Dynamic Explainable Recommendation Based on Neural Attentive Models

Xu Chen
TNList, School of Software
Tsinghua University
xu-ch14@mails.tsinghua.edu.cn

Yongfeng Zhang
Department of Computer Science
Rutgers University
yongfeng.zhang@rutgers.edu

Zheng Qin *
School of Software
Tsinghua University
qinzh@mail.tsinghua.edu.cn

Abstract

Providing explanations in a recommender system is getting more and more attention in both industry and research communities. Most existing explainable recommender models regard user preferences as invariant to generate static explanations. However, in real scenarios, a user's preference is always dynamic, and she may be interested in different product features at different states. The mismatching between the explanation and user preference may degrade costumers' satisfaction, confidence and trust for the recommender system.

With the desire to fill up this gap, in this paper, we build a novel **D**ynamic **E**xplainable **R**ecommender (called **DER**) for more accurate user modeling and explanations. In specific, we design a time-aware gated recurrent unit (GRU) to model user dynamic preferences, and profile an item by its review information based on sentence-level convolutional neural network (CNN). By attentively learning the important review information according to the user current state, we are not only able to improve the recommendation performance, but also can provide explanations tailored for the users' current preferences. We conduct extensive experiments to demonstrate the superiority of our model for improving recommendation performance. And to evaluate the explainability of our model, we first present examples to provide intuitive analysis on the highlighted review information, and then crowd-sourcing based evaluations are conducted to quantitatively verify our model's superiority.

Introduction

Explainable recommendation has attracted increasing attention in both industry and academic communities, because it can not only enhance customers' satisfaction, confidence and trust for the recommender system, but also can help them to make better and faster choices (Zhang and Chen 2018). With the desire to make a recommender system more interpretable, user reviews–which contain rich user preference–have been widely leveraged as an important resource to provide recommendation explanations, for example, EFM (Zhang et al. 2014) explained a recommendation by filling user cared features learned from the review information into pre-defined templates. LRPPM (Chen et al. 2016b) and MTER (Wang et al. 2018) extended EFM for more accurate user-item-feature explanations based on tensor factorization techniques. NARRE (Chen et al. 2018a) leveraged attention mechanism to extract valuable item reviews to explain the rating prediction process.

Despite effectiveness, these explainable recommender models still suffer from some inherent limitations. To begin with, most of them represent a user as a static latent vector, thus, they fail to capture user dynamic preference in the context of explainable recommendation, which may weaken the recommendation performance according to many previous studies (Tang and Wang 2018; Hidasi et al. 2016). More importantly, the explanations provided by these models are usually invariant, which is less effective in satisfying users' actually dynamic preference in real scenarios. For example, a user may care more about the socks' breathability in the beginning, and after a period of time, she may become more interested in the warmth because the weather started getting cold. The mismatching between the explanation and user preference may degrade costumers' satisfaction, confidence and trust for the recommender system.

Inspired by these limitations, in this paper, we design a novel **D**ynamic **E**xplainable **R**ecommender (called **DER**) for more accurate user modeling and recommendation explanations. We believe a persuasive explanation should accurately tell an item's properties (*what*) that the user (*who*) cares most at present (*when*). To solve the problem of *when*, gated recurrent unit (GRU) is selected as the basic architecture to capture user dynamic preference. A key problem of GRU is that it models a sequence without considering the time interval information between two successive steps, which is yet an important signal for user behavior modeling, e.g., a user tend to have similar preferences within a short time, while large time gap may have more opportunities to change user interest (Zhu et al. 2017). To solve this problem, we revise traditional GRU by adding a *time gate* to make it more applicable for the field of personalized recommendation. For *what*, we extract item properties from the review information based on sentence-level convolutional neural network (CNN). Finally, we adopt a "personalized attention mechanism" for merging different review information to address the issue of *who*. Based on the above designs, we predict the likeness from a user to an item, and also highlight valuable review information to explain the prediction according to the user's current preference.

[*]Corresponding author

In a summary, in this paper, we propose to model user dynamic preference in the context of explainable recommendation, and accordingly provide adaptive explanations tailored for different user states. To achieve this goal, we design a novel dynamic explainable recommender model by integrating gated recurrent unit (GRU) with attention mechanism. The carefully designed time-aware gated recurrent unit (GRU) is good at modeling user dynamic preference, while the attention mechanism facilitates explainable recommendation. By combining them, we hope to take advantages of the benefits from both components. We conduct extensive experiments to demonstrate the effectiveness of our model for the recommendation task of rating prediction. And also, we evaluate our model's explainability from both qualitative and quantitative perspectives.

Preliminaries

In this section, we first conduct primary analysis based on a real-world dataset to verify our assumption that: users' cared item aspects are always changeable in real scenarios. Then we formally define the task of explainable recommendation with user dynamic preferences.

Data Analysis

We base our analysis on the Amazon dataset[1], which is collected from www.amazon.com. To alleviate the influence of product diversity, we focus our analysis in the same category (we specified it as *Musical Instruments*). Intuitively, people usually talk about their cared product aspects in the review information (McAuley and Leskovec 2013; Zhang et al. 2014). To verify the dynamic properties of user preferences, we merge all the user reviews posted in the same month into a document, and further leverage Latent Dirichlet Allocation (LDA) (Blei, Ng, and Jordan 2003) to project each document into a topical distribution vector that reflects its relevance to different learned topics. The changes of the topic distribution in each month are presented in Figure 1, we can see: (1) On the whole, users' attention for different topics is dynamic. For example, in November 2008, users discussed more on the third topic (with the probability of 0.27), while in May 2011, the first topic became the most cared aspect (with the probability of 0.28). (2) For each topic, its received attention varies as the time changes. For example, the second topic (denoted in green circle) obtained the most attention in December 2008 (with the probability of 0.51), following which its attractiveness gradually decreased, and reached the lowest point in May 2014 (with the probability of 0.11).

