arXiv preprint arXiv:1806.04068, 2018.

[14] Yang Z, Dai Z, Yang Y, et al. Xlnet: Generalized autoregressive pretraining for language understanding[J]. Advances in neural information processing systems, 2019, 32.

[15] Tay Y, Luu A T, Hui S C. Multi-pointer co-attention networks for recommendation[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2018: 2309-2318.

[16] Wu C, Wu F, Liu J, et al. Hierarchical user and item representation with three-tier attention for recommendation[C]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 2019: 1818-1826.

[17] Zimdars A, Chickering D M, Meek C. Using temporal data for making recommendations[J]. arXiv preprint arXiv:1301.2320, 2013.

[18] Koren Y. Collaborative filtering with temporal dynamics[C]//Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. 2009: 447-456.

[19] Song W, Qiang G, Liu J. Effect of the Time Window on the Personalized Recommendation Algorithm‖[J]. Complex Systems and Complexity Science, 2015, 12(1): 28-32.

[20] Li X, Shang T, Peng D, et al. Deep Sentiment Learning Network for Temporal-aware Recommendation Based on User Reviews[C]//2021 IEEE 6th International Conference on Big Data Analytics (ICBDA). IEEE, 2021: 339-343.