Explainable Recommendation through Attentive Multi-View Learning

Jingyue Gao,[1,2] **Xiting Wang,**[2*] **Yasha Wang,**[1] **Xing Xie**[2]
[1]Peking University, {gaojingyue1997, wangyasha}@pku.edu.cn
[2]Microsoft Research Asia, {xitwan, xingx}@microsoft.com

Abstract

Recommender systems have been playing an increasingly important role in our daily life due to the explosive growth of information. Accuracy and explainability are two core aspects when we evaluate a recommendation model and have become one of the fundamental trade-offs in machine learning. In this paper, we propose to alleviate the trade-off between accuracy and explainability by developing an explainable deep model that combines the advantages of deep learning-based models and existing explainable methods. The basic idea is to build an initial network based on an explainable deep hierarchy (e.g., Microsoft Concept Graph) and improve the model accuracy by optimizing key variables in the hierarchy (e.g., node importance and relevance). To ensure accurate rating prediction, we propose an attentive multi-view learning framework. The framework enables us to handle sparse and noisy data by co-regularizing among different feature levels and combining predictions attentively. To mine readable explanations from the hierarchy, we formulate personalized explanation generation as a constrained tree node selection problem and propose a dynamic programming algorithm to solve it. Experimental results show that our model outperforms state-of-the-art methods in terms of both accuracy and explainability.

Introduction

Personalized recommendation has become one of the most effective techniques to help users sift through massive amounts of web content and overcome information overload. Recently, the recommendation community has reached a consensus that **accuracy** can only be used to partially evaluate a system. **Explainability** of the model, which is the ability to provide explanations for why an item is recommended, is considered equally important (Wang et al. 2018a). Appropriate explanations may persuade users to try the item, increase users' trust, and help users make better decisions (Zhang et al. 2014a).

Determining whether to optimize towards accuracy or explainability poses a fundamental dilemma for practitioners. Currently, their choices are limited to two types of models:

- **Deep but unexplainable.** Deep learning-based recommendation models (Wang et al. 2017; Zheng, Noroozi, and Yu 2017) achieve state-of-the-art accuracy due to their capacity to model complex high-level features. However, the high-dimensional high-level features they learn are beyond the comprehension of ordinary users.
- **Explainable but shallow.** Although many explainable recommendation methods have been proposed, the explainable components in the models are usually shallow. Typical explainable components include one or two layers of attention networks (Chen et al. 2017), matrix factorization (Zhang et al. 2014a), and topic modeling (McAuley and Leskovec 2013). They have achieved considerable success in improving explainability. However, the lack of an effective mechanism to model high-level explicit features limits their accuracy and/or explainability. These methods can only select explanations from a pre-defined level of candidates. They cannot identify which level of features best represents a user's true interest. For example, they are not able to tell whether a user only likes *shrimps* (lower level) or is interested in *seafood* (higher level).

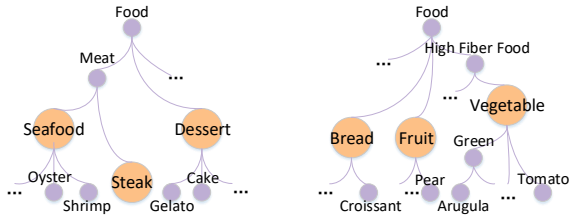

Figure 1: Multi-level user interest extracted using our method. The hierarchies correspond to a 26-year-old female Yelp user (left) and a 30-year-old male Yelp user (right). The features users care most about are highlighted in orange.

In this paper, we aim to mitigate the trade-off between accuracy and explainability by developing an **explainable deep model** for recommendation. The model achieves state-of-the-art accuracy and is highly explainable. Moreover, it enables us to accurately portray hierarchical user interest. As shown in Figure 1, the model can automatically infer multi-level user profiles and identify which level of features best captures a user's true interest, e.g., whether s/he is interested in lower-level features such as *shrimp* or higher-level features such as *seafood*.

*Xiting Wang is the corresponding author

To design an explainable deep model as such, we are faced with two major technical challenges. The first challenge is to accurately model multi-level explicit features from noisy and sparse data. It is very difficult to model the relationships between high-level and low-level features since they have overlapping semantic meanings. For example, learning whether *seafood* is important to the user is challenging because the user may only mention *shrimp* or *meat* in the review. The second challenge is to generate explanations that are easy for common users to understand from the multi-level structure.

To address these challenges, we develop a Deep Explicit Attentive Multi-View Learning Model (DEAML). The basic idea is to build an initial network based on an explainable deep structure (e.g., knowledge graph) and improve accuracy by optimizing key variables in the explainable structure (e.g., node importance and relevance). To improve model accuracy, we **propose an attentive multi-view learning framework** for rating prediction. In this framework, different levels of features are considered as different views. Adjacent views are connected by using the attention mechanism. Results from different views are co-regularized and attentively combined to make the final prediction. This framework helps improve accuracy since it is robust to noise and enables us to fully leverage the hierarchical information in the explainable structure. Second, we **formulate personalized explanation generation as a constrained tree node selection problem**. To solve this problem, we propose a dynamic programming algorithm, which finds the optimal features for explanation in a bottom-up manner.

We conduct two experiments and a user study to evaluate our method. Numerical experiments show that DEAML outperforms state-of-art deep learning-based recommendation models in terms of accuracy. An evaluation with 20 human subjects has shown that the explanations generated by us are considered significantly more useful than state-of-the-art aspect-based explainable recommendation method.

