Factor-level Attentive ICF for Recommendation

ZHIYONG CHENG, Qilu University of Technology (Shandong Academy of Sciences)
SHENGHAN MEI, Shandong University, China
YANGYANG GUO, Shandong University, China
LEI ZHU, Shandong Normal University, China
LIQIANG NIE, Shandong University, China

Item-based collaborative filtering (ICF) enjoys the advantages of high recommendation accuracy and ease in online penalization and thus is favored by the industrial recommender systems. ICF recommends items to a target user based on their similarities to the previously interacted items of the user. Great progresses have been achieved for ICF in recent years by applying advanced machine learning techniques (e.g., deep neural networks) to learn the item similarity from data. The early methods simply treat all the historical items equally and recent ones distinguish the different importance of items for a prediction. Despite the progress, we argue that those ICF models neglect the diverse intents of users on adopting items (e.g., watching a movie because of the director, leading actors, or the visual effects). As a result, they fail to estimate the item similarity on a finer-grained level to predict the user's preference for an item, resulting in sub-optimal recommendation. In this work, we propose a general factor-level attention method for ICF models. The key of our method is to distinguish the importance of different factors when computing the item similarity for a prediction. To demonstrate the effectiveness of our method, we design a light attention neural network to integrate both item-level and factor-level attention for neural ICF models. It is model-agnostic and easy-to-implement. We apply it to two baseline ICF models and evaluate its effectiveness on six public datasets. Extensive experiments show the factor-level attention enhanced models consistently outperform their counterparts, demonstrating the potential of differentiate user intents on the factor-level for ICF recommendation models.

CCS Concepts: • **Information systems → Personalization**; **Recommender systems**; **Collaborative filtering**.

Additional Key Words and Phrases: Attention, Collaborative Filtering, Diverse preference, Item-based Recommendation

ACM Reference Format:
Zhiyong Cheng, Shenghan Mei, Yangyang Guo, Lei Zhu, and Liqiang Nie. 2018. Factor-level Attentive ICF for Recommendation. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 26 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In the information age, we face overwhelming information at almost all aspects of our work and life. How to quickly find the desired information has thus become crucial in our daily lives. Recommendation as an effective information filtering and seeking technique [28, 38] has been widely deployed in current online service platforms, including information/media provider, E-commerce and social platforms. Among the many recommendation methods, collaborative filtering (CF) [29, 51, 74], which is one of the most dominant recommendation techniques, has attracted a

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06...$15.00
https://doi.org/10.1145/1122445.1122456

lot of attention from researchers and practitioners since its birth. In general, CF methods can be categorized into two paradigms: user-based CF (UCF) and item-based CF (ICF) [55]. The key of UCF is that users sharing close preferences often like the same items. That is, previously consumed items by one user will be recommended to another similar user with a large probability. In contrast, ICF methods represent a user with all his/her historically consumed items [51]. Specifically, the similarities between the target item and the previously interacted items are estimated firstly, which are then treated as the pivot for recommending similar items to the target user.

Comparing to UCF, ICF has several advantages in practice. Firstly, representing a user based on previously interacted items provides more accurate user modeling and thus has more potential to improve the performance. The user preference on items are relatively stable unless the background (or context) has changed dramatically, especially for the long-term preference. The aspects (or characteristics) that a user cares in the past is likely to be also important for the users in a long time. The ICF models represent a user by previous interaction items, which is actually profiling the user with the characteristics of those interacted items. Several empirical studies in literature provide some evidences on the superiority of ICF over UCF on accuracy in top-N recommendation [27, 33]. Secondly, ICF enjoys better interpretation, because it can explain a recommendation with similar items that the user interacted before. It is more acceptable for the users than the "similar users" based explanation, as those similar users might be strangers for the target user. In addition, ICF is flexible to incorporate new user-item interactions into the model, which makes it more suitable for online personalization [27, 72]. For new interactions, UCF methods need to re-train the model to update the user representations, which is very time-consuming and impractical in industrial applications. On the contrary, ICF can simply retrieve items similar to the newly interacted ones (i.e., leveraging item similarities) and recommend them to the current user. It does not need the model re-training processing and thus is more time efficient [14, 21, 54].

Early ICF approaches estimate the item similarities using statistical measures, such as Pearson coefficient [35] or cosine similarity [51]. The main drawback of those heuristic approaches is that they often require heavy manual tuning on the similarity measure for good performance on a target dataset. As a result, such methods are hard to be directly applied to a new dataset. To tackle this limitation, data-driven methods [33, 47] have been developed to learn item similarity from data. These methods first calculate the final result by setting an objective function, and then calculate the parameters by passing data into the loss function. Theoretically, the richer the data, the more accurate the model can be. The data-driven methods save the time of parameter adjustment. They can not only improve the efficiency but also enjoy higher accuracy. This is because the calculation of the parameters is based on the real data and does not rely on the experience of the participators. Recently, He et al. [27] pointed out that existing data-driven ICF methods assume all historical items of a user profile contribute equally in estimating the target item for the current user, which will result in sub-optimal performance. They therefore developed a neural attentive item similarity model (NAIS) to distinguish the different importance of previously interacted items for the user preference to the target item.