These results manifest that users' cared product features are always changeable, which motivates us to provide adaptive explanations to satisfy user dynamic preferences.

Problem Definition

Suppose we have a user set $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and an item set $\mathcal{I} = \{i_1, i_2, ..., i_{|\mathcal{I}|}\}$. For each user, we chronologically organize his/her historical behaviors as a sequence

[1]http://jmcauley.ucsd.edu/data/amazon/links.html

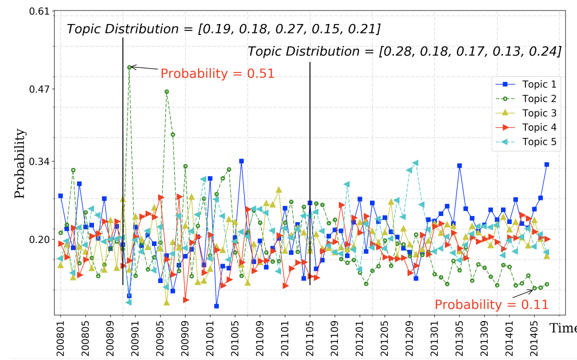

Figure 1: The dynamic nature of the user review topic distribution across different months. The number of topics in LDA is set as 5.

of quadruples $\mathcal{O}^u = \{(v_s^u, r_s^u, \boldsymbol{w}_s^u, t_s^u)\}_{s=1}^{l_u}$, where $t_1^u \leqslant t_2^u \leqslant \ldots \leqslant t_{l_u}^u$, v_s^u is an item in $\mathcal{I}$, and the sth element $(v_s^u, r_s^u, \boldsymbol{w}_s^u, t_s^u)$ means that user u interacted with item v_s^u at time t_s^u by rating r_s^u and review $\boldsymbol{w}_s^u$. Given all the user behaviors in the training set $\mathcal{O} = \{\mathcal{O}^u | u \in \mathcal{U}\}$, our task of explainable recommendation with user dynamic preference is to learn a predictive function f, such that for a user-item pair $(u, v_{l_u+1}^u)$ in the testing set, it can predict the rating $r_{l_u+1}^u$ that reflects the user's likeness towards the item at $t_{l_u+1}^u$. And further, its internal parameters or intermediate outputs should provide explanations for the final predicted rating according to the user's preference at $t_{l_u+1}^u$.

DER: Dynamic Explainable Recommender

The principle of our framework can be seen in Figure 2. User dynamic preference is modeled by a novel time-aware GRU architecture. By aggregating user historical behaviors, the output from the last GRU step encodes the user's current preference. The target item is profiled based on all its received user reviews, where each sentence is projected into a latent vector by a convolutional neural network (CNN). According to the user current preference, different sentence vectors are attentively merged into a unified embedding, which is then leveraged to derive the final rating. Once our model learned, the attention weights encode the importances of different review sentences, which can help us to explain the rating prediction process. In the following, we describe different components in our framework more in depth.

User Profiling based on Time-aware Gated Recurrent Unit

Gated recurrent unit (GRU) (Cho et al. 2014) is a powerful and efficient tool to model sequential features. The computational rule (see Figure 2(b)) of a traditional GRU at each time step can be concluded as:

$$\boldsymbol{z}_s = \sigma(\boldsymbol{W}_z[\boldsymbol{x}_s, \boldsymbol{h}_{s-1}]) \tag{1}$$

$$\boldsymbol{r}_s = \sigma(\boldsymbol{W}_r[\boldsymbol{x}_s, \boldsymbol{h}_{s-1}]) \tag{2}$$

$$\boldsymbol{h}'_s = \mathrm{Tanh}(\boldsymbol{W}_h[\boldsymbol{x}_s, \boldsymbol{r}_s \odot \boldsymbol{h}_{s-1}]) \tag{3}$$

$$\boldsymbol{h}_s = \boldsymbol{z}_s \odot \boldsymbol{h}_{s-1} + (1 - \boldsymbol{z}_s) \odot \boldsymbol{h}'_s \tag{4}$$

