Joint Reason Generation and Rating Prediction for Explainable Recommendation

Jihua Zhu, Yujiao He, Guoshuai Zhao, Xuxiao Bu, Xueming Qian, *Member, IEEE,*

Abstract—Most recommendation systems focus on predicting rating or finding aspect information in reviews to understand user preferences and item properties. However, these methods ignore the effectiveness and persuasiveness of recommendation results. Consequently, explainable recommendation, namely providing recommendation results with recommendation reasons at the same time, has attracted increasing attention of researchers due to its ability in fostering transparency and trust. It is lucky that some E-commerce websites provide a kind of new interaction box called Tips and users can express their comments on items with a simple sentence. This brings us an opportunity to realize explainable recommendation. Under the supervision of two explicit feedbacks, namely rating and textual tips, we can implement a multi-task learning model which can provide recommendation results and generate recommendation reasons at the same time. In this paper, we propose an **E**ncoder-Decoder and **M**ulti-Layer Perception (MLP) based **E**xplainable **R**ecommendation model named **EMER** to simultaneously implement reason generation and rating prediction. Item's title contains significant product-related information and plays an important role in grabbing user's attention, so we fuse it in our model to generate recommendation reasons. Numerous experiments on benchmark datasets demonstrate that our model is superior to the state-of-the-art models.

Keywords—Reason generation, rating prediction, multi-task learning, explainable recommendation

1 Introduction

With the development of e-commerce shopping websites, amounts of products spring up on the Internet and information overload has become very crucial to the success of e-commerce [1], thus recommendation systems emerge as the times require. Early approaches to recommendation systems mostly focus on content-based recommendation or collaborative-filtering based recommendation [2]. Content-based recommendation systems try to recommend items similar to those a given user has liked in the past, whereas collaborative recommendation systems identify users whose preferences are similar to those of a given user and then recommend items the identified users have liked [3].

However, these recommendation systems focus on accuracy of recommended results but ignore their effectiveness and persuasiveness. Recommendations based on explanations can increase user's trust and satisfaction on recommendation systems [4], [5]. Consequently, when recommendation systems implement rating prediction, it will be very useful to generate recommendation reasons at the same time.

It is clear that content-based recommendation systems can identity item content features [3], [4], [6], which users are interested in, so it is intuitive to explain to the users why an item is recommended out of other candidates in this kind of recommendation system. The most well-known usage is "You may like this item because it is good at these features..." used by Amazon[1]. Consequently, finding useful features becomes the key point in this task, while collecting content information in different applications is a time consuming task.

In collaborative recommendation systems, we can provide a given user user-based explanations based on users who have the similar preferences with the user or item-based explanations based on items which are similar to some other items the user bought before [7]. This kind of recommendation system achieves great success when Latent Factor Models (LFM) and Matrix Factorization (MF) were introduced in late 2000's [8], [9]. However, the explanation form of the former is uniform, which can be "Customers who bought this item also bought ...", and the latent factors of the latter could not possess intuitive meanings.

The two kinds of explanations mentioned above are generic and tedious, whose attraction to users is relatively weak. But if we utilize easy-to-understand recommendation reasons explaining why the system recommends this product, there will be great probability that users will accept the recommendation. For example in Figure 1, when recommending this movie to a given user, recommendation systems generate a personalized reason to introduce the feature of the movie according to the user's interests. It seems the personalized explanations are more able to meet the user's preferences and makes users feel friendly.

- *J. Zhu, Y. He, G. Zhao and X. Bu are with the School of Software Engineering, Xi'an Jiaotong University, 710049, Xi'an, China. E-mails: E-mail: zhujh@mail.xjtu.edu.cn; hyjhyj@stu.xjtu.edu.cn; guoshuai.zhao@xjtu.edu.cn; bo951014@stu.xjtu.edu.cn;*
- *X. Qian is with the Ministry of Education Key Laboratory for Intelligent Networks and Network Security and with SMILES LAB, Xi'an Jiaotong University, 710049, Xi'an, China. E-mail: qianxm@mail.xjtu.edu.cn*

1. https://www.amazon.com/

Title : Marvel Studios' Avengers: Endgame (With Bonus)

Rating : ★★★★½

Reason : Surely one of the greatest works of Alan Silvestri.

Fig. 1: One example of recommendation result which contains product's title, predicted rating and generated reason.

In e-commerce shopping websites, user will write some reviews and tips and give numerical ratings after purchasing products. Reviews written by users are always too long and contain a great deal of information, because they describe users' detailed feelings and views on products. So reviews are too redundant to act as recommendation reasons. As for tips, there is always one topic in them which expresses users' feelings and experience on a product, and tips can give other users quick insights for the recommendations. Meanwhile, tips are short and can quickly give users insight into a product [10]. Consequently, we propose a multi-task learning model to model rating through MLP and tip through Encoder-Decoder in this paper. Additionally, reasons we used as explanations in our model, is a general concept and the tips are website specific. Tips can be regarded as a type of reasons and are treated as ground truths of reasons for model training and evaluation in this paper. We set item's title as an important part in the process of reason generation and adapt attention mechanism and coverage mechanism to make full use of the output of the encoder.

Many studies separating the two tasks, namely recommendation and explanation, have achieved good performance [11], [12]. However, in many e-commerce shopping websites, the personalized rating and the personalized reason are quite relevant to each other. Users give high ratings generally relying on some good experiences that may be shown in the personalized reasons. The personalized reason can be regarded as an extra supervised information for the rating prediction model. The personalized reason can be regarded as an extra supervised information for the rating prediction model. Similarly, when users are writing the reasons, their personalized ratings are helpful for us to predict how good or how bad about their experiences. The personalized ratings also can be regarded as a supervised signal to generated the accurate reasons. Consequently, We integrate the feature in the process of rating prediction into reason generation to make the two tasks mutually reinforcing.

The main contributions of our framework are summarized below:

- We propose a multi-task learning model named EMER combining an Encoder-Decoder model and a MLP model. The model can simultaneously generate recommendation reasons and predict precise ratings. Numerous experiments show that it is meaningful to combine the two tasks together, because their learning processes promote each other.
- We combine product's title in Encoder-Decoder model to generate recommendation reasons, because it is simple and contains significant product-related information. Meanwhile, we add user embedding and item embedding to obtain a personalized recommendation reason.
- We have experimented on multiple benchmark datasets to prove our multi-task learning model and numerous experiments demonstrate that our model achieves significant improvements over the state-of-the-art models.

The rest of this paper is organized as follows. We start with an overview of related work in Section 2. Section 3 presents the details of our model. Experimental setup and results are given in Section 4 and 5. At last, Section 6 concludes this paper.

2 Related Work

Explainable recommendation tries to help users understand the recommendation results and make better and faster decisions. It is a multi-task learning framework which contains explanation of recommendation and rating prediction, so we will survey related work on these two areas in this section.

2.1 Explanation of Recommendation

Explanation of recommendation helps users make better decisions by utilizing straightaway explanations. As a less explored direction in recommendation system, it can be divided into four aspects, which are utilizing aspect information extracted from review text [6], [13], incorporating knowledge graph [14]–[16], friend-to-friend dialogue explanations [10], [17], [18] and explanations do not directly deal with texts [11], [12].

The first one is utilizing aspect information extracted from review text to enable an interpretable representation. Hou et al. [6] propose an Aspect-based Matrix Factorization model (AMF) combining two metrics which are User Aspect Preference (UAP) and Item Aspect Quality (IAQ) and they can quantitatively explain why a user chooses an item by the two metrics. Wu et al. [13] propose a context-aware user-item representation learning model containing review-based feature learning and interaction-based feature learning.

The second one is incorporating knowledge graph into recommendation systems to offer explanations. Wang et al. [14] contribute a new model named Knowledgeaware Path Recurrent Network (KPRN) which can generate path representations by composing the semantics of both entities and relations. Xian et al. [15] propose a method called Policy-Guided Path Reasoning (PGPR) which couples recommendation and interpretability by providing actual paths in a knowledge graph. Ma et al. [16] propose

a joint learning framework which can exploit knowledge graphs to induce explainable rules from item associations in the rule learning module and provide rule-guided recommendations based on the rules in the recommendation module. Xian et al. [19] propose a CoArse-to-FinE neural symbolic reasoning approach (CAFE). It first generates user profiles as coarse sketches of user behaviors, which subsequently guide a path-finding process to derive reasoning paths for recommendations as fine-grained predictions.