Though NAIS has achieved superior performance over existing ICF methods, we argue that its performance is still limited because it neglects user diverse intents on adopting items. More specifically, a user often pays attention to certain factors when selecting an item to consume. Accordingly, those factors will dominate the attitude of user preference towards this item. In addition, for each user, the dominant factors are usually different from item to item. For example, a user may favor a movie because of its plot, and likes another movie because he is fan of its director. With this consideration, we deem that treating all the factors[1] equally is not optimal in

[1]In this paper, we regard different dimensions of the item embedding as different factors that reflect user intents.

recommendation. However, it is not straightforward to explicitly model the impact of different factors in ICF for recommendation, because ICF models rely on estimating the similarity between the target item and historical items for prediction. In addition, the item-level attention has been demonstrated to be important for ICF [27]. To further enhance the recommendation accuracy, it is necessary to consider both item-level and factor-level attentions simultaneously in the model. How to combine them without complicating the model and increasing the computational burden much is another problem to be solved.

In this work, we make an effort to tackle aforementioned problems and present a general factor-level attention method for ICF models to consider user diverse intents in recommendation. Our proposed method models user diverse intents by distinguishing the impact of different factors of a historical item to the target item for prediction. More concretely, our method computes a weight vector for each historical item to estimate the similarity between this historical item and the target item. This weight vector is used to differentiate the contributions of different factors for the prediction by assigning different weights to each factor of the embedding vector. Based on this idea, we further design an attention neural network to combine the item- and factor-level attention for neural ICF models. It is light and easy-to-implement into different ICF models. To evaluate its effectiveness, we apply it to two models NAIS [27] & DeepICF [72] and conduct experiments on six Amazon datasets. Extensive experimental results show that the factor-level attention enhanced models can indeed improve the performance consistently over the couterpart model (i.e., NAIS and DeepICF) and achieve the state-of-the-art performance.

In summary, the main contributions of this work are threefolds:

- We highlight the importance of considering user diverse intents in ICF model and propose to model the intents on the factor level. In particular, we present a general factor-level attention method to measure the importance of different factors of a historical item to the target item for ICF models.
- We design a light and model-agnostic attention neural network which can effectively combine the item- and factor-level attention for neural ICF models. It is easy to implement in existing ICF models and we apply it to the NAIS and DeepICF to enhance their performance.
- We conduct extensive experiments on six publicly available datasets and demonstrate the effectiveness of our proposed method. Experimental results show that the enhanced NAIS and DeepICF achieve superior performance over their counterparts, demonstrating the effectiveness of our proposed method. We released our codes and the parameter settings for the experiments to facilitate others to repeat this work. [2]

The rest of this paper is organized as follows. We first introduce some background and the recent advancement of ICF models in Section 2. In Section 3, we elaborate our factor-level attention method and then describe its application to NAIS and DeepICF in Section 4. In the next, we report the experimental results in Section 5 and review related work in Section 6. Finally, we conclude the paper in Section 7.

2 BACKGROUND

This section first introduces the general framework of item-based collaborative filtering (ICF) and then recapitulates the recent advancement of ICF models. Table 1 lists the main notations used in this paper.

[2]https://github.com/masonmsh/FLA

Table 1. The main notations used in this paper.

Notation	Definition
$\mathbf{W}$	the trainable weight
$\mathbf{H}^\top$	the matrix that projects the hidden layer into an output layer
$\mathbf{h}^\top$	the vector that projects the hidden layer into the output layer
$\mathbf{b}$	the bias vector that projects input layer into hidden layer
$\mathbf{p_j}$	the embedding of the historically interacted item j
$\mathbf{q_i}$	the embedding of the target item i
$\mathbf{a}_{ij}$	attention vector for items i and j
b_{ij}	item-level attention for items i and j
a_{ijm}	the m-th dimension of the factor-level attention vector for items i and j
s_{ij}	similarity score between item i and j
r_{ui}	user u's rating for item i
$\mathcal{R}_u^+$	user u's interacted item set
$\mathcal{R}^+$	positive instances set
$\mathcal{R}^-$	negative instances set