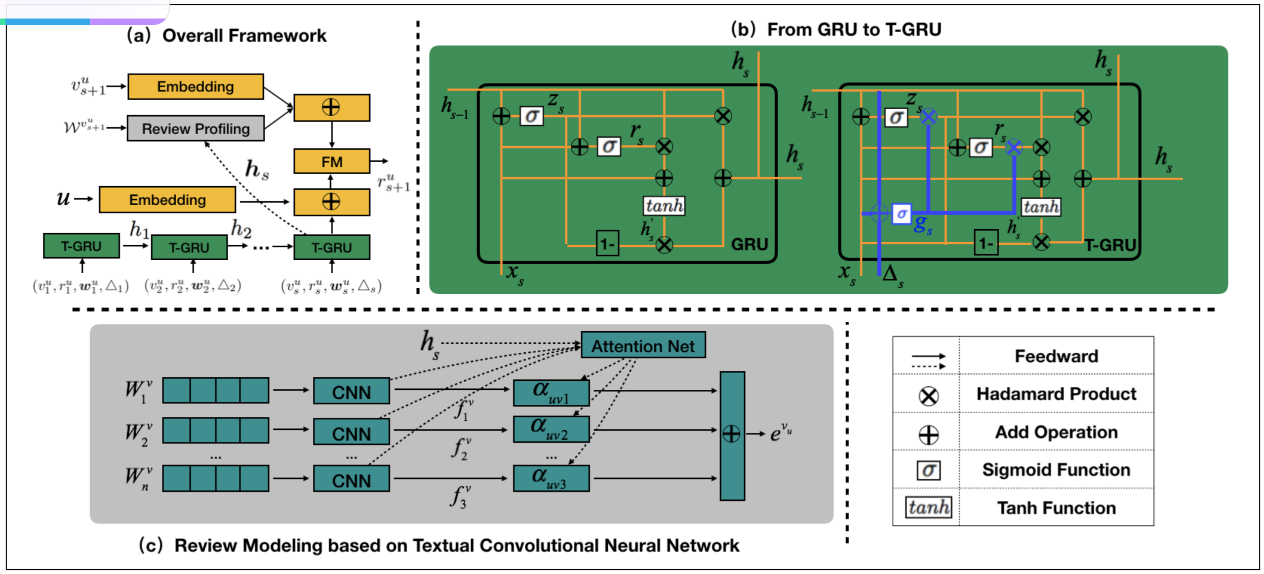

Figure 2: (a) The overall framework. (b) The comparison between traditional GRU and our designed T-GRU. The blue bold lines highlight the augmented operations. (c) The attention mechanism of merging different review sentences for the target item.

where $\boldsymbol{z}_s$ is the update gate and $\boldsymbol{r}_s$ is the reset gate, each of which is obtained by applying a sigmoid function $\sigma(\cdot)$ to the concatenated input. $\boldsymbol{h}'_s$ is the current memory content and $\boldsymbol{h}_s$ is the output hidden state. $\odot$ is Hadamard (element-wise) product and $[\cdot,\cdot]$ is concatenate operation. Tanh$(\cdot)$ is hyperbolic tangent activation function and $\boldsymbol{W}_z, \boldsymbol{W}_r, \boldsymbol{W}_h$ are parameters to be learned. Basically, in the process of sequence modeling, by multiplying with $\boldsymbol{h}_{s-1}$, the reset gate $\boldsymbol{r}_s$ determines what to remove from the previous steps, *e.g.*, if $\boldsymbol{r}_s$ is close to $\mathbf{0}$, GRU will wash out the past information and focus only on the current input $\boldsymbol{x}_s$. At the same time, the update gate $\boldsymbol{z}_s$ determines what to collect from the current memory content $\boldsymbol{h}'_s$ and what from the previous steps $\boldsymbol{h}_{s-1}$. For example, if $\boldsymbol{z}_s$ is close to $\mathbf{1}$, then $\mathbf{1} - \boldsymbol{z}_s$ will be near $\mathbf{0}$. GRU will keep a majority of the previous information $\boldsymbol{h}_{s-1}$, and ignore big portion of the current content $\boldsymbol{h}'_s$.

GRU has been demonstrated to be effective in many neural language processing (NLP) tasks (Cho et al. 2014; Zhou et al. 2017; Noh, Hongsuck Seo, and Han 2016). In our problem, we can straightforwardly regard a user behavior as a word, and the whole behavior sequence as a sentence. However, an important difference between user behavior modeling (UBM) and neural language processing (NLP) is that: the time interval information between two successive steps, which is meaningless in NLP, is an important signal in UBM. Intuitively, a user tend to have similar preferences within a short time, while large time gap may decrease the influence of the former action. To seamlessly embed this insight into our model, we redesign the architecture of GRU. To begin with, we introduce a new time gate $\boldsymbol{g}_s$, which is jointly determined by the current input $\boldsymbol{x}_s$, the previous information $\boldsymbol{h}_{s-1}$ and the time interval $\triangle_s$, that is: $\boldsymbol{g}_s = \sigma(\boldsymbol{W}_g[\boldsymbol{x}_s, \boldsymbol{h}_{s-1}] + \boldsymbol{\lambda}\triangle_s)^2$, where, $\triangle_s = t_{s+1} - t_s$ is the time interval between step s and $s+1$, $\boldsymbol{\lambda} > \mathbf{0}$ (meaning each element in $\boldsymbol{\lambda}$ is larger than 0) is a pre-defined hyper-parameter. $\boldsymbol{W}_g$ is a weighting matrix to be learned. Obviously, each element in $\boldsymbol{g}_s$ is monotonically increasing at $\triangle_s$, which means larger time interval leads to larger $\boldsymbol{g}_s$, and vice versa.

On one hand, we hope a small time interval will not change user preference much, it means if $\triangle_s$ is small, user preference at t_{s+1} should be similar to the one at t_s, which is profiled by $\boldsymbol{x}_s$. On the other hand, we hope long time interval will have the user return to previous long-term preference, it means if t_{s+1} is much larger than t_s, the influence of the current behavior $\boldsymbol{x}_s$ should be little. The user preference tend to be consistent with the historical behaviors embedded in $\boldsymbol{h}_{s-1}$. Based on these analysis, the time gate $\boldsymbol{g}_s$ is infused into equation (3) as:

$$\boldsymbol{h}'_s = \mathrm{Tanh}(\boldsymbol{W}_h[\boldsymbol{x}_s, \underline{\boldsymbol{g}_s} \odot \boldsymbol{r}_s \odot \boldsymbol{h}_{s-1}]) \tag{5}$$

where if the time interval $\triangle_s$ is small (meaning small $\boldsymbol{g}_s$), the influence of the historical information $\boldsymbol{h}_{s-1}$ will be lowered by multiplying $\boldsymbol{g}_s$, and the current input $\boldsymbol{x}_s$ will take the leading role, and vice versa. Similarly, we revise equation (4) as:

$$\boldsymbol{h}_s = \underline{\boldsymbol{g}_s \odot \boldsymbol{z}_s} \odot \boldsymbol{h}_{s-1} + (1 - \underline{\boldsymbol{g}_s \odot \boldsymbol{z}_s}) \odot \boldsymbol{h}'_s \tag{6}$$

where the user preference at t_{s+1} (*i.e.*, $\boldsymbol{h}_s$) will keep a majority of the long-term preference encoded in $\boldsymbol{h}_{s-1}$, if the next action happens after a large time interval $\triangle_s$ (corresponding to large $\boldsymbol{g}_s$), and vice versa. The overall architecture of the modified GRU can be seen in Figure 2(b), we call it T-GRU for short.

For comprehensive user profiling, in our model, $\boldsymbol{x}_s$ not only includes the interacted item ID, but also contains the corresponding rating and review information, that is:

$$\boldsymbol{x}_s = [\boldsymbol{E}_V v^u_s, \boldsymbol{E}_R r^u_s, \boldsymbol{W}^u_s] \tag{7}$$

[2]For simplicity, the label for discriminating different users' sequential behaviors is omitted

where, v_s^u and r_s^u are one-hot representations for the interacted item and its received rating, respectively. $\boldsymbol{E}_V$ and $\boldsymbol{E}_R$ are weighting matrices that project v_s^u and r_s^u into embedding vectors. $\boldsymbol{W}_s^u$ is the representation of the review from u to v_s^u, which simply averages the pre-trained embeddings of the words in $\boldsymbol{w}_s^u$ [3].

Item Profiling based on Textual Convolutional Neural Network

People usually talk about different product features in their reviews. Comparing with item ID, such information can provide more explicit and comprehensive signals for collaborative modeling. Similar to many previous work (Zheng, Noroozi, and Yu 2017; Chen et al. 2018a), the review information in our model is processed based on the convolutional neural network (CNN) (Krizhevsky, Sutskever, and Hinton 2012). However, instead of taking each user review as a whole, we process it on the sentence-level to allow more flexible explanations.

Formally, suppose there are n sentences in all the reviews of an item defined by: $\mathcal{W} = \{\mathcal{W}_1, \mathcal{W}_2, ..., \mathcal{W}_n\}$[4]. Let $\boldsymbol{c}^k = \{\boldsymbol{c}_1^k, \boldsymbol{c}_2^k, ..., \boldsymbol{c}_T^k\}$ be the word embedding list of $\mathcal{W}_k$, where $\boldsymbol{c}_i^k \in \mathbb{R}^d$ is the pre-trained word embedding for the ith word in $\mathcal{W}_k$. Suppose there are m filters in our CNN architecture, each of which is associated with a parameter $\boldsymbol{K}_j \in \mathbb{R}^{d\times t}$, where t is the window size (typically set as 2-5). The filter processes t-length windows of d-dimensional vectors to produce local features, and the output from the lth window of the jth filter is:

$$\begin{aligned} o_{kj}^l &= \text{ReLU}(\boldsymbol{c}_{l:(l+t-1)}^k * \boldsymbol{K}_j + b_j) \\ l &\in \{1, 2, ..., T-t+1\} \end{aligned} \tag{8}$$

where $\boldsymbol{c}_{l:(l+t-1)}^k = \{\boldsymbol{c}_l^k, \boldsymbol{c}_{l+1}^k, ..., \boldsymbol{c}_{l+t-1}^k\}$ is the lth window to be processed. b_j is the bias, $\text{ReLU}(x) = \max\{0, x\}$ is the active function (Nair and Hinton 2010), and $*$ is the Frobenius inner product operation. Given the output from the jth filter, max-pooling operation is conducted to select the most salient feature, *i.e.*, the one with the highest value, that is: $f_{kj} = \max\{o_{kj}^1, o_{kj}^2, ..., o_{kj}^{T-t+1}\}$. At last, the final embedding for the kth sentence $\mathcal{W}_k$ is: $\boldsymbol{f}_k = [f_{k1}, f_{k2}, ...f_{km}]$.

Explainable Rating prediction

Previous methods (Chen et al. 2018a; Seo et al. 2017; Wu et al. 2017) usually interpret a recommendation in user-independent manners. However, in real scenarios, people usually have different personalities and may care about different aspects even for the same item. This intuition motivates us to explain the recommendations in a personalized manner.

Based on the above sections, suppose a user u's dynamic preference at t_{s+1}^u is encoded in $\boldsymbol{h}_s^u \in \mathbb{R}^h$, and the different review sentences of an item v are represented as $\boldsymbol{f}_1^v, \boldsymbol{f}_2^v, ..., \boldsymbol{f}_n^v$. For personalized explanation, we profile the item properties reflected in review information by merging sentence embeddings under "user-aware" attention weights as: $\boldsymbol{e}^{v_u} = \sum_{k=1}^n \alpha_{uvk} \cdot \boldsymbol{f}_k^v$, where α_{uvk} is derived from an attention net as:

$$\begin{aligned} a_{uvk} &= \boldsymbol{W}_2\text{ReLU}(\boldsymbol{W}_1((\boldsymbol{W}_h\boldsymbol{h}_s^u) \odot (\boldsymbol{W}_f\boldsymbol{f}_k^v)) + \boldsymbol{b}_1) + \boldsymbol{b}_2 \\ \alpha_{uvk} &= \frac{\exp(a_{uvk}/\tau)}{\sum_{k'=1}^h \exp(a_{uvk'}/\tau)} \end{aligned} \tag{9}$$

where $\boldsymbol{W}_h$, $\boldsymbol{W}_f$ are weighting parameters that project $\boldsymbol{h}_s^u$ and $\boldsymbol{f}_k^v$ into the same space, $[\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2]$ are parameters of the attention net, τ is the pre-defined temperature parameter. Once the model learned, α_{uvk}, which has taken user dynamic preference into consideration, can be used to explain the rating prediction process by highlighting different sentence importances.