The third one is generating recommendation reasons to make users feel as if they are receiving recommendations from their friends. Li et al. [10] propose a deep learning based framework named NRT which can simultaneously predict precise ratings and generate abstractive tips. Chen et al. [17] propose visually explainable recommendation based on attentive neural networks to model the user attention on images under the supervision of both implicit feedback and textual reviews. Gao et al. [18] develop a Deep Explicit Attentive Multi-View Learning Model (DEAML) which utilizes an attentive multi-view learning framework for rating prediction and formulates personalized explanation generation as a constrained tree node selection problem. Lin et al. [20] study the task of explainable outfit recommendation and proposed a deep learning-based framework, called NOR, which simultaneously gives outfit recommendations and generates abstractive comments as explanations. Sun et al. [21] propose a transfer learning based model for preference prediction and review generation. They argued that these two tasks are presented in dual form to inject the probabilistic duality of the two tasks in the training stage.

The last one does not directly deal with text. For example, the reciprocal explanation introduced by Kleinerman et al. [11], directly takes the user's preferences or the attributes of the recommended user as explanations, and concludes that the choice of reciprocal explanation closely related to users' cost for following the recommendations. Park et al. [12] propose UniWalk, using a unified graph structure to exploit ratings and social graph, which takes similar users' choices or similar items the user likes as explanations. Tal et al. [22] propose a Dual Attention Recommender with Items and Attributes (DARIA), a novel approach able to combine two dependable neural attention mechanisms to better justify its suggestions. They utilize two self-attention components to describe users by their most characteristic past activities and items by their best depicting attributes.

Our work is most similar to Li et al. [10]'s work. By combining neural rating regression and gated recurrent neural networks together, their model can simultaneously predict precise ratings and generate abstractive tips. Our work differs in three ways. First, we combine user embedding, item embedding and item's title into our model to realize personalized reason generation. Second, we adapt Encoder-Decoder model to map the input into recommendation reason. Finally, we combine these the two tasks in a different way.

2.2 Rating Prediction

Recommendation systems are mainly categorized into collaborative filtering, content-based and hybrid recommendation and we firstly discuss about collaborative filtering. Collaborative filtering achieves great success when LFM and MF were introduced in late 2000's [8], [9]. They always map users and items into latent factors in one shared space and each dimension in latent factors represents the degree of preference for implicit attributes.

Then a great deal of models are proposed to solve data sparsity problem and cold start problem. Wang et al. [23] propose a model named Collaborative Topic Regression (CTR) which combines the merits of traditional collaborative filtering and probabilistic topic modeling by using item's specifications. Ling et al. [24] propose a model titled Ratings Meet Reviews (RMR) harnessing the information of both ratings and reviews.

Recently, deep learning has been revolutionizing the recommendation architectures dramatically [25]. This kind of recommendation system is always based on MLP, Autoencoder (AE), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), etc. MLP can be used to add nonlinear transformation. He et al. [26] ensemble MF and MLP under the Neural network-based Collaborative Filtering (NCF) framework. Cheng et al. [27] jointly train wide linear models and deep neural networks to combine the benefits of memorization and generalization.

In contrast to collaborative methods, content-based systems can recommend new items to users without any history, because they analyze items' descriptions to identify items that are of particular interest to users [3]. Content-based systems include the following steps: (1) extract the attributes of items, (2) compare the attributes of items with the preferences of users, (3) recommend items with characteristics that fit users' interests [28]. The key purpose is to determine whether a user will like a specific item. This task can be solved by using heuristic methods [29] and classification algorithms [30].

Hybrid recommendation systems usually combine collaborative filtering with content-based filtering to exploit merits of both techniques. Xiao et al. [31] propose a novel hybrid recommendation system which not only makes full use of content-based and collaborative filtering recommendation to solve the cold-start problem but also improves the accuracy of recommendation by selecting the nearest neighbor. Tzamousis et al. [32] employ various machine-learning (ML) algorithms for learning an efficient combination of a diverse set of recommendation algorithms and select the best blending for a given input. Hybrid recommendation systems can take full advantages of a variety of recommendation technologies and avoid their disadvantages.

2.3 Multi-Task Recommendation

Multi-task recommendation refers to the work that considers both recommended tasks and explainability.

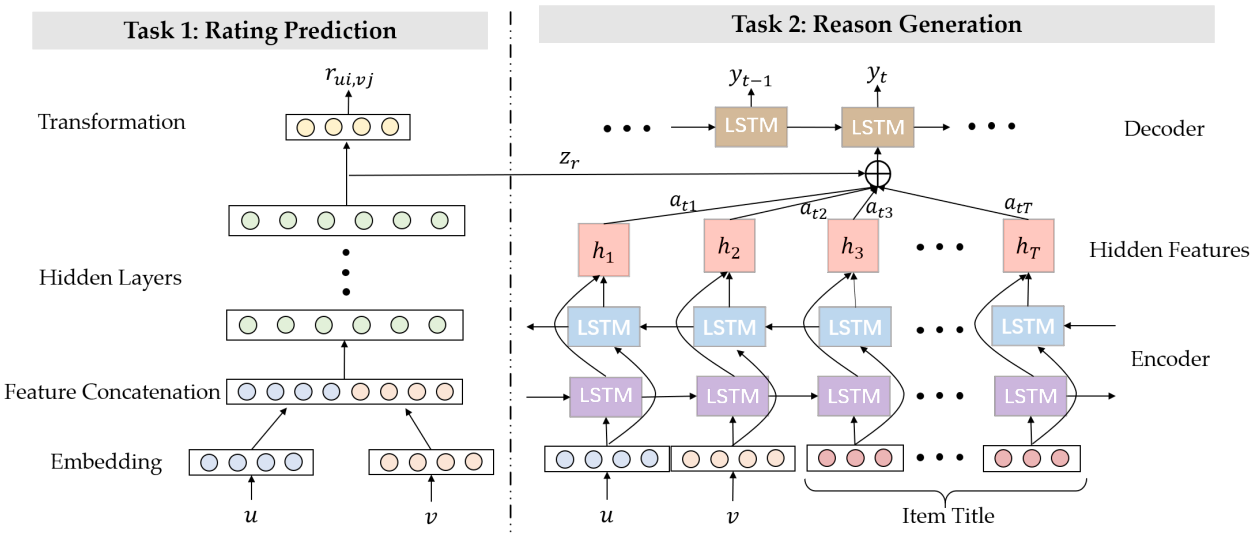


Fig. 2: Our proposed multi-task learning model EMER for rating prediction and reason generation.

J3Rs [33] combines multi-layer perceptron and pointer-generator networks for jointly learning the shared users' and items' latent factors, and uses an MTL approach for predicting user ratings and review summary generation. MT [34] combines matrix factorization, for rating prediction, and adversarial sequence to sequence learning for explanation generation. PETER [35], proposes a personalized Transformer for Explainable Recommendation, using IDs to predict the generation of words, and complete the tasks of rating prediction and explaination at the same time. NETE [36], is a tailored GRU for explainable recommendation, and employ a MLP to capture interactions between users and items.

3 Our Model

Our model is a multi-task learning model, which consists of reason generation and rating prediction. Then we will introduce how we complete these two tasks in this section.

3.1 Overview

Given a user u_i from user pool $\mathcal{U} = \{u_1, u_2, ..., u_m\}$ and an item v_j from item pool $\mathcal{V} = \{v_1, v_2, ..., v_n\}$, the task of our model is to simultaneously yield predicted rating and generate recommendation reason for personalized explainable recommendation. The rating prediction task is to provide predicted ratings $\widehat{r}_{i,j}$ which means whether a user u_i will prefer an item v_j. The reason generation task is to generate a recommendation reason $y_{i,j} = (y^1_{i,j}, y^2_{i,j}, ..., y^k_{i,j})$ consisting of a sequence of words.

We need a dataset $\mathcal{D} = \{\mathcal{U}, \mathcal{V}, \mathcal{T}, \mathcal{R}, \mathcal{Y}\}$, where $\mathcal{T}$ is the set of items' titles, $\mathcal{R}$ is the set of ratings and $\mathcal{Y}$ is the set of reasons. As shown in Figure 2, our model can be divided into two core components, a rating prediction part and a reason generation part. From the left part of Figure 2, we can see that given user u_i and item v_j, our model obtains the predicted rating $\widehat{r}_{i,j}$ through MLP [26]. The right part of Figure 2 depicts the process to generate recommendation reasons and it contains two major components, the encoder for user and item information and the decoder for reason generation. Based on a bi-directional Long-Short Term Memory (BLSTM) [37] in the lower right part of Figure 2, the encoder extracts feature from user u_i, item v_j and item's title $t_j \in \mathcal{T}$. In the upper right part of Figure 2, attention mechanism [38] and multi-layer LSTM are utilized in the reason generation decoder to translate feature extracted by encoder into a sequence of words as an explanation.

3.2 Rating Prediction

The aim of rating prediction is to model users' preference on items based on their past interactions. Although MF model has achieved great success in rating prediction, but inner product, which simply combines the multiplication of latent features linearly, may not be sufficient to capture the complex structure of user historical interaction data [26]. Meanwhile, it has been proved that deep neural networks can generalize better unseen feature combinations through low-dimensional dense embeddings learned for the sparse features [27]. In this paper, we utilize MLP to model interaction between users and items and complete the rating prediction task.