2.1 Item-based CF

Item-based CF (ICF) predicts the preference of a user u to a target item i based on the similarity of i to all the items interacted by u in the past [54]. Formally, the prediction of ICF can be expressed as:

$$\hat{r}_{ui} = \sum_{j \in \mathcal{R}_u^+} s_{ij} r_{uj}, \tag{1}$$

where $\hat{r}_{ui}$ is the predicted ratings, $\mathcal{R}_u^+$ represents the set of items that user u has interacted, s_{ij} measures the similarity between item i and j, and r_{ij} denotes the preference level of user u for item j. Notice that r_{uj} can be real rating score (for explicit feedback) or binary value 0 or 1 (for implicit feedback). The matrix form of Eq. 1 is:

$$\hat{\mathbf{R}} = \mathbf{RS}, \tag{2}$$

where $\mathbf{R} \in \mathbb{R}^{U \times I}$ is the original interaction matrix, U and I represent the number of users and items, respectively. Each element $r_{ui} \in \mathbf{R}$ represents the rating score of a user u given to an item i. $\hat{\mathbf{R}} \in \mathbb{R}^{U \times I}$ is the reconstructed interaction matrix based on the ICF model. $\mathrm{S} \in \mathbb{R}^{I \times I}$ represents the item-item similarity matrix.

We can see that ICF is easy to take the top similar or recently interacted items into the model for prediction [2, 15]. This nice property makes it suitable for online learning and real-time personalization. The key of ICF lies in how to accurately and efficiently compute the similarity score s_{ij}. Early ICF models usually adopt heuristic-based approaches, such as using similarity measures like cosine similarity and Pearson coefficient [51] or applying random walks on the user-item interaction bipartite graph [40]. Such approaches are designed in an intuitive way and lack the tailored optimization for recommendation, resulting in suboptimal performance. In contrast, the learning-based methods optimize the ICF models with specially designed recommendation-oriented object function to learn item similarities. In the next subsections, we sequentially introduce several learning-based ICF models, including SLIM [47], FISM [33], NAIS [27], DeepICF [72].

2.2 Sparse Linear Methods (SLIM)

SLIM [47] is among the earliest attempts on learning-based ICF models which learn item-item similarity from the user-item interaction matrix. The basic idea is to minimize the reconstruction errors between the original user-item interaction matrix and the reconstructed matrix based on the item-based CF model. Specifically, the objective function is formulated as:

$$\begin{aligned} \underset{\mathbf{S}}{\text{minimize}}\ & \frac{1}{2}||\mathbf{R} - \mathbf{RS}||_F^2 + \frac{\beta}{2}||\mathbf{S}||_F + \gamma||\mathbf{S}||_1; \\ & \text{subject to } \mathbf{S} \geq 0,\ diag(\mathbf{S}) = 0, \end{aligned} \tag{3}$$

where $||\cdot||_F$ is the matrix Forbenius norm and the ℓ_2-norm is commonly used to prevent overfitting. The ℓ_1-norm regularization is to introduce sparsity to the model, i.e., enforcing only a few items that are similar to an item in the solution. β and γ are constants to balance the ℓ_F-norm and ℓ_1-norm regularization, respectively. The non-negative constraint $\mathbf{S} \geq 0$ is to ensure the similarity between each pair of items is positive; and the constraint $diag(\mathbf{S}) = 0$ is to avoid trivial solutions (i.e., $\mathbf{S}$ is an identical matrix to minimize $||\mathbf{R} - \mathbf{RS}||_F^2$).

With the designed object function optimized for recommendation, SLIM can achieve higher recommendation accuracy. However, with I^2 elements in $\mathbf{S}$, it is space- and time-consuming to learn the similarity matrix $\mathbf{S}$, which makes SLIM unscalable and limits its application in real systems, considering the tens of millions of items in modern E-commerce platforms. Another limitation is that SLIM can only learn the similarity between items which have been co-interacted by users and thus fails to capture the transitive relations between items. To address the second limitations, HOSLIM [10] was designed to model the high-order relations. In particular, it first mines itemsets which are frequently co-interacted by users, and then learns both item-item similarity and itemset-item similarity jointly. As it is a direct extension of SLIM and learns the similarity in the same manner, we omit the introduction of HOSLIM here. In the next, we would like to introduce FISM, which adopts a different learning strategy.

2.3 Factored Item Similarity Model (FISM)

SLIM directly learns the whole similarity matrix $\mathbf{S}$, which causes unaffordable resource consumption in terms of both space and item. To address the limitation, FISM [33] first represents each item as a low-dimensional vector and then models the similarity between each pair of items by the inner product of their embedding vectors. Specifically, let $\mathbf{p}_i$ and $\mathbf{q}_j$ be respectively the embedding vectors of item i and j, the similarity between i and j can be computed by $\mathbf{p}_i^T \mathbf{q}_j$. In FISM, the prediction of a user u's preference to an item i is modeled as:

$$\hat{r}_{ui} = \frac{1}{(|\mathcal{R}_u^+| - 1)^\alpha} \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} \mathbf{p}_i^T \mathbf{q}_j, \tag{4}$$

where $\alpha \in [0, 1]$ is a predefined hyper-parameter to control the normalization effect. $\mathcal{R}_u^+ \setminus \{i\}$ denotes all the items interacted by user u, excluding the current item i. It equals the constraint of $diag(\mathbf{S}) = 0$ to avoid the self-similarity of the target item in the modeling. There are two embeeding vectors for each item to distinguish the role of a prediction target or a historical interaction in FISM, which increases the expressiveness of the model. With the predicted ratings based on Eq. 4, the parameters of FISM can be learned by optimizing the reconstruction loss as Eq. 3 without the item similarity constraints. As all the items are represented as embedding vectors, even two items have not co-interacted by users, FISM can also estimate their similarity. To this end, FISM addresses the aforementioned limitations of SLIM and achieves better performance.