Related Work

Many methods have been proposed to improve recommendation accuracy, including content-based (Kompan and Bieliková 2010), collaborative filtering-based (Das et al. 2007) and hybrid methods (De Francisci Morales, Gionis, and Lucchese 2012). As an integration and extension of previous methods, deep learning-based models are proposed to further improve accuracy. For example, CDL (Wang, Wang, and Yeung 2015) jointly performs deep representation leaning and collaborative filtering by employing a hierarchical Bayesian model. He et al. propose a Neural Collaborative Filtering framework to learn nonlinear interactions between users and items (He et al. 2017). In DeepCoNN and NARRE (Zheng, Noroozi, and Yu 2017; Chen et al. 2018), convolutional neural networks are leveraged to extract features from textual reviews. Although these methods achieve significant improvement in accuracy, the high-level features extracted are beyond the understanding of common users. Except for unstructured textual data, researchers have also designed models that leverage structural information. Famous examples are taxonomy-based methods that incorporate taxonomies of items into latent factor models (Kanagal et al. 2012; Zhang et al. 2014b). These methods can effectively mitigate data sparsity. However, their accuracy and explainability are limited due to the lack of a mechanism to model multi-level item features.

Recently, researchers have discovered that providing explanations may improve persuasiveness, effectiveness, efficiency and user trust (Zhang et al. 2014a). Thus, many methods have been developed to improve the explainability of recommendation models (Ren et al. 2017; Peake and Wang 2018; Wang et al. 2018b). Pioneer works (McAuley and Leskovec 2013; Zhang et al. 2014a) focus on improving the explainability of collaborative filtering models. These works usually rely on shallow explainable components such as matrix factorization (Zhang et al. 2014a) and generative models (Diao et al. 2014; Wu and Ester 2015) for explanation generation. More recently, researchers discovered the attention mechanism's capability in improving the explainability of deep learning-based methods. By using the attention mechanism, the models can automatically learn the importance of explicit features and at the same time refine user and/or item embeddings (Chen et al. 2017; 2018). Researchers have also built explainable models based on interpretable structures such as graphs (He et al. 2015) and trees (Wang et al. 2018a).

The aforementioned explainable recommendation methods have achieved considerable success in improving explainability. However, to the best of our knowledge, none of the existing explainable recommendation methods can model multi-level explicit (explainable) features. As a result, the accuracy and/or explainability of these methods are limited. Compared with these methods, our DEAML is an explainable deep model that achieves state-of-the-art accuracy and meanwhile is highly explainable. Moreover, our model can automatically learn multi-level user profile and infer which level of features best captures a user's interest.

Problem Definition

We aim to build an explainable deep recommendation model by incorporating an explicit (explainable) feature hierarchy.

Input: The input of our model includes a user set U, an item set V, and an explicit feature hierarchy Υ.

Definition 1. *An **explicit feature hierarchy** Υ is a tree where each node F_l is an explicit feature or aspect (e.g.,* meat*) of the items (e.g., restaurants). Each edge is a tuple (F_{l_1}, F_{l_2}), which represents that the child F_{l_1} (e.g.,* beef*) is a sub-concept of the parent F_{l_2} (e.g.,* meat*). We denote the set of nodes in Υ as $\mathcal{F}$, where $\mathcal{F} = \{F_1, ..., F_L\}$ and L represents the total number of nodes in Υ.*

To build Υ, we leverage the Microsoft Concept Graph[1] (Wu et al. 2012; Wang et al. 2015), which is a widely used knowledge graph with over 5 million concepts and 85 million "IsA" relations (e.g., *cat* IsA *animal*). We first map n-grams in the reviews to concepts (or instances) in the concept graph. Only the frequently used concepts that are highly

[1] https://concept.research.microsoft.com/

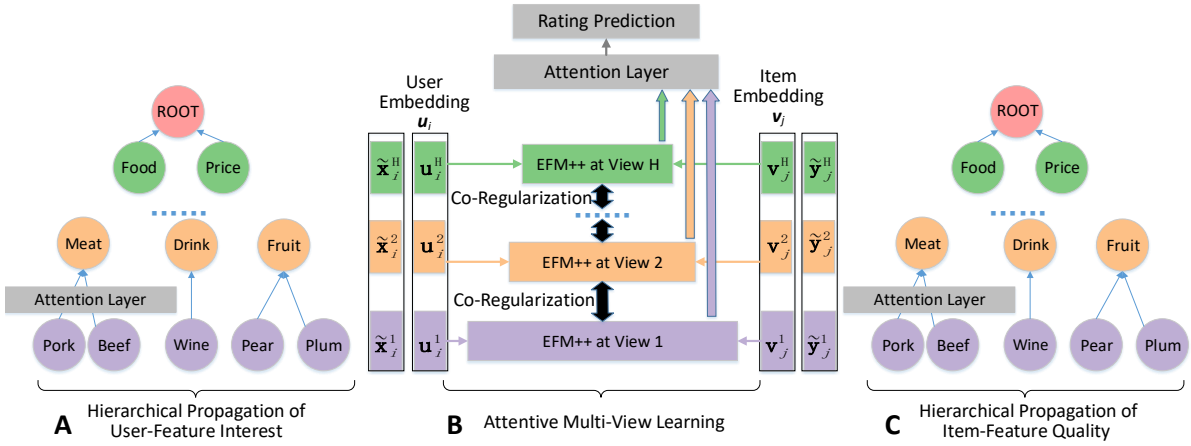

Figure 2: Architecture of our proposed DEAML model for rating prediction.

correlated with the ratings are kept and regarded as the explicit features. We then recursively search the explicit features in the concept graph and utilize the "IsA" relationships to build Υ. On average, the depth of Υ is 5.

Next, we define the hierarchical level of each feature.