As shown in the left part of Figure 2, we employ MLP in this part to implement rating prediction. First, for a user u_i and an item v_j, we can obtain their embeddings $p_{u_i} \in \mathbb{R}^{k_u}$ and $q_{v_j} \in \mathbb{R}^{k_v}$. Firstly, we concatenate these two vectors together:

$$z_0 = [p_{u_i}, q_{v_j}]. \tag{1}$$

After concatenating user embedding and item embedding, we adapt a standard MLP to learn the interaction between user and item latent features. MLP can utilize nonlinear activation function and deep structure to model high-order feature interaction and map the concatenated vector z_0 into a shared hidden space. We can describe it as follows

$$z_1 = \varphi(W_1 \cdot z_0 + b_1), \tag{2}$$

where $W_1 \in \mathbb{R}^{k_r \times (k_u + k_v)}$, $b_1 \in R^{k_r}$ and φ denote the weight matrix, bias vector, and activation function. For activation function, we employ sigmoid here and other choices including tanh have also been tested, but they lead to unfavored performance. For better performance, we can add more layers of non-linear transformations into our model and get the final output z_r.

Finally, we transform the final output z_r into a real-valued rating $\widehat{r}_{i,j}$ as follows

$$\widehat{r}_{i,j} = W_r \cdot z_r + b_r, \tag{3}$$

where $W_r \in \mathbb{R}^{1 \times k_r}$ and $b_r \in \mathbb{R}^1$.

3.3 Reason Generation

Generating reasons with only user embedding and item embedding is one challenge, so we propose to combine item's title with user embedding and item embedding to generate explanations. There is plenty of key information in item's title due to the nature of item's title which lies in how to describe an item. Consequently, how to make full use of item's title becomes the key problem in our paper. Item's title consists a sequence of words and there is a certain sequence relationship between these words. Encoder-Decoder model becomes the most appropriated model [39], because it has made a great achievement in sequence feature extraction and text generation related tasks. Inspired by its performance, we combine it in our model to model the generation probability $p(y_{ij}|u_i, v_j, t_j, z_r)$, where $y_{ij} \in \mathcal{Y}$. Firstly, we need to utilize the user and item encoder to model user u_i, item v_j and item's title t_j. After obtaining the sequence features extracted by encoder, we can base our recommendation reasons on these features by the decoder for reason generation.

3.3.1 The Encoder for User and Item Information

The encoder in Encoder-Decoder model encodes the input sequence to get a semantic representation for decoder to use, so we need to obtain the input sequence first. In this paper, we set user u_i, item v_j and item's title t_j as the input sequence. We set the items' titles to the same length, which can be represented as $t_j = (t_{j1}, t_{j2}, ..., t_{jl})$. Then, we concatenate u_i, v_j and t_j to compose X as follows

$$X = (u_i, v_j, t_{j1}, t_{j2}, ..., t_{jl}). \tag{4}$$

The length of X can be $T = l + 2$, where l is the length of item's title.

It will be very beneficial for many sequential feature extraction tasks to access the context of the past as well as the context of the future. Standard recurrent neural networks (RNN) processes sequences in time series, which often ignores future context information [38]. Luckily, the Bidirectional Recurrent Neural Networks (BRNN) can be trained without the limitation because it is trained simultaneously in forward and backward direction [40]. LSTM solves the gradient vanishing/exploding problems in RNN. Consequently, we choose BLSTM in encoder part and LSTM in decoder part.

Through BLSTM, we can transform input sequence X from x_1 to x_T into output h from h_1 to h_T. In the positive time direction, we can get the forward hidden states $\{\overrightarrow{h}_1, \overrightarrow{h}_2, ..., \overrightarrow{h}_T\}$. Meanwhile, we can get the backward states $\{\overleftarrow{h}_1, \overleftarrow{h}_2, ..., \overleftarrow{h}_T\}$ from the negative time direction. And then, we connect the hidden states of the corresponding time to get $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$, which captures past and future information. For the final forward outputs $\{\overrightarrow{c}_T, \overrightarrow{h}_T\}$ and backward outputs $\{\overleftarrow{c}_1, \overleftarrow{h}_1\}$, we process them as follows

$$c_{initial} = \varphi(W_c \cdot ([\overrightarrow{c}_T, \overleftarrow{c}_1]) + b_c), \tag{5}$$

$$h_{initial} = \varphi(W_h \cdot ([\overrightarrow{h}_T, \overleftarrow{h}_1]) + b_h), \tag{6}$$

where $W_c \in \mathbb{R}^{k \times 2 \times k_e}$, $b_c \in \mathbb{R}^k$, $W_h \in \mathbb{R}^{k \times 2 \times k_e}$ and $b_h \in \mathbb{R}^k$. φ is the activation function and here we use tanh.

3.3.2 The Decoder for Reason Generation

In this paper, we employ multi-layer LSTM as our decoder. First, we take $[c_{initial}, h_{initial}]$ as the initial state for the decoder. This is reasonable and useful because we can generate explanations based on the output of the encoder for user and item information. Just utilizing a fixed-length vector brings us a bottleneck in our model, so we combine attention mechanism to avoid this problem [38].

We can define each conditional probability in decoder as

$$p(y_{i,j}^t | y_{i,j}^1, ...y_{i,j}^{t-1}, x_1, ...x_T) = p(y_{i,j}^t | y_{i,j}^{t-1}, s_{t-1}, h_t^*), \tag{7}$$

where s_{t-1} is the hidden state and h_t^* is the context vector for time t.

The context vector h_t^* can be calculated as follows

$$e_{ti} = g(s_{t-1}, h) = V \cdot tanh(W \cdot h_i + U \cdot s_{t-1} + b), \tag{8}$$

$$a_{ti} = \frac{exp(e_{ti})}{\sum_{k=1}^{T} exp(e_{tk})}, \tag{9}$$

$$h_t^* = \sum_{k=1}^{T} a_{ti} h_i, \tag{10}$$

where $W \in \mathbb{R}^{k_a \times 2 \times k_e}$, $U \in \mathbb{R}^{k_a \times k}$, $b \in \mathbb{R}^{k_a}$ and $V \in \mathbb{R}^{1 \times k_a}$.

TABLE 1: Dataset statistics.

Dataset	#Users	#Items	#Interactions	#Density	#Words	#Title Length
Movies_and_TV	115,522	29,667	1,039,619	0.030%	22,159	11
Electronics	192,365	61,820	1,642,882	0.014%	17,401	32
CD_and_Vinyl	72,111	49,872	864,572	0.024%	19,894	10
Home_and_Kitchen	66,516	28,130	549,506	0.029%	9,380	19
Yelp	9487	53266	149583	0.029%	19280	20

At each step, we will calculate weight a_{ti} for each hidden state in h and there will be a very large probability that some states will always be the focus. To solve this problem, we employ coverage mechanism [41], which can be described as follows

$$c_t = \sum_{t'=0}^{t-1} a_{t'}, \tag{11}$$

$$e_{ti} = g(s_{t-1}, h, c_t) = V \cdot tanh(W \cdot h_i + U \cdot s_{t-1} + W_c \cdot c_{ti} + b), \tag{12}$$

where $W_c \in \mathbb{R}^{k_a \times 1}$ and in this paper, we set the matrix elements of W_c as 1.

Then at each step, the input of multi-layer LSTM can be represented as $[y_{t-1}, h_t^*, z_r]$. z_r is the feature of rating prediction and we combine it here to combine the two subtasks to get a better performance.

We add the final generative layer to map $[s_t, h_{t+1}^*]$ into a vocabulary-size vector o_t, which can be described as follows

$$o_t = \varsigma(W_o \cdot [s_t, h_{t+1}^*] + b_o), \tag{13}$$

where $W_o \in \mathbb{R}^{k_o \times (k+2\times k_e)}$, $b_o \in \mathbb{R}^{k_o}$ and ς is the softmax function.

3.4 Multi-task Learning

We formulate rating prediction as a regression problem and the mean squared loss function is formulated as

$$\mathcal{L}_r = \frac{1}{\mid \chi \mid} \sum_{u_i \in \mathcal{U}, v_j \in \mathcal{V}} (\widehat{r}_{i,j} - r_{i,j})^2, \tag{14}$$

where χ is the training set and $r_{i,j}$ is the ground truth rating between user u_i and item v_j. This objective can be understood as a penalized log likelihood under a Gaussian model for each entry [39].

For reason generation, our loss function is divided into two parts, which can be described as follows

$$\mathcal{L}_t = -\log p(w_t^*), \tag{15}$$

$$\mathcal{L}_{coverage} = \sum_t min(a_t, c_t), \tag{16}$$

where $p(w_t^*)$ means the probability of the target word w_t^* based on o_t. Then we add the two parts together

$$\mathcal{L}_g = \mathcal{L}_t + \mu \mathcal{L}_{coverage}, \tag{17}$$

where μ means the weight of coverage loss.