From Eq. 4, we can see that the preference of a user u towards a target item i depends on the aggregating effects of the similarity between i and all the items interacted by user u. In particular, when $\alpha = 0$, the predicted rating is the aggregated similarity between i and all the items that u interacted ($\mathcal{R}_u^+ \backslash \{i\}$); and when $\alpha = 1$, the predicated rating is the average similarity between i and the items in $\mathcal{R}_u^+ \backslash \{i\}$. The underlying assumption of FISM is that each item contributes equally for the preference prediction to the target item. However, this is often not true in practice, because some items are more relevant to the currently targeted item. For example, to infer a user's preference on a "keyboard", the previously purchased "computer" plays a more important role in the prediction than a pair of "shoes" purchased by the user in the past. As different items are relevant to the current items at different levels, it is beneficial to assign different weights to the historical interacted items for more accurate prediction.

2.4 Neural Attentive Item Similarity Model (NAIS)

To capture the different contributions of historical items to the preference prediction of a user to the target item, NAIS [27] introduces the attention mechanism [4] to assign different weights to different items. Specifically, the prediction of NAIS is formulated as:

$$\hat{r}_{ui} = \sum_{j \in \mathcal{R}_u^+ \backslash \{i\}} a_{ij} \mathbf{p}_i^T \mathbf{q}_j, \tag{5}$$

where a_{ij} denotes the attentive weight assigned to the similarity s_{ij}, indicating the contribution of item j to the preference prediction of item i. The attentive neural network is used to automatically learn a_{ij} by taking $\mathbf{p}_i$ and $\mathbf{q}_i$ as input. Two different methods have been presented in NAIS to combine $\mathbf{p}_i$ and $\mathbf{q}_i$, i.e., vector concatenation and element-wise product:

$$\begin{cases} f_{concat}(\mathbf{p}_i, \mathbf{q}_j) = \mathbf{h}^T ReLU(\mathbf{W} \begin{bmatrix} \mathbf{p}_i \\ \mathbf{q}_j \end{bmatrix} + \mathbf{b}) \\ f_{prod}(\mathbf{p}_i, \mathbf{q}_j) = \mathbf{h}^T ReLU(\mathbf{W}(\mathbf{p}_i \odot \mathbf{q}_j) + \mathbf{b}), \end{cases} \tag{6}$$

where $\mathbf{W} \in \mathbb{R}^{d' \times d}$ and $\mathbf{b} \in \mathbb{R}^{d'}$ represent the weight matrix and bias vector of the attention network, respectively. d' denotes the size of the hidden layer. $\mathbf{h}$ is the weight vector of the output layer of the attention network. *ReLU* [45] is used as the activation function. The a_{ij} is then normalized via a modified softmax function:

$$a_{ij} = \frac{\exp(f(\mathbf{p}_i, \mathbf{q}_j))}{[\sum_{j \in \mathcal{R}_u^+ \backslash \{i\}} \exp(f(\mathbf{p}_i, \mathbf{q}_j))]^{\beta}}, \tag{7}$$

where β is a hyperparameter to smooth the denominator of the softmax function. The rational lies in the fact that the number of users' interacted items can vary in a wide range. As a result, the standard softmax normalization will overly punish the weights of active users, who have much more interacted items than inactive users. With a smaller β, the denominator can be suppressed and thus reduce the punishment on the attention weights of active users [27]. Notice that with the normalization of the modified softmax, the normalization term ($\frac{1}{(|\mathcal{R}u^+|-1)^{\alpha}}$) is discarded in NAIS.

With the item-level attention modeling[3], NAIS can distinguish the different importance of interacted items on the preference of a user to the target item and thus achieve better performance. Both FISM and NAIS only focus on the modeling of the second-order item relations, however, they

[3]Becuase the attentive weight (or contribution) is assigned to each historical item in NAIS, we call it item-level attention modeling.

ignore the higher-order item relations in the data and thus may yield suboptimal performance [10, 72]. In the next, we introduce the recently proposed DeepICF which applies deep interaction layers to capture the higher-order item relations.