Definition 2. *The **hierarchical level** h_l of F_l is recursively defined based on its children: $h_l = \max_{F_c \in children(F_L)} h_c + 1$. If F_l is a leaf, its hierarchical level is set to 1. Let H denote the largest hierarchical level of internal nodes in Υ.*

Besides Υ, two other types of input are the user set U and the item set V. Each user in U is represented as a tuple $(i, \boldsymbol{x}_i)$, where i is the user id and $\boldsymbol{x}_i \in \mathbb{R}^L$ is the observed user-feature interest vector. Here the l-th element in $\boldsymbol{x}_i$ measures how much user i cares about feature F_l. Similarly, each item in V is represented as $(j, \boldsymbol{y}_j)$ and the observed item-feature quality vector $\boldsymbol{y}_j \in \mathbb{R}^L$ measures how well j performs on these features. We adopt the method proposed by (Zhang et al. 2014a) to calculate $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$. The basic idea is to compute $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ based on the mentioned times and the sentiments of the features in the reviews.

Output: Given a user i and an item j, the output of our model includes the predicted rating $\tilde{r}_{ij}$ and a personalized feature-level explanation E, where $E = \{F_{l_1}, ..., F_{l_T}\}$ is a subset of $\mathcal{F}$. Similar to (Zhang et al. 2014a), the final explanations presented to users are in the template as follows.

> You might be interested in [features in E], on which this item performs well.

Approach Overview

Given an explicit feature hierarchy Υ, we build a Deep Explicit Attentive Multi-View Learning Model (DEAML) that is able to 1) accurately predict ratings and 2) produce useful explanations. Our method consists of two steps.

Step 1: Prediction with Attentive Multi-View Learning. In this step, we build the DEAML model by minimizing the rating prediction error. The architecture of the model is shown in Figure 2. For each user i, we first build a user-feature interest hierarchy (Figure 2A) by using Υ and $\boldsymbol{x}_i$. While $\boldsymbol{x}_i$ measures user interest to some extent, it is inaccurate due to data sparsity, noises in the data, and also overlapping semantics of the features. To improve accuracy, we propose a **hierarchical propagation** method that propagates user interest along the hierarchy based on user attention. Similarly, for each item j, we build an item-feature quality hierarchy (Figure 2C) by propagating $\boldsymbol{y}_j$ along edges of Υ. To effectively predict ratings by using the two hierarchies, we propose an **attentive multi-view learning** framework. As shown in Figure 2B, features at different hierarchical levels are regarded as different views. Predictions from multiple views are co-regularized to enforce the agreement among them. Considering a user's interest is not equally distributed over different hierarchical levels, we calculate the final prediction by attentively integrating results from multiple views. Hierarchical propagation and attentive multi-view learning are jointly optimized, which allows us to simultaneously infer user profiles and predict ratings in an end-to-end manner.

Step 2: Personalized explanation generation. In this step, we generate the explanations by using the DEAML model. To this end, we first define a utility function for candidate features, which jointly considers user-feature interest, item-feature quality and attention scores of different views. Then we formulate explanation generation as a constrained tree node selection problem and propose a dynamic programming algorithm to solve it efficiently.

Prediction with Attentive Multi-View Learning

In this section, we introduce how to predict the rating given $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$. We first describe the hierarchical propagation method used to build an accurate user-feature interest hierarchy. Then we propose the attentive multi-view learning framework that leverages complementary information in the feature hierarchy to improve rating prediction accuracy.

Hierarchical Propagation

A straightforward way to build the user-feature interest hierarchy is to map elements in $\boldsymbol{x}_i$ to Υ. Specifically, for each

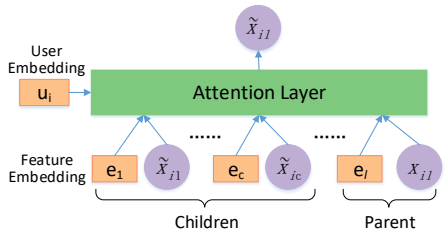

Figure 3: Hierarchical propagation of user interest.

feature F_l in Υ, we can assume that user i's interest in F_l is equal to x_{il}, which is the lth element in $\boldsymbol{x}_i$. Since x_{il} is calculated by counting the number of times F_l appears in the reviews written by user i, this method captures user interest to some extent. However, due to data sparsity, noises in the data, and overlapping semantics of features in Υ, the constructed user-feature interest hierarchy can be inaccurate. For instance, a user interested in *meat* may comment frequently on *beef* and *pork* but rarely mentions the word *meat*.

To solve this issue, we infer users' true interest $\tilde{\boldsymbol{x}}_i$ by propagating $\boldsymbol{x}_i$ along Υ. For each parent F_l, we calculate $\tilde{x}_{il}$ by considering personalized user attention (Figure 3):

$$\begin{aligned} \tilde{x}_{il} &= \alpha_{ll} x_{il} + \textstyle\sum_{F_c \in children(F_l)} \alpha_{lc} \tilde{x}_{ic}, \\ \alpha_{lc} &= \frac{exp(\alpha^*_{lc})}{\sum_{F_{c'} \in children(F_l) \cup \{F_l\}} exp(\alpha^*_{lc'})}, \\ \alpha^*_{lc} &= \boldsymbol{h}_1^\top ReLU(\boldsymbol{W}_l \boldsymbol{e}_l + \boldsymbol{W}_c \boldsymbol{e}_c + \boldsymbol{W}_u \boldsymbol{u}_i + \boldsymbol{b}_1) + b_2, \end{aligned} \tag{1}$$