We integrate both subtasks, namely rating prediction and reason generation, into a unified multi-task learning framework and then final objective function to be minimized is

$$\mathcal{L} = \lambda_g \mathcal{L}_g + \lambda_r \mathcal{L}_r + \lambda_n \parallel \theta \parallel_2^2, \tag{18}$$

where θ is the set of parameters in our model. λ_r, λ_g and λ_n are the weight proportion of each term.

4 EXPERIMENTAL SETUP

In this section we set up experiments aimed at assessing the performance of rating prediction and reason generation. Here, the details of datasets, evaluation metrics, baseline descriptions and experimental settings are given.

4.1 Datasets

We conduct our experiments on the Amazon e-commerce datasets[2] [42], [43]. To guarantee sufficient data, we choose four larger collections of data from Amazon 5-core and their corresponding meta data. 5-core means each of the remaining users and items have 5 reviews each. The datasets selected can be listed as follows: Movies_and_TV, Electronics, CD_and_Vinyl and Home_and_Kitchen. It has been decipted that item's title plays an important role in our model, so we need to preprocess these datasets to select items with titles. After preprocessing, we show the statistics of our datasets in Table 1. Taking Movies_and_TV for example, there are 115,522 users and 29,667 items in this dataset after preprocessing and the dataset is are very sparse which can be seen from #Density. After filtering out the words with low term frequency in the tips and titles, the words left in each dataset can be found in #Words and the length of titles of each dataset can be see from #Title Length.

We divide each dataset after processing into three subsets: 80%, 10%, and 10%, for training, validation, and testing and utilize the validation dataset to tune all parameters in our model.

2. http://jmcauley.ucsd.edu/data/amazon/index.html

Considering the sensitivity of the algorithms to data, we perform 10-fold cross validation on each dataset. Moreover, we perform the same dataset input for each algorithm for the sake of fair comparison.

4.2 Evaluation Metrics

We use the following measures for evaluation on different tasks.

RMSE: To evaluate the performance of all algorithms, we calculate Root Mean Square Error (RMSE), which is widely used for rating prediction in recommendation systems. The lower RMSE score is, the better performance is. Given a predicted rating $\widehat{R}_{u,i}$ and a ground-truth rating $R_{u,i}$ for user u and item i, the RMSE is calculated as

$$RMSE = \sqrt{\frac{1}{N}\sum_{u,i}(\widehat{R}_{u,i} - R_{u,i})^2}, \tag{19}$$

where N indicates the number of ratings between users and items.

ROUGE: ROUGE score includes measures to automatically determine the quality of text generation by comparing it to ideal summaries created by humans[3] [44]. So we apply it in our model to evaluate the quality of the generated reasons. The measures in ROUGE count the number of overlapping units such as n-gram, word sequences, and word pairs between generated reasons and the ground truth written by users. Take ROUGE-N for example. Assuming that $\widetilde{s}_{u,i}$ is the generated reasons of length m and $s_{u,i}$ is the ground-truth sentence of length n between user u and item i. We sign the overlapping n-grams between them as $gram_n$. Then the Precision, Recall and F-score can be calculated as follows

$$ROUGE - N(S) = \frac{\sum_{u,i} Count(gram_n)}{\sum_{u,i} Count(s)}, \tag{20}$$

When $s = s_{u,i}$, we can get the recall of ROUGE-N. When $s = \widetilde{s}_{u,i}$, we can get the precision of ROUGE-N. After obtaining recall and precision of ROUGE-N, we can use the equation of F-score to compute the F-score of ROUGE-N. We can calculate other ROUGE scores in the same way. In this paper, we use Recall, Precision, and F-measure of ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-SU4 to evaluate our model.

4.3 Baselines

To evaluate the performance of rating prediction and reason generation, we compare our model with the following methods.

MF: Matrix Factorization [9]. It is a classical model and can characterize both items and users by vectors of factors inferred from item rating patterns.

3. ROUGE-1.5.5.pl -n 4 -w 1.2 -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0

NNMF: Neural network matrix factorization [39]. It replaces the inner product by a multi-layer feed-forward neural network and optimizing the latent features for a fixed network.

RMR: Ratings Meet Reviews [24]. It applys topic modeling techniques on the review text and aligns the topics with rating dimensions to improve prediction accuracy.

DeepCoNN: Deep Cooperative Neural Networks [45]. It consists of two parallel neural networks coupled in the last layers for rating prediction. One of the networks focuses on learning user behaviors exploiting reviews written by the user, and the other one learns item properties from the reviews written for the item.

NRT: Neural Rating Regression with Abstractive Tips Generation [10]. It is a deep learning based framework which can simultaneously predict precise ratings and generate abstractive tips with good linguistic quality simulating user experience and feelings.

Re-VECF: Review-enhanced visually explainable collaborative filtering [17]. It is based on attentive neural networks to model the user attention on images under the supervision of both implicit feedback and textual reviews.

J3Rs: Joint Multi-task Learning of Ratings and Review Summaries for Explainable Recommendation [33]. It combines multi-layer perceptron and pointer-generator network for jointly learning the shared users' and items' latent factors, and uses MTL approach for user ratings prediction and review summary generation.

MT: Why I like it: Multi-task Learning for Recommendation and Explanation [34]. It is a multi-task recommendation model, which jointly learns to perform rating prediction and recommendation explanation by combining matrix factorization, for rating prediction, and adversarial sequence to sequence learning for explanation generation.

For rating prediction, we compare our model EMER with MF, NNMF, RMR, DeepCoNN, NRT, J3Rs and MT, which including models for rating prediction.

For reason generation, we compare our model EMER with RMR, NRT, Re-VECF, J3Rs and MT. Because RMR is not specialized for reason generation and Re-VECF is designed to combine visual features to provide visually explainable recommendations. So we refine the two models to make them capable for reason generation.

RMR utilizes a topic modeling technique to model review texts and achieves significant improvements in rating prediction. It can generate topics for each item and we choose the topic with the highest probability for a given item. Then we choose top-50 words under this topic for the item. Then we select all reviews for a given user and all reviews for the item and remove reviews whose all words do not appear in the other collection. Finally, we choose one review with the highest degree of similarity with the words selected from the remainder reviews as the generated reason.

Re-VECF is designed to discover user visual preference, with the supervision of implicit feedback as well

as textual reviews. However, for the datasets we selected, there is no means to model the user attention on images. So we replace the image features with latent factors.

Since MT originally generates users' comments, in order to meet our needs, we use MT to generate tips, and finally evaluate the results based on the original tips in the dataset.

4.4 Experimental Settings

In data processing, we filter out the words less than 5 occurrences. The reason is that these low-frequency words may be not commonly used. Learning the embeddings of low-frequency words is difficult. Therefore, filtering out low-frequency words in the data processing phase is a common approach such as in references [38] and [46].

In reason generation, we use K to refer to the maximum length of the generated sentences. In the training phase, we set K to be equal to the maximum length of the tips used for training, in order to make the generated sentence close to the target tip. Note that we require each sentence to end with a special end-of-sentence symbol "$\langle EOS \rangle$", which enables the model to define a distribution over sequences of all possible lengths as in [46]. Then, in the testing phase, we generate the sentences word by word. It indicates the generation is completed when the end-of-sentence symbol "$\langle EOS \rangle$" has been generated. Through this way, we can generate sentences with different lengths.

In model buiding, we firstly tuning the model parameters through validation dataset, then set the dimension of user embedding, item embedding, and word embedding as 300. We set the number of hidden layers in rating prediction as 2 and the dimension of hidden layer as 400. The number of LSTM layer in decoder for reason generation is set as 2 and the dimension of it is set as 512. We set the weight parameters μ as 0.1, and λ_r and λ_g in optimization objective as 1.

In training process, the batch size for mini-batch training is 200 and the learning rate in the beginning is 0.001. We initialize all the optimization parameters according to a uniform distribution in the range of (-0.01, 0.01), and update them by Adam optimizer. When getting the best experimental results on validation dataset, we halt iteration.

In experiments, We mixed the originally divided datasets, and performed 10-fold cross validation on each dataset, then calculated average results. And to strengthen the results, We performed Tukey's HSD (Honestly Significant Difference) test on each dataset. Moreover, to verify the validation of generated explanations, we conducted human evaluation. We randomly selected 500 groups of ratings and explanations generated on our test set to make a questionnaire, and 10 volunteers were invited in this investigation. Volunteers are instructed to give 1-5 points to the generated explanations according to three criteria: (1) fluency, which refers to whether the statement is fluent, whether there are semantic and grammatical errors. (2) clarity: whether the sentence's meaning is clear; (3) rationality: whether the generated sentence and the predicted rating have similar emotional tendencies.