2.5 Deep Item-based Collaborative Filtering (DeepICF)

There are two steps in the above introduced ICF models to predict user preference: 1) item similarity estimation and 2) similarity aggregation. SLIM and FSIM focus on the first step, i.e., proposing different learning methods to estimate the item similarity, and NAIS contributes to the second step by introducing weights to similarities from different items in the aggregation. In [72], authors argue that previous models fail to capture high-order item relations and propose the DeepICF model, which adopts a different strategy for preference prediction. Specifically, DeepICF first adopts a pairwise interaction layer to model the interaction between each historical item and the target item, and then introduces an attention-based pooling layer to assign different weights to the outputs (of the pairwise interaction layer) from different historical layer. In the next, the output from the previous two layers are fed into deep interaction layers, which consists of a multi-layer perceptron, to model the high-order interaction between items. Finally, a linear regression model is applied to predict the preference. In the next, we introduce each component in details for a clear impression of the DeepICF model. The pairwise interaction layer and the attention-based pooling layer are expressed as:

$$\mathbf{e_{ui}} = \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} a_{ij}(\mathbf{p}_i \odot \mathbf{q}_j), \tag{8}$$

where $\odot$ indicates element-wise product. a_{ij} is the item-level attention, which denotes the contribution of item j to the user preference on item i. The attention is computed as:

$$a_{ij} = softmax'(\mathbf{h}^{\mathrm{T}} ReLU(\mathbf{W}(\mathbf{p}_i \odot \mathbf{q}_j) + \mathbf{b})), \tag{9}$$

where $\mathbf{W}$, $\mathbf{b}$, and $\mathbf{h}$ are defined as in Eq. 6 represent the weight matrix and bias vector of the attention network, respectively. $softmatx'$ is a modified softmax function as defined in NAIS (see Eq. 7). The deep interaction layers are stacked above the output of the interaction layer to model the higher-order item relations as follows:

$$\mathbf{e_L} = ReLU(\mathbf{W_L}(ReLU(\mathbf{W_{L-1}} \cdots ReLU(\mathbf{W_1 e_{ui}} + \mathbf{b_1})) + \mathbf{b_{L-1}}) + \mathbf{b_L}), \tag{10}$$

where $\mathbf{W_l}$, $\mathbf{b_l}$, and $\mathbf{e_l}$ denote the weight matrix, bias vector, and output vector of the lth hidden layer respectively. L is the total number of network layers. Finally, the prediction is achieved by a linear regression model:

$$\hat{r}_{ui} = \mathbf{V}^T \mathbf{e_L} + b_u + b_i, \tag{11}$$

where $\mathbf{V}$ is the weight vector for the prediction; b_u and b_i are the user and item bias as in the standard matrix factorization [38]. As we can see, different from previous models that predict preference based on the aggregation of the similarity between the target item and the historical items, DeepICF first models the complicated interactions between the target item and the historical item (based on the attention-based pairwise interaction pooling and deep interaction modeling), and then uses a simple regression model based on the interaction vectors for preference prediction. Note that DeepICF also adopts the item-level attention (in the attention-based pooling layers) as NAIS. Due to the high-order item relation modeling, DeepICF achieves better performance than NAIS.

2.6 Motivation of Our Work

Despite the success of NAIS and DeepICF by considering the different contributions of historical items to the target item, however, we argue that the item-level attention cannot well capture the fine-grained preferences of users on items. The overall preference of a user towards an item depends on the user's attention and satisfaction on different features (or factors) of the item [8]. For example, when a user cares more about the "taste" and "price" for dinner, she will choose a restaurant mainly based on the consideration of the price and taste of the food; in contrast, if the user pays more attention to the "service" and "ambience" of the restaurant when have dinner with friends, the service and ambience will become the dominant factors. As we can see, for different items, a user may focus on different factors. The item-level attention cannot distinguish the importance of different factors, and thus cannot capture the fine-grained preference of a user to the different factors of an item. In the next section, we present our method to model the factor-level attention, which can be easily applied to existing ICF models for fine-grained preference modeling.

3 FACTOR-LEVEL ITEM ATTENTION

The underlying intuition of NAIS is that the more relevant of a historical item to the target item, the more important role it plays in the preference prediction. Therefore, NAIS introduces an attention mechanism to estimate the contribution of each historical item to the target item. The item-level attention only computes an attentive weight based on the overall relevance between the two items while ignore the attention on different factors, and thus fail to capture the fine-grained user preference. It is well-known that an item is depicted by different factors [7] and the preference of a user to an item often depends on a few factors, such as the *directors* or *actors* of an movie. Therefore, a user u's preference to an item i depends on *u's attention on which factors of the item* and *whether those factors of the item fit the user's tastes*. Based on this consideration, when considering the contribution of a historical item to the target item in preference prediction in ICF, it is better to measure the importance of different factors of the historical item. In this section, we will introduce a factor-level attention method, which considers the contribution of historical items on the factor-level for ICF recommendation. We first introduce the general method of computing the factor-level attention between two items, and then introduce the method to consider both item-level and factor-level attention in ICF models.