where α_{lc} is the attention score of feature F_c. $\boldsymbol{e}_l, \boldsymbol{e}_c \in \mathbb{R}^{d_1}$ are pre-trained embeddings of F_l and F_c. We follow the method proposed in (Choi et al. 2017) to train feature embeddings that capture both semantic and hierarchical information. As there are H levels of features except F_L in Υ, the user embedding at level h is $\boldsymbol{u}_i^h \in \mathbb{R}^k$ and the global user embedding $\boldsymbol{u}_i \in \mathbb{R}^{kH}$ is the concatenation of $\boldsymbol{u}_i^h$ at all levels (shown in Figure 2). The hidden layer size of the attention network is d_2. $\boldsymbol{W}_l \in \mathbb{R}^{d_2 \times d_1}, \boldsymbol{W}_c \in \mathbb{R}^{d_2 \times d_1}, \boldsymbol{W}_u \in \mathbb{R}^{d_2 \times kH}, \boldsymbol{h}_1 \in \mathbb{R}^{d_2}, \boldsymbol{b}_1 \in \mathbb{R}^{d_2}, b_2 \in \mathbb{R}$ and $\boldsymbol{u}_i \in \mathbb{R}^{kH}$ are model parameters to be learned.

Propagation of $\boldsymbol{y}_j$ is similar to that of $\boldsymbol{x}_i$. After that we obtain a more accurate item-feature quality vector $\tilde{\boldsymbol{y}}_j$.

Attentive Multi-View Learning

After we obtain accurate estimates of user-feature interest and item-feature quality, we predicate how much user i likes item j based on $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{y}}_j$. To accurately predict the rating, we design an attentive multi-view learning framework. In this section, we first introduce how to predict the ratings based on the information in a single view. Then, we describe the co-regularization loss that enforces agreement among different views. Finally, we illustrate how to combine different views in a unified model and learn parameters from different views jointly.

Loss in each view. In our framework, each view consists of features at the same hierarchical level. We first consider rating prediction in a single view. Suppose there are L_h features at level h. By only considering users' interest on these features, we obtain a local user-feature interest vector $\tilde{\boldsymbol{x}}_i^h \in \mathbb{R}^{L_h}$, which is a slice of $\tilde{\boldsymbol{x}}_i$. Similarly, we can obtain $\tilde{\boldsymbol{y}}_j^h \in \mathbb{R}^{L_h}$, which is a slice of $\tilde{\boldsymbol{y}}_j$.

At level h, we employ an extended version of EFM (Zhang et al. 2014a) to predict the rating based on $\tilde{\boldsymbol{x}}_i^h$ and $\tilde{\boldsymbol{y}}_j^h$. Compared with collaborative filtering, EFM enriches the user and item representations by adding an additional set of latent factors learned from the explicit features. Specifically, the user embedding for view h consists of two parts: $\boldsymbol{u}_i^h = \boldsymbol{p}_i^h \oplus \boldsymbol{c}_i^h \in \mathbb{R}^k$. Here $\boldsymbol{p}_i^h \in \mathbb{R}^{k_1}$ are latent factors learned from explicit features (explicit factors), $\boldsymbol{c}_i^h \in \mathbb{R}^{k_2}$ are implicit factors, and $\oplus$ is the concatenation operator. Similarly, the item embedding at level h is $\boldsymbol{v}_j^h = \boldsymbol{q}_j^h \oplus \boldsymbol{d}_j^h$, where $\boldsymbol{q}_j^h \in \mathbb{R}^{k_1}$ denote the explicit factors and $\boldsymbol{d}_j^h \in \mathbb{R}^{k_2}$ are the implicit factors. The explicit factors $\boldsymbol{p}_i^h$ and $\boldsymbol{q}_j^h$ are used to fit $\tilde{\boldsymbol{x}}_i^h$ with $\boldsymbol{Z}^h \boldsymbol{p}_i^h$ and fit $\tilde{\boldsymbol{y}}_j^h$ with $\boldsymbol{Z}^h \boldsymbol{q}_j^h$, where $\boldsymbol{Z}^h \in \mathbb{R}^{L_h \times k_1}$ is the projection matrix of features at level h.

Traditional EFM directly predicts the rating of user i on item j as ${\boldsymbol{u}_i^h}^\top \boldsymbol{v}_j^h$. We further extend it by incorporating global average rating bias μ, user bias o_i and item bias o_j:

$$\tilde{r}_{ij}^h = {\boldsymbol{u}_i^h}^\top \boldsymbol{v}_j^h + o_i + o_j + \mu. \tag{2}$$

Let Ω denote the set of training instances and r_{ij} denote the ground-truth rating of user i on item j, the loss in view h is:

$$\begin{aligned} \mathcal{L}_h = \lambda_a \textstyle\sum_{i,j \in \Omega} (\tilde{r}_{ij}^h - r_{ij})^2 \ + \\ \lambda_x \textstyle\sum_i \|\tilde{\boldsymbol{x}}_i^h - \boldsymbol{Z}^h \boldsymbol{p}_i^h\|^2 + \lambda_y \textstyle\sum_j \|\tilde{\boldsymbol{y}}_j^h - \boldsymbol{Z}^h \boldsymbol{q}_j^h\|^2, \end{aligned} \tag{3}$$

where λ_a, λ_x and λ_y are weights of corresponding items.