At last, our model is implemented in Tensorflow[4]. All models are trained and tested on an NVIDIA 1080Ti GPU. Our code is released on Github[5].

5 Results and Discussions

In this section, we first present our 10-fold cross validation experimental results on both rating prediction and reason generation, and display the Tukey Test results. Then we completely discuss the importance of multi-task learning and the impact of different module in our proposed model EMER. Finally, we will discuss about the parameters in Eq. 18 which mean the weight of the two tasks in our model.

5.1 Results on Rating Prediction

In this subsection, we evaluate our model for the task of rating prediction. We compare our model with MF [9], NNMF [39], RMR [24], DeepCoNN [45], NRT [10], J3Rs [33], MT [34] and the rating prediction results of our model and comparative models on all datasets are given in Table 2.

It is obvious that in most cases, our model outperforms all baseline methods in terms of MAE and RMSE metrics on all datasets. And our model achieves the best performance on both MAE and RMSE metrics on average, which can be seen from the last two columns of Table 2. This is because we combine the feature of rating prediction into reason generation. Under the supervision of reason generation, we can obtain more useful information in the feature of rating prediction, so we can get a better result in rating predition. We also performed a Tukey's HSD test on each dataset. From Table 3 We can see that EMER can get a significant improvement on rating prediction most of time, and it is not significant when our model perfroms little worse than DeepCoNN.

From the results we can simultaneously see that models that utilize textual information (EMER, RMR, DeepCoNN, NRT, J3Rs) are generally superior to models without this kind of information (MF, NNMF). It is logical that textual information is complementary to ratings and it can be utilized to improve the representation quality of user's preferences and item's features. So we can improve the accuracy of rating prediction through textual information and give user a better recommendation result.

4. https://tensorflow.google.cn/
5. https://github.com/Anxbbq/EMER

TABLE 2: Performance comparison on rating prediction.

Methods	Movies_and_TV	Electronics	CDs_and_Vinyl	Home_and_Kitchen	Yelp	AVERAGE
MF	1.064	1.142	0.987	**1.055**	0.995	1.049
NNMF	1.114	1.170	1.033	1.095	0.990	1.080
RMR	1.031	1.126	1.025	1.062	0.997	1.048
DeepCoNN	**1.026**	1.122	0.964	1.065	1.003	1.036
NRT	1.027	1.132	0.967	1.090	1.040	1.051
J3Rs	1.070	1.135	0.987	1.061	1.043	1.061
MT	1.364	1.421	1.255	1.488	1.232	1.352
EMER (Ours)	**1.026**	**1.104**	**0.958**	1.057	**0.984**	**1.026**

TABLE 3: Tukey's HSD test on rating prediction ($\times 10^{-2}$).

Method pair		Mean-diff ($X - Y$)				
X	Y	Movies_and_TV	Electronics	CDs_and_Vinyl	Home_and_Kitchen	Yelp
EMER	MF	3.64*	2.88*	3.09*	2.43*	1.12*
EMER	NNMF	6.20*	6.18*	5.00*	8.45*	1.00*
EMER	RMR	0.52	0.03	2.19*	1.01*	1.56*
EMER	DeepCoNN	-0.79	0.17	-0.49	1.40*	2.57*
EMER	NRT	0.07	3.08*	0.86*	3.54*	3.13*
EMER	J3Rs	0.76	1.88*	1.84*	1.41*	4.56*
EMER	MT	18.63*	27.99*	19.94*	42.94*	30.65*

The numbers are mean-diff between X and Y. Positive mean-diff means X performs better. "*" means that the difference is statistically significant at $\alpha = 0.05$.

TABLE 4: Performance comparison on reason generation on Movies_and_TV.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
RMR	13.01	3.43	5.01	2.09	0.39	0.66	12.98	3.25	5.13	5.79	0.08	1.31
NRT	10.49	13.06	10.82	2.00	2.41	2.02	10.26	12.75	10.57	4.24	6.21	4.1
Re-VECF	8.81	12.24	9.38	1.52	1.81	1.45	8.69	12.00	9.22	3.39	5.86	3.34
J3Rs	12.60	11.98	11.04	3.03	2.59	2.44	12.37	11.70	10.81	5.44	5.06	4.02
MT	9.59	13.01	10.18	1.75	2.26	1.79	9.40	12.78	9.97	3.77	6.19	3.81
EMER (Ours)	**13.36**	**16.75**	**13.90**	**4.05**	**4.90**	**4.13**	**13.11**	**16.41**	**13.66**	**6.17**	**8.74**	**6.12**

5.2 Results on Reason Generation

As a multi-task model, our model EMER solves not only the rating prediction problem but also the reason generation problem which can encourage user to accept the recommendation result. In this subsection, we evaluate the second subtask, namely reason generation, of our model by comparing the predicted reasons with the truly posted ones. In order to capture more details of the generated reasons based on our model, we report the result of Recall, Precision, and F-measure (in percentage) of ROUGE-1, ROUGE-2, ROUGE-L and ROUGE-SU4.

The evaluation results of our model and comparative models, namely RMR [24], NRT [10] , Re-VECF [17], J3Rs [33], MT [34] on the reason generation task are given in Table 4, Table 5, Table 8 and Table 9. From the results we can see that our model EMER generally achieves better performance than other models. There are a few values of our model on Recall is slightly lower than compared methods, but the performance on F1-score which is the weighted average of Precision and Recall is more important. We also show the Tukey'HSD test results in Table 11 and find that our EMER model has significant improvement over other methods. Besides, on F1-score of ROUGE-2, the EMER is little weaker than J3Rs on Electronics and Home_and_Kitchen, but the deficiency is in a low level at -0.0060 and -0.0074 shown in Table 11, and in ROUGE-1 and ROUGE-L, our model performs significantly better than J3Rs. Our model's better performance is as expected because product aspects that users are interested in may be directly aligned with the product titles, and meanwhile, the relationship among these features can be effectively captured by our encoder for user and item information. Then through attention

TABLE 5: Performance comparison on reason generation on Electronics.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
RMR	10.56	2.57	4.02	0.85	0.19	0.28	9.56	2.35	2.78	3.57	0.50	0.79
NRT	10.13	13.74	10.74	1.87	2.6	1.98	10.03	13.57	10.63	4.49	8.25	4.66
Re-VECF	11.05	16.73	12.15	2.23	3.48	2.42	10.97	16.55	12.05	4.47	9.52	4.83
J3Rs	**17.71**	15.59	14.33	**5.40**	**4.43**	**4.00**	**17.45**	13.86	14.09	**8.55**	7.53	5.61
MT	13.16	17.97	13.93	3.02	4.18	3.15	13.03	**17.71**	13.76	5.58	**9.71**	5.71
EMER (Ours)	15.05	**17.59**	**14.74**	3.70	4.21	3.45	14.84	17.29	**14.51**	6.59	9.04	**5.83**

TABLE 6: Results of User Study.

Methods	Fluency	Clarity	Rationality
NRT	4.57	4.67	4.33
RMR	3.31	3.27	3.16
EMER (Ours)	**4.81**	**4.89**	**4.35**

TABLE 7: Tukey's HSD Test on User Study.

Method pair		Mean-diff ($X - Y$)		
X	Y	Fluency	Clarity	Rationality
EMER	NRT	0.2381*	0.2381*	0.0198
EMER	RMR	1.5119*	1.6310*	1.1825*

The numbers are mean-diff between X and Y. Positive mean-diff means X performs better. "*" means that the difference is statistically significant at $\alpha = 0.05$.

mechanism in decoder for reason generation, we can identify the importance of every hidden state in encoder in the modeling process. Meanwhile, fusing the feature of rating prediction into reason generation can bring user's emotional tendency on this product and obtain better result on reason generation.

For the reason generation task on Yelp as shown in Table 10, the performance of our EMER model is lower than others. We explored the utilized Yelp dataset, and found the reason that our model requires the attribute of product title since product titles generally have rich fields, but the item titles in yelp dataset are very short. We made a statistics and found that the average length of item titles on Amazon is 9.8, while the average length on Yelp is 2.4. At the same time, the compared methods utilized the review comments but not the titles. So they are not affected by the low informative title on Yelp. Therefore, the generation performance of our model on yelp is lower than other methods. Then we could conclude that our model is more suitable to be applied on the product recommendation which has informative titles such as Movies_and_TV, Electronics, CD_and_Vinyl and Home_and_Kitchen datasets, but for the business recommendation such as Yelp dataset, we suggest other methods for the generation task.

Additionally, from Table 6 we can see that the EMER algorithm can get the highest average score in terms of fluency, clarity and rationality with contrast to NRT and RMR. We also utilize the Fleiss kappa method as the inter-rater agreement and we got Fleiss Kappa coefficient $K = 0.256$ which means it is a fair agreement. Table 7 shows our generated reasons have a significant improvement on user evaluations most of time, which proves that the explanation generated by our algorithm is generally well in grammar and meaning in the eyes of users.