3.1 Factor-level Attention

In the embedding-based ICF models (such as FISM, NAIS, DeepICF), items are mapped into a latent feature space and each item is represented by a vector in this space. Let $\mathbf{p_i} \in \mathbb{R}^d$ and $\mathbf{q_j} \in \mathbb{R}^d$ denote the feature vector of the target item i and a historical item j, respectively. d is the dimensionality of the latent space and each dimension can be regarded as a factor to describe the items. Our intuition is that different factors of a historical item j contribute differently to a target item i. Taking a toy example: if we only use two factors - "leading actors" and "director" to describe a movie, given a historical movie m_0 of a user u, it has the same *leading actors* but different *directors* from a movie m_1; and for another movie m_2, it has different *leading actors* but the same *director*. When predicting u's preference to m_1, the factor of "leading actors" should play a more important role than that of the "director", but for predicting u's preference to m_2, the factor of "director" will be more important. Therefore, for each historical item $j \in \mathcal{R}_u^+ \backslash \{i\}$, our goal is to compute an attentive vector $\mathbf{a_{ij}}$, in which each element $\{a_{ijk} | k = \{1, \cdots, d\}\}$ indicates the importance of k-th factor of the item j with respect to the target item i.

Notice that the value of a_{ijk} indicates the relatively importance of the k-th factor, it depends on the similarity of other factors between the two items i and j. For example, if all the factors of the two items are the same (e.g., the leading actors and directors are all the same for two movies), the factors are equally important. Inspired by the effectiveness of NAIS and DeepICF, we first use the element-wise product on the vectors of two items to obtain an interacted vector, i.e., $\mathbf{p_i} \odot \mathbf{q_j}$, where $\odot$ indicates element-wise product. We then follow the standard attention mechanism by applying a non-linear transformation to obtain the attentive weights:

$$\mathbf{\hat{a}_{ij}} = \mathbf{H}^{\top} ReLU(\mathbf{W}(\mathbf{p}_i \odot \mathbf{q}_j) + \mathbf{b}), \tag{12}$$

where $\mathbf{W} \in \mathbb{R}^{d' \times d}$ and $\mathbf{b} \in \mathbb{R}^{d'}$ denote the weight matrix and bias vector of the attention network, respectively. $\mathbf{H} \in \mathbb{R}^{d' \times d}$ denotes the weight matrix of the output layer of the attention network.[4] d' denotes the size of the hidden layer. Because our goal is to compute the importance of different factors of the items. The softmax function is then used to normalize the attentive weights:

$$a_{ijk} = \frac{\exp(\hat{a}_{ijk})}{\sum_{k'=1}^{d} \exp(\hat{a}_{ijk'})}. \tag{13}$$

In this design, the attentive weights of factors are computed based on the interaction between the vectors of two items. Theoretically, other functions can be also applied to encode the interaction, for example, addition, subtraction, etc. We use the element-wise product because it is a generalization of inner product to vector space. Notice that each element in $\mathbf{v_{ij}}$ (i.e., v_{ijk}) is a product of the corresponding factor of the two vectors (i.e., $p_{ik} \cdot q_{jk}$), and it can be regarded as a similarity of the corresponding factor between two items. This is similar to the inner product to compute the similarity between two vectors.

The proposed factor-level attention looks similar to that of NAIS, because both methods compute the attention of a historical item to a target item. The difference is that NAIS computes an attentive weight for the historical item, but we compute an attentive weight vector for all the factors of the historical item. Our method takes one-step further to consider the different contributions of factors in items than NAIS which considers the contributions of different items. As aforementioned, our model computes the relatively importance of each factor among all the factors of an item. Considering that historical items have different relevance levels to a target item, we should also consider the item-level attention simultaneously. Because for a target item, the factors with high attentive weight of an irrelevant item could contribute less than the factors with relatively low attentive weights of a relevant item. In the next section, we introduce our design to consider both item-level attention and factor-level attention in ICF models.

3.2 Item- and Factor-level Attention

Design 1. To integrate both the item-level and factor-level attention in an ICF model, an straightforward method is first to use two attention networks to compute the two types of attention separately, and then combine them together. Fig. 1 shows the network of this design. Specifically, one network is to model the different importance of previous items, and the other network is to capture the different contributions of factors inside items. For the item-level attention, the attentive weight b_{ij} of a historical item j to a target item i is computed according to the method described in NAIS, and we use the element-wise product method in our implementation[5]. Formally, the b_{ij} is computed as following:

$$v_{ij} = \mathbf{h}^{T} ReLU(\mathbf{W}(\mathbf{p}_i \odot \mathbf{q}_j) + \mathbf{b}), \tag{14}$$

[4]Note that in the NAIS, it is a weight vector $\mathbf{h} \in \mathbb{R}^{d'}$ for the output layer of the attention network, see Eq. 6.