For more robust user and item profiling, we further introduce auxiliary embeddings $\boldsymbol{e}^u \in \mathbb{R}^K$ and $\boldsymbol{e}^v \in \mathbb{R}^K$ to derive the final rating from u to v, that is:

$$\hat{r}_{uv} = \text{FM}([(\boldsymbol{e}^u + \boldsymbol{H}_h\boldsymbol{h}_s^u),\ (\boldsymbol{e}^v + \boldsymbol{H}_e\boldsymbol{e}^{v_u})]) \tag{10}$$

where $\boldsymbol{H}_h \in \mathbb{R}^{K\times h}, \boldsymbol{H}_e \in \mathbb{R}^{K\times m}$ are weighting parameters, $[\cdot,\cdot]$ is the concatenate operation, and $\text{FM}(\cdot)$ is the factorization machine layer, which is effective in capturing nested variable interactions between the heterogeneous information learned from user reviews (*e.g.* $\boldsymbol{h}_s^u$ and $\boldsymbol{e}^{v_u}$) and the other ones (*e.g.* $\boldsymbol{e}^u$ and $\boldsymbol{e}^v$) (Zheng, Noroozi, and Yu 2017). At last, our final objective function to be minimize is:

$$L = \sum_{(u,v)\in\mathcal{R}} (r_{uv} - \hat{r}_{uv})^2 + \gamma \sum_{\theta\in\Theta} ||\theta||_2^2 \tag{11}$$

where $\mathcal{R}$ is the set of user-item pairs in the training set. r_{uv} is the real rating from user u to item v. Θ is the set of parameters to be regularized. In this equation, the first term is used to minimize the distance between the predicted and the real ratings, while the second term aims to regularize the parameters to avoid over fitting.

Experiments

In this section, we evaluate our model by comparing it with several state-of-the-art models. We begin by introducing the experimental setup, and then report and analyze the experimental results.

Experiment Setup

Datasets We use two publicly available datasets from different domains to evaluate our models, that is:

• **Amazon**[5]: This dataset contains user rating and review information for different products on www.amazon.com. According to the product category, the raw data is divided into 24 subsets. To cover different data characters, we select three categories in our experiments, that is, *Musical Instruments*, *Automotive* and *Toy*. In specific, *Toy* is a larger and sparser dataset, while *Musical Instruments* and *Automotive* are smaller, but denser.

[3] It should be noted that more advanced methods, such as the hierarchical model (Cheng and Lapata 2016), can also be leveraged to derive $\boldsymbol{W}_s^u$, which are left for future exploration.

[4] For simplicity, the label for discriminating different items is omitted

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

Table 1: Statistics of the datasets. *Musical Instruments* is abbreviated as Music.

Datasets	#User	#Item	#Interaction	Density
Music	1429	900	10245	0.797%
Automotive	2928	1835	20441	0.380%
Toy	19412	11924	167472	0.072%
Yelp	34547	47010	1523939	0.094%

• **Yelp**[6]: This is a large-scale dataset including users' rating and review behaviors for different restaurants. Because the raw data is very large, we pre-process it by removing the users and items with less than 20 ratings. The statistics of these datasets can be seen in Table 1.

Evaluation method and baselines Root Mean Square Error (RMSE) is leveraged in our experiments to evaluate different models. Suppose the predicted and real ratings from u to v are $\hat{r}_{uv}$ and r_{uv}, respectively. The RMSE score is calculated by:

$$\text{RMSE} = \sqrt{\frac{1}{|\mathcal{T}|}\sum_{(u,v)\in\mathcal{T}}(r_{uv}-\hat{r}_{uv})^2}, \qquad (12)$$

where $\mathcal{T}$ is the set of user-item pairs in the testing set, and lower RMSE score means better performance. In our experiments, the following representative models are selected as the baselines:

- **PMF:** This is a traditional matrix factorization method (Mnih and Salakhutdinov 2008), and the model parameters are learned by stochastic gradient decent (SGD).
- **GRU4Rec:** This is a well known sequential recommender model (Hidasi et al. 2016), where each previously interacted item is accordingly fed into each time step.
- **Time-LSTM** This is a time-aware sequential recommender method, where the time interval information (Zhu et al. 2017) is incorporated in the modeling process.
- **Time-LSTM++:** This method is an advanced version of Time-LSTM, where the input of each step not only contains item ID, but also includes review and rating information as in equation 7 for more comprehensive user profiling.
- **NARRE:** This is a state-of-the-art explainable recommendation method (Chen et al. 2018a), which has been verified to outperform many promising algorithms including NMF, SVD++, HFT and DeepCoNN on Amazon and Yelp datasets. We implemented it based on the authors' public code[7].

Implementation details For each user behavior sequence, the last and second last interactions are used for testing and validation, while the other interactions are left for training. In our model, the batch size as well as the learning rate are determined in the range of $\{50, 100, 150\}$ and $\{0.001, 0.01, 0.1, 1\}$, respectively. The user/item embedding size K is tuned in the range of $\{8, 16, 32, 64, 128\}$, and we will discuss its influence on the model performance in the following sections. For the user review information, we first pre-process it based on the Stanford Core NLP tool[8], and then the word embeddings are pre-trained based on the Skip-gram model[9]. The baselines designed for Top-N recommendation are revised to optimize the RMSE score.