5.3 Multi-task Learning Analysis

To verify the effectiveness of the proposed multi-task learning model on rating prediction and reason generation, we conduct experiments with individual task of EMER on datasets it performs best during 10-cross validation. The experimental results are shown in Table 13, Table 14 and Table 15.

From Table 13, we can see our model EMER(w/o reason) (our model without reason generation) does poorly without the supervision of reason generation. This is because ratings and reasons have similar emotional tendencies and the training process of the two tasks are complementary. So it is beneficial to train them together for rating prediction.

It is obvious that there is a little improvement from Table 15. Due to the fact that we utilize Encoder-Decoder model and attention mechanism, the user embedding and item Embedding in encoder bring some personalized information and item's title plays an more important role in reason generation. Finally, the feature of rating prediction brings a small profit. In Table 14 our proposed model has a slightly worse ROUGE score than the EMER(w/o rating) and EMER(same) versions on the Electronics dataset. We analyzed the statistical features of Electronics and found it has a very low data density at 0.014% and relatively long title length at 32 to make it more sensitive to the model complexity. Generally, training a deep model with a large number of parameters needs more data. However, the Electronics dataset is sparse and the length of the target title is much longer than other datasets, so it cannot afford enough information to train our deep model. Therefore, when we reduce the model parameters by cutting off the rating part of EMER and sharing the same embeddings for different tasks, the lite models EMER(w/o rating) and EMER(same) could achieve better performance than the full model EMER on Electronics dataset.

TABLE 8: Performance comparison on reason generation on CDs_and_Vinyl.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
RMR	11.69	3.43	5.01	1.29	0.31	0.45	10.45	2.64	4.27	4.36	0.67	1.01
NRT	8.22	10.28	8.49	1.32	1.64	1.34	8.05	10.05	8.31	3.27	4.77	3.12
Re-VECF	7.47	9.58	7.70	1.05	1.29	1.03	7.32	9.35	7.52	2.91	4.45	2.70
J3Rs	**12.45**	8.51	8.78	**2.88**	1.79	1.82	**12.06**	8.21	8.48	**5.58**	3.24	2.86
MT	8.16	10.81	8.65	1.20	1.58	1.24	7.80	10.57	8.47	3.26	5.17	3.26
EMER (Ours)	9.56	**12.04**	**9.91**	1.80	**2.26**	**1.84**	9.33	**11.74**	**9.67**	3.77	**5.50**	**3.64**

TABLE 9: Performance comparison on reason generation on Home_and_Kitchen.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
RMR	13.01	3.43	5.01	2.09	0.39	0.66	12.98	3.25	5.13	5.79	0.08	1.31
NRT	10.49	13.06	10.82	2.00	2.41	2.02	10.26	12.75	10.57	4.24	6.21	4.10
Re-VECF	8.81	12.24	9.38	1.52	1.81	1.45	8.69	12.00	9.22	3.39	5.86	3.34
J3Rs	12.60	11.98	11.04	3.03	2.59	2.44	12.37	11.70	10.81	5.44	5.06	4.02
MT	9.59	13.01	10.18	1.75	2.26	1.79	9.40	12.78	9.97	3.77	6.19	3.81
EMER (Ours)	**13.36**	**16.75**	**13.90**	**4.05**	**4.90**	**4.13**	**13.11**	**16.41**	**13.66**	**6.17**	**8.74**	**6.12**

TABLE 10: Performance comparison on reason generation on Yelp.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
RMR	7.55	7.63	7.50	0.46	0.31	0.38	7.35	6.65	7.01	2.31	1.98	2.16
NRT	7.62	6.86	6.86	0.42	0.36	0.39	7.28	6.45	6.84	2.24	1.82	2.01
Re-VECF	7.19	8.24	7.68	0.21	0.32	0.25	6.87	7.81	7.31	1.93	1.88	1.91
J3Rs	9.41	10.41	9.89	2.21	2.15	2.18	8.83	9.75	9.27	3.59	3.74	3.67
MT	11.76	6.47	8.35	0.79	0.45	0.57	10.48	5.52	7.23	3.68	1.58	2.22
EMER (Ours)	9.03	7.19	8.00	0.50	0.37	0.43	8.53	6.78	7.56	2.64	1.70	2.07

We list some examples of generated reasons and their corresponding predicted ratings of EMER and NRT in Table 12. We can see that EMER predicted ratings are more closed to the real ratings. What's more, our predicted ratings shows the consistent sentiment with generated reasons. For example, when we generate "i don't believe this album is a great live album" for an item, its corresponding rating is a medium 3.74, but when we generate "a great recording of the best symphony of the 20th century", its corresponding rating become higher at 4.87. Moreover, the explanations we generate tend to have richer meanings, for example, "a true story of passion and faith" generated by EMER is more specific by pointing out the theme "passion and faith" of the story than "a true story of a great story" generated by NRT.

5.4 The Necessity of Coverage Mechanism

In order to explore the effectiveness of coverage mechanism in our model, we perform experiments without coverage mechanism and test the performance on rating prediction and reason generation without coverage mechanism. Table 13, Table 14 and Table 15 show some of those results.

As we can observe from Table 13, coverage mechanism always brings benefit to our model in rating prediction. For instance, our model EMER performs better in terms of MAE and RMSE metrics on all datasets comparing with EMER(w/o cover) (our model without coverage mechanism). Although coverage mechanism plays a major role in reason generation, it can also brings a balance between the two tasks. So through coverage mechanism, we can get a better performance in rating prediction.

As presented in Table 14 and Table 15, EMER which combines coverage mechanism outperforms EMER(w/o cover). Thus we conclude that the coverage mechanism is helpful for the reason generation task. This is reasonable due to the very nature of coverage mechanism. Coverage mechanism will bring punishment when the decoder always pays attention to several words in encoder. We can pay attention to each element in encoder in the process of reason generation and then get a better result.

TABLE 11: Tukey's HSD Test on reason generation ($\times 10^{-2}$).

Method pair		Mean-diff ($X-Y$)											
		Movies_and_TV			Electronics			CDs_and_Vinyl			Home_and_Kitchen		
X	Y	1	2	L	1	2	L	1	2	L	1	2	L
EMER	RMR	8.15*	3.52*	9.75*	10.37*	2.53*	12.55*	4.85*	1.67*	7.39*	8.56*	1.89*	11.90*
EMER	NRT	3.28*	1.96*	3.21*	5.01*	1.11*	6.25*	1.99*	0.57*	2.55*	3.16*	0.35*	5.52*
EMER	Re-VECF	4.79*	2.73*	5.66*	4.48*	0.63*	4.68*	2.40*	1.10*	1.10*	4.41*	1.14*	7.36*
EMER	J3Rs	3.21*	1.82*	2.41*	1.82*	-0.60*	1.27*	1.59*	0.35*	1.04*	2.74*	-0.74*	1.32*
EMER	MT	4.57*	2.50*	5.37*	3.04*	0.58*	3.42*	1.87*	0.60*	1.78*	5.43*	0.96*	8.33*

The numbers are mean-diff between X and Y on F1 score of ROUGE 1, ROUGE 2, ROUGE L. Positive mean-diff means X performs better. "*" means that the difference is statistically significant at $\alpha = 0.05$.

TABLE 12: Examples of the predicted ratings by EMER and NRT, the generated reason by EMER and NRT.

Samples	Methods	Generated Reason	Prediction	Predicted Rating
1	NRT	a true story of a great story	4.86	5.0
	EMER	a true story of passion and faith	**4.98**	
2	NRT	a classic album from the masters of the 90's	4.72	4.0
	EMER	a classic album from the greatest female singer of the 90 's	**4.34**	
3	NRT	the best of the best of the best	4.00	2.0
	EMER	i don't believe this album is a great live album	**3.74**	
4	NRT	a great album , but not the best	4.43	4.0
	EMER	a great recording of the best symphony of the 20th century	**4.32**	

TABLE 13: Comparing effect of different part in EMER on rating prediction.

	Electronics		Home_and_Kitchen	
	MAE	RMSE	MAE	RMSE
EMER(w/o reason)	0.935	1.184	0.826	1.100
EMER(w/o cover)	0.837	1.102	0.711	1.046
EMER(same)	0.827	1.104	0.743	1.048
EMER(w/o embedding)	0.827	1.102	0.727	1.034
EMER	0.826	1.101	0.716	1.044

5.5 Combining User Embedding and Item Embedding for Personalized Reason Generation

We further examine whether it is useful to combine user embedding and item embedding in reason generation. Table 13, Table 14 and Table 15 report the performance on rating prediction and reason generation.