[5]Note that the concatenation method in Eq. 6 can be also applied here. We use the element-wise product for the ease of computing both the item-level attention and factor-level attention

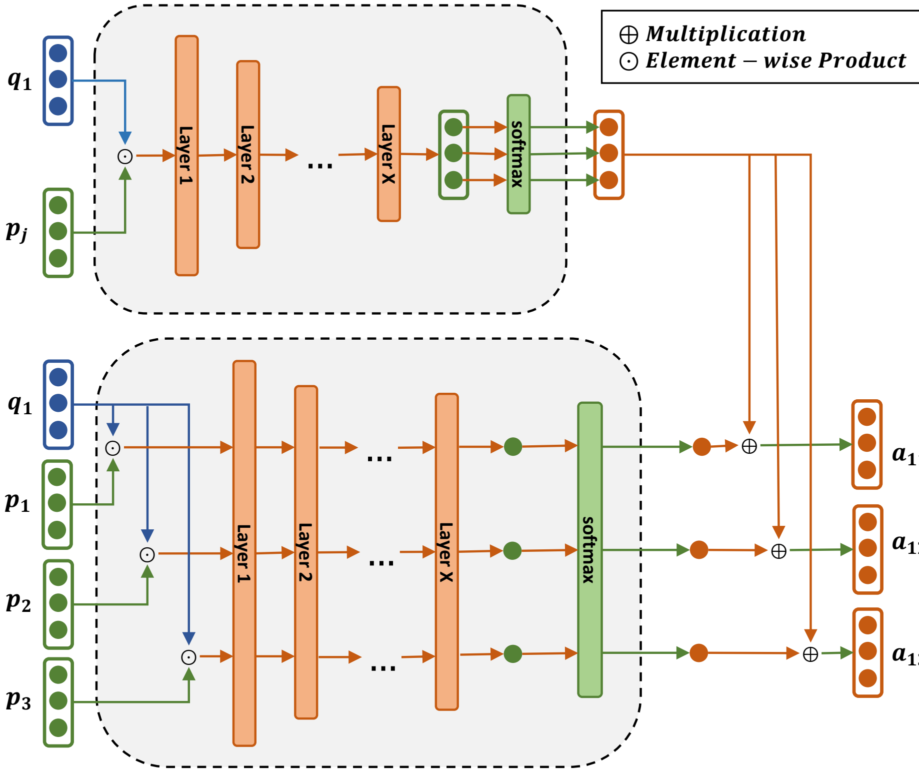

Fig. 1. The structure of the item- and factor-level attention network for the first design.

$$b_{ij} = \frac{\exp(v_{ij})}{[\sum_{j' \in \mathcal{R}_u^+ \setminus \{i\}} \exp(v_{ij'})]^{\beta}},. \tag{15}$$

Here the parameters $\mathbf{W}$, $\mathbf{b}$, $\mathbf{h}$ and the variant of softmax function (Eq. 15) are defined as they are in NAIS (i.e., Eq. 6 & 7). For the factor-level attention, the attentive weight vector $\mathbf{a}'_{\mathbf{ij}}$ is computed based on the Eq. 12 and 13. The final attentive weight of a factor of the item j is weighted by the item attentive weight.

$$\mathbf{a}_{\mathbf{ij}} = b_{ij} \cdot \mathbf{a}'_{\mathbf{ij}}. \tag{16}$$

It is easy to understand the intuition of the equation. If the historical item itself is irrelevant to the target item, the impact of its factors on predicting user preference to the target item should also be small. This method is easy to understand but the network structure is complicated.

Design 2. Since our goal is still to assign an attentive weight vector for each historical item, we attempt to simplify the network structure in the first design and propose a fusion method, whose structure is shown in Fig. 2. In this method, the attentive weights of a historical item j's factors for a target item is computed as:

$$\begin{cases} \hat{\mathbf{a}}_{\mathbf{ij}} = \mathbf{H}^{\top} ReLU(\mathbf{W}(\mathbf{p}_i \odot \mathbf{q}_j) + \mathbf{b}) \\ a_{ijk} = softmax'(\mathbf{f}(\hat{a}_{ijk}), \quad k = 1, 2, \cdots, d \end{cases} \tag{17}$$

Similar to the calculation of the factor-level attention, we first model the interactions between the historical items and the target items using a nonlinear transformation upon the element-wise

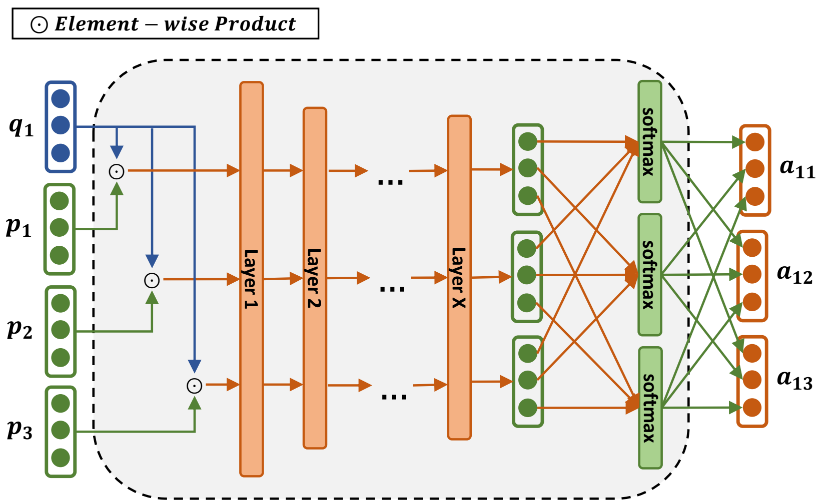

Fig. 2. The structure of the item- and factor-level attention network for the second design.