[6]https://www.kaggle.com/yelp-dataset/yelp-dataset/data

[7]https://github.com/THUIR/NARRE

Table 2: The results of comparing our model with the baselines in terms of RMSE.

Dataset	Music	Automotive	Toy	Yelp
PMF	1.0706	1.0100	1.1220	1.3411
GRU4Rec	1.0111	0.9723	1.0363	1.3011
Time-LSTM	0.9901	0.9615	0.9963	1.2821
Time-LSTM++	0.9878	0.9435	0.9805	1.2711
NARRE	0.9784	0.9199	0.9690	1.2507
DER	**0.9678**	**0.8981**	**0.9535**	**1.2314**

Evaluation on Rating Prediction

Overall performance From the results shown in Table 2, we can see: the simple PMF method performed worst because it fails to capture the sequential properties for user behavior modeling, and also cannot borrow the power of review information to enhance the user/item representations. Time-LSTM and Time-LSTM++ performed better than GRU4Rec, which is consistent with the previous study (Zhu et al. 2017), and verifies the effectiveness of time interval information for user dynamic preference modeling. NARRE outperformed Time-LSTM++, and the reason can be that, for a target item, NARRE utilizes all its review information to provide informative signals to assist the rating prediction process, and the attention mechanism further make it powerful to discriminatively enhance the impact of the valuable review information, while reducing the noise influence. Encouragingly, DER consistently performed better than the best baseline NARRE on all the datasets. Comparing with NARRE, which represents each user as a static embedding, the carefully designed T-GRU architecture enables us to accurately model user dynamic preference, which facilitates more adaptive and reasonable user profiling, and eventually leads to improved rating prediction.

Influence of the embedding size K. In this section, we investigate how the embedding size influences our model's performance, and due to the space limitation, unless specified, we only report the results on the Automotive dataset. We observe the performance changes by tuning the embedding size K in the range of $\{8, 16, 32, 64, 128\}$. From the result presented in Figure 3, we can see: our model achieved the best performances when the embedding size was relative small (*i.e.*, $K = 8$) , while larger K didn't help to further improve the results. This observation actually agrees with many previous studies (Li et al. 2016; Zhang et al. 2017;

[8]https://stanfordnlp.github.io/CoreNLP/

[9]http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

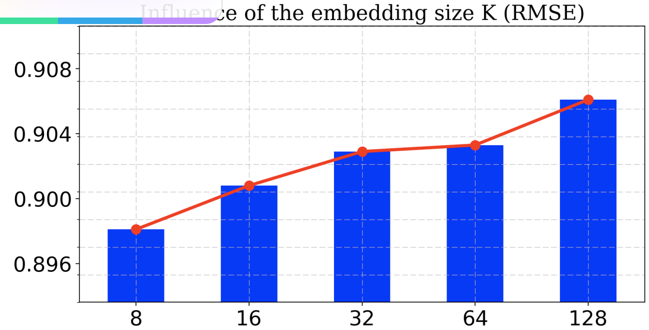

Figure 3: Influence of the embedding size K for our model's final performance

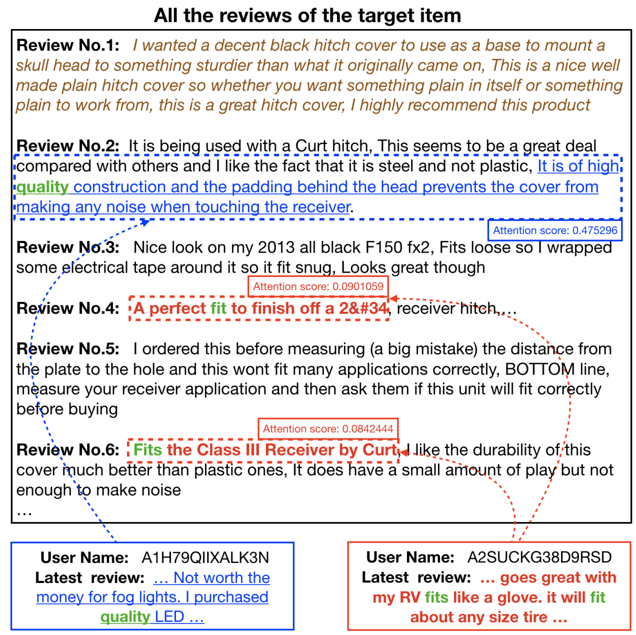

Figure 4: The example of an item's highlighted reviews for two different users. The most important review learned by NARRE is labeled by italic, and the sentences selected by our model are presented in underlined and bold fonts for different users, respectively. The latest reviews from these two users are listed below for reference, and the green words indicate similar preferences.

Chen et al. 2016a), and the reason can be that: in our dataset, a small number of parameters are enough for capturing user different behavior patterns, using redundant dimensions will increase the model complexity and over fit the training set, which may degrade our model's generalization capability on the testing set.

Evaluation on Recommendation Explainability

As described by the previous work (Chen et al. 2018a), the review information of a target item can provide users with detailed information and suggestions to make informed decisions. By presenting valuable review sentences for the recommended items, the transparency and explainability of the recommender system can be improved. In this section, we evaluate our highlighted review information from both qualitative and quantitative perspectives based on the Automotive dataset.