Co-regularization loss. We then consider how the models in multiple views can complement each other. Based on the consensus principle which aims to maximize the agreement on multiple views (Xu, Tao, and Xu 2013), a common paradigm of multi-viewing learning is co-regularization. Actually, information distributed in different views describes the inherent characteristics of user-feature interest and item-feature quality from various aspects. We can regularize the learning of multiple views by enforcing agreement among their predictions. This way, complementary information can be leveraged to benefit the learning of each single view. By enforcing the agreement between two adjacent views, the co-regularization loss is:

$$\mathcal{L}_v = \sum_{i,j \in \Omega} \sum_{h=1}^{H-1} (\tilde{r}_{ij}^h - \tilde{r}_{ij}^{h+1})^2. \tag{4}$$

Joint learning. Predictions from multiple views should be combined for the final prediction. One method is to average the predictions of all views. However, this assumes that each view contributes equally to the rating. In reality, different views reveal user-feature interest and item-feature quality at different levels of abstractness. The levels of interest also vary among users and items. Some users may care more about general features (e.g. *food* and *service*) while others may be more attracted by a specific feature (e.g. *beef*). This is the same for items. The views are not equally useful and should be assigned different weights. The weights are supposed to be determined in a user-item-specific way to better consider users' interest and items' quality. Thus we use

the attention mechanism to combine multiple views. Let I_h denote the H-dimensional one-hot vector for view h. The attention network for view combination is:

$$w_h^* = \boldsymbol{h}_2^\top ReLU(\boldsymbol{W}_1\boldsymbol{u}_i + \boldsymbol{W}_2\boldsymbol{v}_j + \boldsymbol{W}_3\boldsymbol{I}_h + \boldsymbol{b}_3) + b_4,$$
$$w_h = \frac{exp(w_h^*)}{\sum_{h'=1}^{H} exp(w_{h'}^*)}. \quad (5)$$

where $\boldsymbol{u}_i, \boldsymbol{v}_j \in \mathbb{R}^{kH}$ are the global user embedding and item embedding used in hierarchical propagation. $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times kH}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times kH}, \boldsymbol{W}_3 \in \mathbb{R}^{d_3 \times H}, \boldsymbol{h}_2 \in \mathbb{R}^{d_3}, \boldsymbol{b}_3 \in \mathbb{R}^{d_3}, b_4 \in \mathbb{R}$ are model parameters and d_3 is the hidden layer size of this attention network. The final prediction is a weighted sum of predictions in each view, where w_h is the weight of view h:

$$\tilde{r}_{ij} = \sum_{h=1}^{H} w_h \tilde{r}_{ij}^h. \quad (6)$$

Let $\|\boldsymbol{\Theta}\|^2$ denote the L2 norm of all parameters in our model. We optimize the following objective function:

$$\mathcal{L} = \sum_{i,j \in \Omega} (\tilde{r}_{ij} - r_{ij})^2 + \sum_{h=1}^{H} \mathcal{L}_h + \lambda_v \mathcal{L}_v + \lambda_r \|\boldsymbol{\Theta}\|^2, \quad (7)$$

where λ_v, λ_r are weights of corresponding items. We use the Adam optimizer (Kingma and Ba 2014) because it can automatically adjust the learning rate during the training phase.

Personalized Explanation Generation

The goal of personalized explanation generation is to select T features from Υ that are most useful in helping user i decide whether s/he will try item j. Selecting features distributed in multiple hierarchical levels is challenging because 1) users' interest is not evenly distributed over different hierarchical levels and 2) we need to avoid selecting features with overlapping semantics. In this section, we first introduce our utility function, which estimates the usefulness of each feature. Then, we formulate the feature selection as a constrained tree node selection problem and propose a dynamic programming method to solve it efficiently.

Utility function. Given user i and item j, three factors are considered when judging whether a feature F_l is useful:

- Whether user i is interested in F_l.
- How well item j performs on F_l.
- The weight of the view that F_l belongs to.

In view h, the explicit factors $\boldsymbol{p}_i^h$ and $\boldsymbol{q}_j^h$ are used to fit the user-interest vector $\tilde{\boldsymbol{x}}_i^h$ with $\boldsymbol{Z}^h\boldsymbol{p}_i^h$ and fit $\tilde{\boldsymbol{y}}_j^h$ with $\boldsymbol{Z}^h\boldsymbol{q}_j^h$. Let F_l be the $\hat{l}$-th feature at level h. We define the utility score of F_l in the recommendation instance (i, j) as:

$$\Psi(F_l) = \left(\boldsymbol{Z}^h\boldsymbol{p}_i^h\right)_{\hat{l}} \left(\boldsymbol{Z}^h\boldsymbol{q}_j^h\right)_{\hat{l}} w_h, \quad (8)$$

where w_h is the user-item-specific attention weight for view h, which is calculated by using Equation (5). The utility score for the root feature is set to 0 since it does not have a concrete meaning. This utility function jointly incorporates three factors mentioned above.

Constrained tree node selection. Next, we select T features to be included in the explanation. The goal is to maximize the total utility score of the selected features. We further require that features (e.g. *beef*) cannot be selected simultaneously with their ancestors in Υ (e.g. *meat*) since they are semantically overlapping. Mathematically, we formulate personalized explanation generation as a constrained tree node selection problem:

$$\begin{aligned} &\underset{\phi_1,\ldots,\phi_L}{\operatorname{argmax}} \quad \sum_{l=1}^{L} \left(\phi_l \Psi(F_l)\right), \\ &s.t. \quad \sum_{l=1}^{L} \phi_l = T, \phi_l \in \{0,1\}, \\ &\phi_{l_1}\phi_{l_2} Anc(l_1, l_2) = 0, \forall\, h_{l_1} < h_{l_2}, \end{aligned} \quad (9)$$

where $\phi_l = 1$ ($\phi_l = 0$) means feature F_l is (not) selected and $Anc(l_1, l_2)$ returns 1 if F_{l_2} is an ancestor of F_{l_1}.