We can see that the performance of rating prediction is very similar under both EMER and EMER(w/o embedding) (reason generation of our model without user embedding and item embedding) through Table 13. This is because the performance of rating prediction is mainly affected by MLP and the supervision of reason generation. User embedding and item embedding in the encoder will not have significant impact on it.

It can be observed that EMER with user embedding and item embedding in reason generation achieves better prediction accuracy than EMER(w/o embedding) through Table 14 and Table 15. This proves that incorporating user embedding and item embedding in reason generation can provide more semantic information for understanding user review behaviors. When combining user embedding and item embedding in reason generation, we can obtain personalized recommendation reasons for user.

5.6 Discussion about the Usage of User Embedding and Item Embedding

Due to the fact that we use user embedding and item embedding in both rating prediction and reason generation, we test the performance of our model EMER when we use the same user embedding and item embedding in these two submodels. We can see experimental results from Table 13, Table 14 and Table 15.

From Table 13, we can find out that when we use different embeddings in both tasks, EMER outperforms EMER(same) (our model with the same user embedding and item embedding) on both datasets in terms of MAE and RMSE metrics. This is due the fact that different user embedding and item embedding have larger parameter space and can learn different features in both tasks. Through different embeddings in the two tasks, we can obtain more useful feature aimed at rating prediction and get a better result in rating prediction.

However, the performance in reason generation on different datasets gives us two different results. We can see from Table 15 that the performance of EMER(same) is poor, but obtain the opposite conclusion from Table 14.

TABLE 14: Comparing effect of different part in EMER on reason generation on Electronics.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
EMER(w/o rating)	16.78	16.98	15.33	4.58	4.27	3.88	16.50	16.62	15.04	7.93	8.02	6.15
EMER(w/o cover)	14.99	17.72	14.80	3.63	4.07	3.40	14.81	17.45	14.60	6.56	8.92	5.97
EMER(same)	15.40	18.41	15.36	3.81	4.41	3.62	15.19	18.10	15.12	6.73	9.37	6.19
EMER(w/o embedding)	14.21	15.82	13.21	3.40	3.39	2.85	14.07	15.59	13.05	6.20	7.57	4.82
EMER	15.07	17.78	14.84	3.62	4.07	3.35	14.88	17.48	14.61	6.54	8.88	5.81

TABLE 15: Comparing effect of different part in EMER on reason generation on Home_and_Kitchen.

Methods	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
EMER(w/o rating)	14.21	17.01	14.34	3.83	4.42	3.76	14.07	16.79	14.17	6.32	8.14	5.85
EMER(w/o cover)	15.03	15.29	13.86	4.02	3.89	3.54	14.83	15.03	13.65	6.81	6.88	5.42
EMER(same)	13.98	17.22	14.26	3.70	4.30	3.63	13.84	17.00	14.11	6.10	8.34	5.78
EMER(w/o embedding)	11.42	16.06	12.27	2.44	3.37	2.59	11.36	15.93	12.20	4.50	8.11	4.79
EMER	14.20	17.40	14.49	3.83	4.60	3.85	14.04	17.16	14.32	6.20	8.48	5.96

TABLE 16: Comparing effect of different parameter settings on reason generation on Home_and_Kitchen.

λ_r	ROUGE-1			ROUGE-2			ROUGE-L			ROUGE-SU4		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
0.0	14.21	17.01	14.34	3.83	4.42	3.76	14.07	16.79	14.17	6.32	8.14	5.85
0.5	14.70	16.44	14.26	4.02	4.25	3.73	14.54	16.21	14.08	6.56	7.81	5.73
1.0	14.20	17.40	14.49	3.83	4.60	3.85	14.04	17.16	14.32	6.20	8.48	5.96
1.5	13.61	17.16	14.09	3.53	4.46	3.66	13.81	16.95	13.94	5.94	8.47	5.83
2.0	14.20	17.21	14.38	3.76	4.40	3.71	14.07	16.99	14.23	6.19	8.31	5.82

This is because the difference between the two datasets, namely Electronics and Home_and_Kitchen. Items in Electronics are always well-known and most people have the same similar view to them, so there is no need to utilize different embeddings to obtain more information. But for Home_and_Kitchen, We can not get enough personalized information and emotional information from the user embedding and item embedding in rating prediction, so we can get a better result when we use different embeddings in the two tasks.

5.7 The Effectiveness of Item's Title

To verify the effectiveness of combining item's title into encoder, we visualize the relationship between each element in sequence X which contains item's title and each word in generated reason through attention mechanism. In Figure 3, the transverse axes represents sequence X and the vertical axis represents the generated reason. All little boxes with different colors means different correlations between the two sequences. The brighter the color, the higher the correlation between two elements in the two sequence. From Figure 3, we can clearly see that there is a strong correlation between knife, henckels in sequence X and knives in generated reason, that is reasonable because Peter Henckels created a famous brand ZWILLING which sell knives, cookers, etc.. Consequently, it is quite useful to combine item's title into the encoder for user and item information. This also proves that attention mechanism plays an important role in generating recommendation reasons.

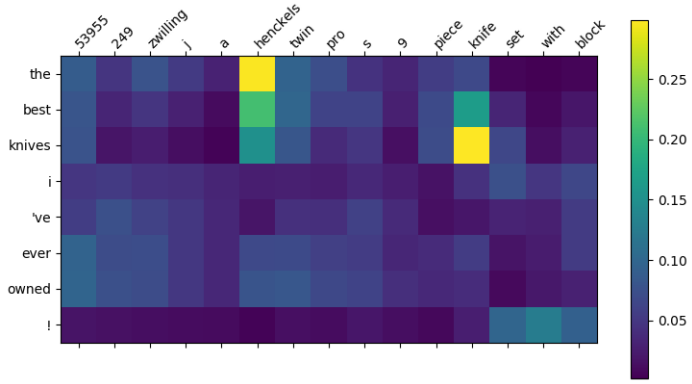


Fig. 3: Visualization of attention mechanism.

5.8 Discussion on the Hyperparameters in Multi-task Learning

In this subsection, we mainly discuss about the parameter λ_g and λ_r in Eq. 18. From Eq. 18 we can know parameter λ_g and λ_r mean a balance between loss in reason generation and loss in rating prediction. We fix λ_g as 1 and tune λ_r from 0.0 to 2.0. we can see that our

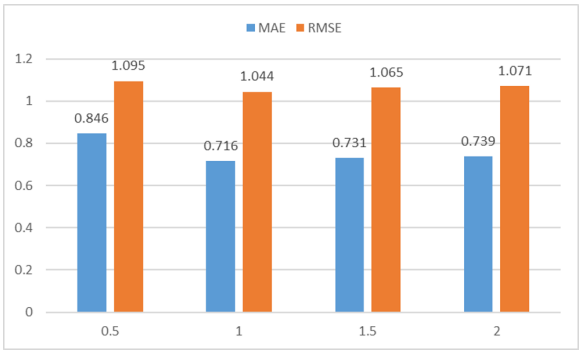


Fig. 4: Comparing effect of different parameter settings on rating prediction on Home_and_Kitchen.

model gets the best result on Home_and_Kitchen when we set λ_r as 1 in Table 16 and Figure 4. In other word, when we set both λ_g and λ_r as 1, we can keep a good balance between the two tasks and obtain the best result in both tasks. This is because the two tasks in our model are mutually reinforcing and there exists no competition between them. There is no need to set different weights to make one of these tasks dominant. And we set 1 as the final values for both λ_g and λ_r and this is used in our all other experiments.

6 Conclusion and Future Work

Accuracy and explanation are both important for recommendation systems. So we identify two main tasks: rating prediction for providing recommendation results and reason generation for providing straightaway explanations. To tackle the two problems, we propose a multi-task learning model named EMER which can simultaneously implement reason generation and rating prediction. The model combines an Encoder-Decoder model and a MLP model. Numerous experiments show that it is meaningful to combine the two tasks together, because their learning processes promote each other. Item's title contains significant product-related information and plays an important role in grabbing users attention, so we fuse it in our model to get more useful information. Experimental results on benchmark datasets show that EMER achieves better performance than the state-of-the-art models on both tasks of rating prediction and reason generation.

In the future, we will incorporate more information into the recommendation system, such as external knowledge graph, so as to make the recommendation results more accurate. Besides, we will exploit the graph reasoning on the knowledge graph to make the explaination more personalized.

Acknowledgments

This work was supported in part by the NSFC under Grants 61902309, 61732008, 61772407, and 1531141; in part by the National Key RD Program of China under Grant 2017YFF0107700; in part by the World-Class Universities (Disciplines) and the Characteristic Development Guidance Funds for the Central Universities (PY3A022); in part by the Fundamental Research Funds for the Central Universities, China (xxj022019003); in part by China Postdoctoral Science Foundation (2020M683496); and in part by the National Postdoctoral Innovative Talents Support Program, China (BX20190273).