Qualitative evaluation To provide intuitive analysis about our model's explainability, we present an example in Figure 4. In specific, we compare the highlighted review information of a target item by NARRE and our model for different users. To evaluate our model's capability on dynamic modeling, we also list the latest reviews from these users for reference.

We can see: NARRE highlighted the same review for different users due to its user-independent attention mechanism, while our model can highlight personalized review information, which is more practical and reasonable in real scenarios. With the help of sentence-level attention mechanism, our model can flexibly extract useful information across different reviews to form the final explanations. For example, for the user 'A2SUCKG38D9RSD', our model highlighted the sentences, which are both about the aspect of "fit", from the 4th and 6th reviews. At last, the review information highlighted by our model can effectively track user recent preference, for example, user 'A1H79QIIXALK3N' recently expressed some opinions about the "quality" aspect, which is also highlighted by our model for the current recommendation. However, NARRE didn't exhibit such capability. This observation, to some extent, verifies our model's superiority for learning user dynamic preference in the context of explainable recommendation.

Quantitative evaluation To quantitatively evaluate our model's explainability, we conduct crowd-sourcing evaluation by comparing our model with NARRE. In our model, we highlight two most valuable sentences according to the learned attention weights (*i.e.* α_{uvk}), while in NARRE, we predict the most important review for the target item. For each user-item pair, the workers are provided with two information sources: (1) The user's real review for the current item, and (2) All the reviews recently posted by the user. Based on these information, the workers are asked to select one from three options (*i.e.*, A:DER, B:NARRE, C:Tie) to answer the following questions:

- **Q1:** Which model can highlight more accurate information as compared with user real reviews?
- **Q2:** For each user, which model can highlight textual information that is more consistent with her recent reviews?

For more accurate evaluation, we employed 3 workers, and one result was valid only when more than 2 workers shared the same opinion. From the result of **Q1** shown in Figure 5, we can see: by comparing with user real reviews, our model was more accurate than NARRE. We speculate that the personalized attention mechanism in our model can discover tailed information for different users, while NARRE only highlights the same review for them, which is limited in capturing user different personalities revealed in the ground truth. The result in **Q2** manifests that the review

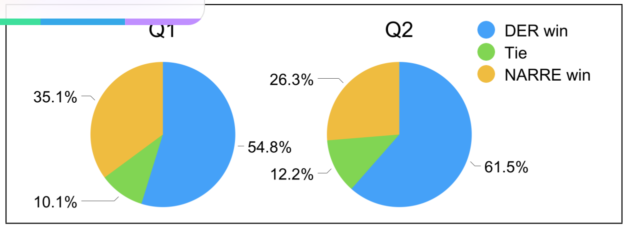

Figure 5: Results of the quantitative evaluation on the recommendation explanations.

information highlighted by our model can be much more effective in tracking user recent preference, which verifies the effectiveness of our designed dynamic architecture.

Related Work

Explainable Recommendation

Explainable recommendation is becoming more and more popular in both research and industry communities (Zhang and Chen 2018), and many promising models have been proposed in the recent years. More specifically, early methods (Zhang et al. 2014; Chen et al. 2016b) mainly based themselves on the combination between matrix factorization (MF) and sentiment analysis (SA). In a nutshell, they first extracted feature-opinion-sentiment triplets from the user review information, are then infused them into MF for collective user preference modeling. At last, the explanations were provided by filling the predicted user cared features into pre-defined templates. Despite effectiveness, the final results of these models may be limited by the accuracy of the review preprocessing tools, and it may also be less efficient in practice due to the complex process for extracting feature-opinion-sentiment triplets. Recently, with the ever prospering of deep learning technology, many algorithms (Seo et al. 2017; Chen et al. 2018a; Tay, Tuan, and Hui 2018) were designed to explain a recommendation based on the attention mechanism. Basically, in these models, the raw user review information related to a user (or an item) was merged into a document, and by attentively discovering valuable information in the document, the explanations were provided by highlighting the words with the highest attention weights. In particularly, D-Attn (Seo et al. 2017) and NARRE (Chen et al. 2018a) automatically learned the importances of different review sentences under the supervision of user-item rating information. For providing tailored explanations for different target items, MPCN (Tay, Tuan, and Hui 2018) leveraged ”co-attention” mechanism to capture the correlations between the users and the items. In addition to user-review explanations, Ai et al (Ai et al. 2018) conducted explainable recommendation by reasoning over knowledge graph embeddings, where explanation paths between user and item were constructed to generate knowledge-enhanced explanations.

Although these models have achieved promising results, they failed to model user dynamic preference, and the provided explanations were usually static at different times, which may weaken the persuasiveness of the explanations as mentioned before.

Recommendation based on User Dynamic Preference

Recently, many models have been designed to incorporate temporal information into recommender system to capture user dynamic preference. In specific, early methods care more about transition properties between two successive behaviors. For instance, the factorized personalized Markov chains (FPMC) (Rendle, Freudenthaler, and Schmidt-Thieme 2010) combined matrix factorization with one-order Markov chain to capture the influence of the last behavior towards the next one. The hierarchical representation model (HRM) (Wang et al. 2015) generalized FPMC into a representation learning framework, and significantly improved the recommendation performance. The major limitation of these methods lies in the ignoring of long-term preference dependency. To solve this problem, many models were proposed to capture user multi-step behaviors based on the recurrent neural network (RNN) (Yu et al. 2016; Hidasi et al. 2016; Tan, Xu, and Liu 2016; Donkers, Loepp, and Ziegler 2017; Liu et al. 2016; Song, Elkahky, and He 2016), the convolutional neural network (CNN) (Tang and Wang 2018) or the memory network (Chen et al. 2018b). Basically, these methods attempt to transfer the power of deep learning in sequence modeling to the field of recommender system, while an important information–the time interval between two adjacent behaviors–has been totally ignored. Recently, Zhu et al (Zhu et al. 2017) designed a model called Time-LSTM to demonstrated the importance of time interval information for user dynamic preference modeling. Different from this method, our model is built upon a more simply and lightweight architecture–GRU, which can be more effective for practical recommender systems. More importantly, our model is able to incorporate side information (*i.e.*, user reviews) for explainable recommendations.