We solve the above problem by using dynamic programming. Let $G(l, t)$ denote the maximum utility we can obtain by selecting t nodes in the subtree rooted at F_l. Let us use ID_{ls} to represent the id of the s-th child feature of F_l, employ $J(l, s, t)$ to denote the maximum utility we can obtain by selecting t nodes from the first s children of F_l, and use S to denote the total number of children of F_l. We have the following transition equations:

$$\begin{aligned} G(l,t) &= \begin{cases} \max\left(\Psi(F_l), J(l, S, t)\right) & t = 1 \\ J(l, S, t) & t > 1 \end{cases} \\ J(l,s,t) &= \max_{0 \leqslant t' \leqslant t} \left(J(l, s-1, t') + G(ID_{ls}, t - t')\right). \end{aligned} \quad (10)$$

Here, we need to carefully consider whether $t = 1$ because if we select more than one node (i.e., $t > 1$) from the subtree rooted at F_l, F_l itself can not be selected. Otherwise, the constraint will be violated. Supposing that the root of Υ is F_L, we can obtain the final solution with the maximum utility G(L,T) by using Equation (10) iteratively.

Time complexity. If the maximum number of children of a feature is M, the time complexity of our dynamic programming algorithm is O(LMT^2). It is more efficient than the brute force algorithm whose time complexity is O(L^T).

Experiments

We conduct two experiments and a user study to evaluate the effectiveness of our method. First, we demonstrate that our method performs better than the state-of-the-art methods in terms of accuracy. Then, we conduct parameter sensitivity analysis to validate the robustness of our model. Finally, we evaluate the explainability of DEAML with 20 human subjects in a user study.

Table 1: Statistics of three public datasets.

Dataset	#Users	#Items	#Reviews
Toys and Games	19,412	11,924	167,597
Digital Music	5,541	3,568	64,706
Yelp	8,744	14,082	212,922

Experimental Settings

Datasets. We use three datasets from different domains for evaluation. Table 1 summarizes the statistics of the datasets.

- **Toys and Games** is the part of the Amazon dataset[2] that

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

Table 2: RMSE comparison with baselines on three datasets. Best results are highlighted in bold.

	G1			G2	G3				Ours	
	NMF	PMF	SVD++	CKE	HFT	EFM	DeepCoNN	NARRE	DEAML-V	DEAML
Toys and Games	1.1489	1.1832	0.9071	0.9923	0.9958	0.9534	0.9199	0.9084	0.9062	**0.9040**
Digital Music	1.1520	1.2619	0.9211	0.9849	1.0910	0.9696	0.9212	0.9209	0.9190	**0.9118**
Yelp	1.2678	1.2413	1.1561	1.2279	1.2738	1.2019	1.1503	1.1348	1.1343	**1.1333**

focuses on Toys and Games. We choose the 5-core version where all users and items have at least 5 reviews.

- **Digital Music** is also from the Amazon 5-core dataset. It focuses on the domain of digital music.
- **Yelp** consists of restaurant reviews from Yelp Challenge 2018[3]. Since the raw data is very large and sparse, we follow (Zhang et al. 2014a) to preprocess the data. Specifically, we select restaurants located in the Phoenix city and ensure that all users and items have at least 10 ratings.

Baselines. We select eight competitive methods for comparison. These methods can be divided into three groups according to the type of data they use.

- The first group (**G1**) only relies on the observed rating matrix for rating prediction. This group consists of three methods: **NMF** (Lee and Seung 2001), **PMF** (Mnih and Salakhutdinov 2008) and **SVD++** (Koren 2008).
- The second group (**G2**) contains a knowledge-based method, **CKE** (Zhang et al. 2016). Here, we regard the feature hierarchy Υ as the structural knowledge in CKE.
- The third group (**G3**) consists of four methods that leverage the textual reviews for rating prediction: **HFT** (McAuley and Leskovec 2013), **EFM** (Zhang et al. 2014a), **DeepCoNN** (Zheng, Noroozi, and Yu 2017) and **NARRE** (Chen et al. 2018). Among them, DeepCoNN and NARRE are deep learning-based. EFM is the state-of-the-art method for mining feature-level explanations from a single layer of features (no hierarchical structure).

Evaluation metric. We adopt the widely-used Root Mean Square Error (RMSE) to evaluate the accuracy of all algorithms on rating prediction. A lower RMSE indicates a better performance. This is calculated as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i,j}(\tilde{r}_{ij} - r_{ij})^2}, \tag{11}$$

where N is the total number of testing instances.

Details. We randomly split the dataset into training (70%), validation (15%) and test (15%) sets. We tune the hyper-parameters of each algorithm on the validation set and evaluate their performance on the test set. The number of latent factors k for algorithms is searched in [8,16,32,64,128]. After parameter tuning, we set $k = 8$ for NMF, PMF and HFT, and $k = 16$ for SVD++. We set $k = 32$ for EFM, CKE, DeepCoNN, NARRE and DEAML. We reuse most hyper-parameters of DeepCoNN and NARRE reported by their authors. For simplicity, the number of explicit and implicit factors are set equally in EFM and DEAML. When constructing Υ in DEAML, we select the top-2000 frequently mentioned concepts in reviews, sort them by their correlations with the ratings and keep only top-500 of them. We set d_1, d_2, d_3, λ_v, and λ_a to 20, 10, 10, 10.0, and 3.0, respectively.

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

Study on Model Accuracy

Overall performance. The accuracy of our approach and the baselines on three datasets are shown in Table 2. After analyzing the results, we made the following conclusions.

First, methods that incorporate review information (G3) generally perform better than others. Compared with those only relying on a rating matrix (G1) and the method that incorporates the feature hierarchy (G2), methods in G3 respectively achieve 8.9% and 3.0% improvement in accuracy on average. We ascribe this to the fact that reviews provide more information about users' interest and items' quality.

Second, deep recommendation models (DeepCoNN, NARRE and ours) perform significantly better than those with shallow models (10.7% improvement in accuracy on average). This is because deep models can model high-level features more effectively and better capture non-linear interactions. This again demonstrates the necessity of developing a deep version of the explainable models.