References

[1] M. Gupta, P. Kumar, and R. Mishra, "Interest diffusion in heterogeneous information network for personalized item ranking," in *Proc. CIKM*. ACM, 2017, pp. 2087–2090.

[2] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender systems handbook*. Springer, 2011, pp. 1–35.

[3] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *Proc. The Adaptive Web*, 2007, pp. 325–341.

[4] B. Ferwerda, K. Swelsen, and E. Yang, "Explaining content-based recommendations," *Bruceferwerda Com*.

[5] L. Li, Y. Zhang, and L. Chen, "EXTRA: explanation ranking datasets for explainable recommendation," in *Proc. ACM SIGIR*, 2021, pp. 2463–2469.

[6] Y. Hou, N. Yang, Y. Wu, and S. Y. Philip, "Explainable recommendation with fusion of aspect information," *World Wide Web*, vol. 22, no. 1, pp. 221–240, 2019.

[7] B. Abdollahi and O. Nasraoui, "Using explainability for constrained matrix factorization," in *Proc. RecSys*. ACM, 2017, pp. 79–83.

[8] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proc. SIGKDD*. ACM, 2008, pp. 426–434.

[9] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[10] P. Li, Z. Wang, Z. Ren, L. Bing, and W. Lam, "Neural rating regression with abstractive tips generation for recommendation," in *Proc. SIGIR*. ACM, 2017, pp. 345–354.

[11] A. Kleinerman, A. Rosenfeld, and S. Kraus, "Providing explanations for recommendations in reciprocal environments," in *Proc. ACM RecSys*. ACM, 2018, pp. 22–30.

[12] H. Park, H. Jeon, J. Kim, B. Ahn, and U. Kang, "Uniwalk: Explainable and accurate recommendation for rating and network data," *CoRR*, vol. abs/1710.07134, 2017. [Online]. Available: http://arxiv.org/abs/1710.07134

[13] L. Wu, C. Quan, C. Li, Q. Wang, B. Zheng, and X. Luo, "A context-aware user-item representation learning for item recommendation," *ACM Trans. Inf. Syst.*, vol. 37, no. 2, p. 22, 2019.

[14] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," *arXiv preprint arXiv:1811.04540*, 2018.

[15] Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," in *Proc. ACM SIGIR*, 2019, pp. 285–294.

[16] W. Ma, M. Zhang, Y. Cao, W. Jin, C. Wang, Y. Liu, S. Ma, and X. Ren, "Jointly learning explainable rules for recommendation with knowledge graph," in *The World Wide Web Conference*. ACM, 2019, pp. 1210–1221.

[17] X. Chen, Y. Zhang, H. Xu, Y. Cao, Z. Qin, and H. Zha, "Visually explainable recommendation," *arXiv preprint arXiv:1801.10288*, 2018.

[18] J. Gao, X. Wang, Y. Wang, and X. Xie, "Explainable recommendation through attentive multi-view learning." AAAI, 2019.

[19] Y. Xian, Z. Fu, H. Zhao, Y. Ge, X. Chen, Q. Huang, S. Geng, Z. Qin, G. de Melo, S. Muthukrishnan, and Y. Zhang, "CAFE: coarse-to-fine neural symbolic reasoning for explainable recommendation," in *Proc. ACM CIKM*, 2020, pp. 1645–1654.

[20] Y. Lin, P. Ren, Z. Chen, Z. Ren, J. Ma, and M. de Rijke, "Explainable outfit recommendation with joint outfit matching and comment generation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 8, pp. 1502–1516, 2020.

[21] P. Sun, L. Wu, K. Zhang, Y. Fu, R. Hong, and M. Wang, "Dual learning for explainable recommendation: Towards unifying user preference prediction and review generation," in *Proc. WWW*, 2020, pp. 837–847.

[22] O. Tal, Y. Liu, J. Huang, X. Yu, and B. Aljbawi, "Neural attention frameworks for explainable recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 2137–2150, 2021.

[23] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. SIGKDD*. ACM, 2011, pp. 448–456.

[24] G. Ling, M. R. Lyu, and I. King, "Ratings meet reviews, a combined approach to recommend," in *Proc. RecSys*. ACM, 2014, pp. 105–112.

[25] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, p. 5, 2019.

[26] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. World Wide Web*, 2017, pp. 173–182.

[27] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir *et al.*, "Wide & deep learning for recommender systems," in *Proc. DLRS@RecSys*. ACM, 2016, pp. 7–10.

[28] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-based systems*, vol. 46, pp. 109–132, 2013.

[29] R. Garcia and X. Amatriain, "Weighted content based methods for recommending connections in online social networks," in *Workshop on Recommender Systems and the Social Web*. Citeseer, 2010, pp. 68–71.

[30] M. De Gemmis, P. Lops, G. Semeraro, and P. Basile, "Integrating tags in a semantic content-based recommender," in *Proc. RecSys*. ACM, 2008, pp. 163–170.

[31] Y. Xiao and R. Zhong, "A hybrid recommendation algorithm based on weighted stochastic block model," *arXiv preprint arXiv:1905.03192*, 2019.

[32] E. Tzamousis and M. Papadopouli, "On hybrid modular recommendation systems for video streaming," *arXiv preprint arXiv:1901.01418*, 2019.

[33] P. V. S. Avinesh, Y. Ren, C. M. Meyer, J. Chan, Z. Bao, and M. Sanderson, "J3R: joint multi-task learning of ratings and review summaries for explainable recommendation," in *Proc. ECML PKDD*, 2019, pp. 339–355.

[34] Y. Lu, R. Dong, and B. Smyth, "Why I like it: multi-task learning for recommendation and explanation," in *Proc. ACM RecSys*, S. Pera, M. D. Ekstrand, X. Amatriain, and J. O'Donovan, Eds. ACM, 2018, pp. 4–12.

[35] L. Li, Y. Zhang, and L. Chen, "Personalized transformer for explainable recommendation," *arXiv preprint arXiv:2105.11601*, 2021.

[36] ——, "Generate neural template explanations for recommendation," in *Proc. CIMK*, 2020, pp. 755–764.

[37] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith, "Transition-based dependency parsing with stack long short-term memory," in *Proc. ACL*, 2015, pp. 334–343.

[38] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. ICLR*, 2015.

[39] G. K. Dziugaite and D. M. Roy, "Neural network matrix factorization," *arXiv preprint arXiv:1511.06443*, 2015.

[40] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[41] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proc. ACL*, 2017, pp. 1073–1083.

[42] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proc. World Wide Web*, 2016, pp. 507–517.

[43] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proc. SIGIR*. ACM, 2015, pp. 43–52.

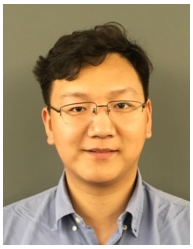
[44] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," *Text Summarization Branches Out*, 2004.

[45] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. WSDM*. ACM, 2017, pp. 425–434.

[46] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. NIPS*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., 2014, pp. 3104–3112.

Jihua Zhu received the B.E. degree in automation from Central South University, China, and the Ph.D. degree in pattern recognition and intelligence system from Xi'an Jiaotong University, China, in 2004 and 2011, respectively. He is currently an Associate Professor with the School of Software Engineering, Xi'an Jiaotong University. His research interests include computer vision and machine learning.

Yujiao He received the B.E. degree from Xi'an Jiaotong University, Xi'an, China, in 2020, and is expected to get the MS degree from Xi'an Jiaotong University in 2023. Her current research interests include social media big data analysis and recommender systems.



Guoshuai Zhao received the PhD degree from Xi'an Jiaotong University, Xi'an, China, in 2019. He was an intern at Microsoft Research Asia, and was a visiting scholar with Northeastern University and with Massachusetts Institute of Technology. Now he is an Associate Professor with Xi'an Jiaotong University. His research interests include social media big data analysis, recommender systems, and natural language generation.

Xuxiao Bu received the B.E. degree from Xi'an Jiaotong University, Xi'an, China, in 2017, and is expected to get the MS degree from Xi'an Jiaotong University in 2020. Her current research interests include social media big data analysis and recommender systems.

Xueming Qian (M'10) received the B.S. and M.S. degrees from Xi'an University of Technology, Xi'an, China, in 1999 and 2004, respectively, and the Ph.D. degree from the School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an, China, in 2008. He was a Visiting Scholar with Microsoft Research Asia from 2010 to 2011. He is currently a Full Professor with Xi'an Jiaotong University. He is also the Director of the SMILES Laboratory, Xi'an Jiaotong University. His research is supported by the National Natural Science Foundation of China, Microsoft Research, and the Ministry of Science and Technology. His research interests include social media big data mining and search. He received the Microsoft Fellowship in 2006 and the Outstanding Doctoral Dissertations of Xi'an Jiaotong University and Shaanxi Province, in 2010 and 2011, respectively.