Conclusion

In this paper, we propose to model user dynamic preference in the context of explainable recommendation. Based on our designed model, we can improve the performance of user-item rating prediction as compare with several state-of-the-art methods, and more importantly, we can provide adaptive recommendation explanations according to the user dynamic preference. We conduct extensive experiments to demonstrate the superiority of our model.

As for future work, we will investigate the potential advantages of stochastic process (*e.g.*, point process (Yan et al. 2018)) for user dynamic preference modeling in the context of explainable recommendation, where we may focus on two questions: one is how to seamlessly equip stochastic process with side information (*e.g.*, user reviews), and the other is how to relate stochastic process with attention mechanism for explaining the recommendations. As there is an emerging trend to leverage user visual preference for enhancing recommendation performance, in the future, we will also study how to integrate product images into our model for more comprehensive recommendation explanations.

References

Ai, Q.; Azizi, V.; Chen, X.; and Zhang, Y. 2018. Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms.*

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research*.

Chen, C.; Li, D.; Lv, Q.; Yan, J.; Chu, S. M.; and Shang, L. 2016a. Mpma: Mixture probabilistic matrix approximation for collaborative filtering. In *IJCAI*.

Chen, X.; Qin, Z.; Zhang, Y.; and Xu, T. 2016b. Learning to rank features for recommendation over multiple categories. In *SIGIR*.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2018a. Neural attentional rating regression with review-level explanations. In *WWW*.

Chen, X.; Xu, H.; Zhang, Y.; Tang, J.; Cao, Y.; Qin, Z.; and Zha, H. 2018b. Sequential recommendation with user memory networks. In *WSDM*.

Cheng, J., and Lapata, M. 2016. Neural summarization by extracting sentences and words. In *ACL*.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Donkers, T.; Loepp, B.; and Ziegler, J. 2017. Sequential user-based recurrent neural network recommendations. In *Recsys*.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2016. Session-based recommendations with recurrent neural networks. *ICLR*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

Li, D.; Chen, C.; Lv, Q.; Yan, J.; Shang, L.; and Chu, S. 2016. Low-rank matrix approximation with stability. In *ICML*.

Liu, Q.; Wu, S.; Wang, D.; Li, Z.; and Wang, L. 2016. Context-aware sequential recommendation. In *ICDM*.

McAuley, J., and Leskovec, J. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Recsys*.

Mnih, A., and Salakhutdinov, R. R. 2008. Probabilistic matrix factorization. In *NIPS*.

Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML*.

Noh, H.; Hongsuck Seo, P.; and Han, B. 2016. Image question answering using convolutional neural network with dynamic parameter prediction. In *CVPR*.

Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*.

Seo, S.; Huang, J.; Yang, H.; and Liu, Y. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *Recsys*.

Song, Y.; Elkahky, A. M.; and He, X. 2016. Multi-rate deep learning for temporal recommendation. In *SIGIR*.

Tan, Y. K.; Xu, X.; and Liu, Y. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on DLRS*.

Tang, J., and Wang, K. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*.

Tay, Y.; Tuan, L. A.; and Hui, S. C. 2018. Multi-pointer co-attention networks for recommendation. *arXiv preprint arXiv:1801.09251*.

Wang, P.; Guo, J.; Lan, Y.; Xu, J.; Wan, S.; and Cheng, X. 2015. Learning hierarchical representation model for nextbasket recommendation. In *SIGIR*.

Wang, N.; Wang, H.; Jia, Y.; and Yin, Y. 2018. Explainable recommendation via multi-task learning in opinionated text data. *SIGIR*.

Wu, L.; Quan, C.; Li, C.; Wang, Q.; and Zheng, B. 2017. A context-aware user-item representation learning for item recommendation. *arXiv preprint arXiv:1712.02342*.

Yan, J.; Liu, X.; Shi, L.; Li, C.; and Zha, H. 2018. Improving maximum likelihood estimation of temporal point process via discriminative and adversarial learning. In *IJCAI*.

Yu, F.; Liu, Q.; Wu, S.; Wang, L.; and Tan, T. 2016. A dynamic recurrent model for next basket recommendation. In *SIGIR*.

Zhang, Y., and Chen, X. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192*.

Zhang, Y.; Lai, G.; Zhang, M.; Zhang, Y.; Liu, Y.; and Ma, S. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*.

Zhang, Y.; Ai, Q.; Chen, X.; and Croft, W. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. *CIKM*.

Zheng, L.; Noroozi, V.; and Yu, P. S. 2017. Joint deep modeling of users and items using reviews for recommendation. In *WSDM*.

Zhou, H.; Huang, M.; Zhang, T.; Zhu, X.; and Liu, B. 2017. Emotional chatting machine: emotional conversation generation with internal and external memory. *arXiv preprint arXiv:1704.01074*.

Zhu, Y.; Li, H.; Liao, Y.; Wang, B.; Guan, Z.; Liu, H.; and Cai, D. 2017. What to do next: Modeling user behaviors by time-lstm. In *IJCAI*.