Third, our DEAML model achieves the best performance on all three datasets. Our approach performs significantly better than EFM, achieving 5.7%, 5.2% and 6.0% improvements on three datasets. This demonstrates the effectiveness of our attentive multi-view learning framework. While EFM predicts ratings from a single view, our design of attentive multi-view learning allows different views to complement and benefit each other. Compared with CKE that simply learns item embeddings from knowledge bases, our approach fully leverages the feature hierarchy and improves accuracy by 8.0%. The infusion of explicit feature hierarchy provides our approach with additional knowledge that DeepCoNN and NARRE cannot leverage. Given a set of explicit features, DEAML can differentiate low-level features from high-level ones and learn the associations between the features, while DeepCoNN and NARRE treat all features as if they are at the same level. On average, our method is 1.0% more accurate than DeepCoNN and NARRE.

Effectiveness of the attention mechanism. In DEAML, we employ the attention mechanism to combine predictions from different views. To validate the effectiveness of the attention mechanism, we design DEAML-V, which is a variant that assigns same weights to all views. From Table 2 we can observe that DEAML constantly performs better than DEAML-V. It verifies our consideration that different views contain features at different levels of abstractness and should be assigned weights according to users' interest and items' quality. The attention mechanism enables DEAML to combine multiple views more adaptively and perform better.

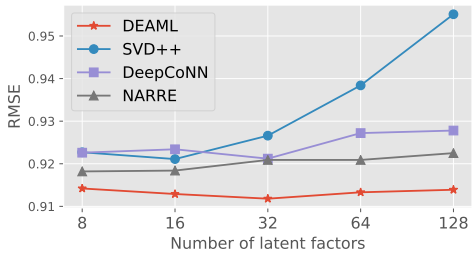

Figure 4: RMSE on Digital Music with different numbers of latent factors (compared with three competitive baselines).

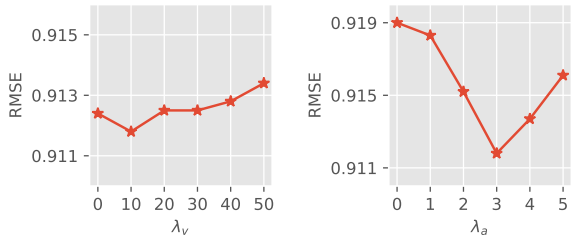

Figure 5: RMSE on Digital Music with different weights λ_v of co-regularization (left) and weights λ_a of errors in each view (right).

Parameter Sensitivity Analysis

In this section, we study the sensitivity of several important parameters in our approach. We only report results on Digital Music here due to the space limitation. Results on other datasets are similar.

Effect of number of latent factors. We first compare our approach with three competitive baselines (SVD++, DeepCoNN and NARRE) at different numbers of latent factors. The results are shown in Figure 4. We can see that DEAML consistently achieves the best performance with varying numbers of latent factors. This demonstrates the robustness of our approach. We further observe that increasing the number of latent factors does not necessarily improve accuracy, because too many latent factors may lead to over fitting.

Effectiveness of co-regularization. We then study how RMSE changes with λ_v, which is the weight of the co-regularization loss term. The left part of Figure 5 shows that our approach achieves the best performance when λ_v is 10.0. Smaller or larger λ_v will hurt performance. This demonstrates the importance of involving agreement (co-regularization) among different views.

Effectiveness of enforcing accurate prediction in each view. The right part of Figure 5 shows how RMSE changes with increasing λ_a, which is the weight of rating prediction errors in each single view. The lowest RMSE is achieved when $\lambda_a = 3.0$. When λ_a is 0, the performance degrades since we fail to train a good predictor in each single view.

Study on Explainability

In this section, we evaluate the usefulness of our explanations in helping real-world users make better decisions.With the help of a data annotation company, we recruit 20 participants who have written at least 15 Yelp reviews. The participants are independent and do not know each other. All reviews on the restaurants they mentioned are crawled and merged with other reviews in the Yelp dataset. Two baselines are used for comparison. The first baseline is PAV, which is the famous "People Also Viewed" explanation. The second baseline is EFM, which is the state-of-the-art method for generating feature-level explanations. We apply our method and baselines on the merged dataset and randomly sample 100 restaurants for testing. We ensure that none of the participants go to these restaurants. For each restaurant, we generate three explanations for each participant using PAV, EFM, and DEAML. The template of EFM is same as DEAML and PAV has it own format:"People also viewed this item". T is set to 3 for both EFM and DEAML. The participants are required to give annotations from 1 to 5 on each explanation according to its usefulness in helping them decide whether they will go to the restaurants. We randomly shuffle the order of explanations so that they do not know which explanation comes from which method[4].

Table 3: Average score on explanation usefulness. <30 and ≥30 refer to two age groups.

	Male	Female	<30	≥30	Overall
PAV	1.35	1.51	1.65	1.11	1.41
EFM	3.18	3.13	3.03	3.32	3.16
DEAML	**3.69**	**3.52**	**3.58**	**3.68**	**3.63**

Table 3 shows the average score on the usefulness of explanations generated by the three methods. We observe that PAV performs the worst since it only suggests that one restaurant is popular and is not personalized. Our DEAML outperforms EFM in explanation usefulness (14.9% improvement). This suggests that DEAML can accurately identify a user's interest and a restaurant's quality on multi-level features while EFM suffers from the sparse and noisy data. Additionally, EFM often selects features that are semantically overlapped (e.g., *food* and *cheese*) since it ignores the hierarchical structure of features while DEAML considers such constraints to make the generated explanation more informative. We also study the annotations given by participants of different profiles such as gender and age (see Table 3). DEAML consistently performs the best for both male and female participants. It is the same for participants in the two age groups. This suggests that DEAML is robust to various user profiles and is personalized enough to generate useful explanations that help each user make better decisions.

Conclusions

In this paper, we propose a Deep Explicit Attentive Multi-View Learning Model (DEAML) for explainable rec-

[4]Explanations generated are at https://www.microsoft.com/en-us/research/uploads/prod/2018/10/explanations.csv

ommendation, which combines the advantages of deep learning-based methods and existing explainable methods. The proposed attentive multi-view learning frameworks enables the model to accurately predict ratings and infer multi-level user profiles. To support personalized explanation generation from multi-level features, we formulate the problem as constrained tree node selection and solve it efficiently by using dynamic programming. Experimental results show that our model performs better than state-of-the-art methods in both accuracy and explainability.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant No. 61772045).

References

Chen, J.; Zhang, H.; He, X.; Nie, L.; Liu, W.; and Chua, T.-S. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *SIGIR*, 335–344. ACM.

Chen, C.; Zhang, M.; Liu, Y.; and Ma, S. 2018. Neural attentional rating regression with review-level explanations. In *WWW*, 1583–1592. International World Wide Web Conferences Steering Committee.

Choi, E.; Bahadori, M. T.; Song, L.; Stewart, W. F.; and Sun, J. 2017. Gram: graph-based attention model for healthcare representation learning. In *SIGKDD*, 787–795. ACM.

Das, A. S.; Datar, M.; Garg, A.; and Rajaram, S. 2007. Google news personalization: scalable online collaborative filtering. In *WWW*, 271–280. ACM.

De Francisci Morales, G.; Gionis, A.; and Lucchese, C. 2012. From chatter to headlines: harnessing the real-time web for personalized news recommendation. In *WSDM*, 153–162. ACM.

Diao, Q.; Qiu, M.; Wu, C.-Y.; Smola, A. J.; Jiang, J.; and Wang, C. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation. In *SIGKDD*, 193–202.

He, X.; Chen, T.; Kan, M.-Y.; and Chen, X. 2015. Trirank: Review-aware explainable recommendation by modeling aspects. In *CIKM*, 1661–1670. ACM.

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural collaborative filtering. In *WWW*, 173–182. International World Wide Web Conferences Steering Committee.

Kanagal, B.; Ahmed, A.; Pandey, S.; Josifovski, V.; Yuan, J.; and Garcia-Pueyo, L. 2012. Supercharging recommender systems using taxonomies for learning user purchase behavior. *Proceedings of the VLDB Endowment* 5(10):956–967.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kompan, M., and Bieliková, M. 2010. Content-based news recommendation. In *International conference on electronic commerce and web technologies*, 61–72. Springer.

Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*, 426–434. ACM.

Lee, D. D., and Seung, H. S. 2001. Algorithms for non-negative matrix factorization. In *NIPS*, 556–562.

McAuley, J., and Leskovec, J. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*, 165–172. ACM.

Mnih, A., and Salakhutdinov, R. R. 2008. Probabilistic matrix factorization. In *NIPS*, 1257–1264.

Peake, G., and Wang, J. 2018. Explanation mining: Post hoc interpretability of latent factor models for recommendation systems. In *SIGKDD*, 2060–2069. ACM.

Ren, Z.; Liang, S.; Li, P.; Wang, S.; and de Rijke, M. 2017. Social collaborative viewpoint regression with explainable recommendations. In *WSDM*, 485–494. ACM.

Wang, Z.; Wang, H.; Wen, J.-R.; and Xiao, Y. 2015. An inference approach to basic level of categorization. In *CIKM*, 653–662. ACM.

Wang, X.; Yu, L.; Ren, K.; Tao, G.; Zhang, W.; Yu, Y.; and Wang, J. 2017. Dynamic attention deep model for article recommendation by learning human editors' demonstration. In *SIGKDD*, 2051–2059. ACM.

Wang, X.; He, X.; Feng, F.; Nie, L.; and Chua, T.-S. 2018a. Tem: Tree-enhanced embedding model for explainable recommendation. In *WWW*, 1543–1552. International World Wide Web Conferences Steering Committee.

Wang, X.; Chen, Y.; Yang, J.; Wu, L.; Wu, Z.; and Xie, X. 2018b. A reinforcement learning framework for explainable recommendation. In *ICDM*. IEEE.

Wang, H.; Wang, N.; and Yeung, D.-Y. 2015. Collaborative deep learning for recommender systems. In *SIGKDD*, 1235–1244. ACM.

Wu, Y., and Ester, M. 2015. Flame: A probabilistic model combining aspect based opinion mining and collaborative filtering. In *WSDM*, 199–208.

Wu, W.; Li, H.; Wang, H.; and Zhu, K. Q. 2012. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD*, 481–492. ACM.

Xu, C.; Tao, D.; and Xu, C. 2013. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*.

Zhang, Y.; Lai, G.; Zhang, M.; Zhang, Y.; Liu, Y.; and Ma, S. 2014a. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *SIGIR*, 83–92. ACM.

Zhang, Y.; Ahmed, A.; Josifovski, V.; and Smola, A. 2014b. Taxonomy discovery for personalized recommendation. In *WSDM*, 243–252. ACM.

Zhang, F.; Yuan, N. J.; Lian, D.; Xie, X.; and Ma, W.-Y. 2016. Collaborative knowledge base embedding for recommender systems. In *SIGKDD*, 353–362. ACM.

Zheng, L.; Noroozi, V.; and Yu, P. S. 2017. Joint deep modeling of users and items using reviews for recommendation. In *WSDM*, 425–434. ACM.