product of their embedding vectors. The computation of $\hat{\mathrm{a}}_{\mathrm{ij}}$ is the same as in Eq. 12 and the notations are defined in the same way. The difference comes from the normalization part. For the factor-level attention inside an item in the "Design 1", the attentive weight of a factor is normalized over the weights of *all the factors of this item*. In this design, the weight of a factor is normalized over the weights of *all the historical items on this particular factor*. In particular, the final attentive weight of the k-th factor inside an item j is obtained via a normalization based on the variant of the softmax function [27]:

$$softmax'(\hat{\mathrm{a}}_{\mathrm{ijk}}) = \frac{\exp(a_{ijk})}{[\sum_{j' \in \mathcal{R}_u^+ \setminus \{i\}} \exp(\hat{a}_{ij'k})]^{\beta}}. \tag{18}$$

Comparing to the Eq. 13, we can see that for a factor of a historical item, its importance is evaluated among the same factor of all the historical items in the normalization. In this way, the computation of the factor-level takes the item-level effects into consideration. Notice that it is possible that the attentive weights of all the factors of an item are small because this item is not relevant to the target item. Similar in NAIS, the hyper-parameter β is to smooth the value of the denominator in softmax. It can help regulate the weights of the item factors for users with different numbers of interacted items.

The mechanisms of the above two designs for considering both the item- and factor-level attentions are different. It is theoretically difficult to analyze which one works better in practice. The advantage of the second method is that the network structure is simple and computationally efficient. Besides, it has less parameters and thus is relatively more resistant to overfitting over the first method. We compare the recommendation performance of the two methods in experiments.

3.3 Prediction

With the attentive weight vector for each historical item $j \in \mathcal{R}_u$ of a user u, the preference to a target item i based on the factor-level attentive method is predicted by:

$$\hat{r}_{ui} = \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} \mathbf{p}_i^T (\mathbf{a_{ij}} \odot \mathbf{q}_j). \tag{19}$$

From this equation, we can see that our model considers the influence of different factors of all the historical items for the preference prediction.

4 FACTOR-LEVEL ATTENTION ENHANCED ICF MODELS

The proposed factor-level attention model can be easily applied to existing embedding-based ICF models. In this section, we show the applications of our factor-level attentive (FLA) method to two recently proposed ICF models: NAIS [27] and DeepICF [72]. For the ease of presentation, we name the two models with the use of our factor-level attention method as $\mathbf{FLA}_{NAIS}$ and $\mathbf{FLA}_{DICF}$, respectively.

$\mathbf{FLA}_{NAIS}$**.** NAIS [27] considers the different contributions of historical items. The application of the factor-level attention to NAIS is the integration of the item- and factor-level. Therefore, the methods described in section 3.2 is applied to compute the attentive weight vectors for historical items (Eq. 16 or Eq. 18), and then the preference to the target item is predicted by Eq. 19.

$\mathbf{FLA}_{DICF}$**.** DeepICF also considers the item-level attention. Similar to NAIS, the methods in section 3.2 is used to compute the attentive weight vectors for historical item model. The attentive weight (a_{ij}) in Eq 8 is replaced by the obtained weight vector ($\mathbf{a_{ij}}$), and Eq 8 becomes:

$$\mathbf{e_{ui}} = \sum_{j \in \mathcal{R}_u^+ \setminus \{i\}} \mathbf{p}_i \odot (\mathbf{a_{ij}} \odot \mathbf{q}_j) \tag{20}$$

We keep the other parts as the same as DeepICF, and thus the preference is still predicted by Eq. 11

4.1 Optimization

In this work, we target at the top-n recommendation, which is a more practical task than rating prediction in real commercial systems [50]. It aims to recommend a set of n to-ranked items which match the target user's preferences. Similar to other rank-oriented recommendation work [25, 27, 62], we adopt the pairwise-based learning method for optimization. As we would like to validate the effectiveness of the proposed factor-level attention, we strictly follow the implicit feedback setting in the work of NAIS [27] and DeepICF [72], w here each user-item interaction has a value of 1 and other non-observed user-item pairs have a 0 value. The recommendation model is also treated as a binary classification task, and the objective function is as follows:

$$L = -\frac{1}{|\mathcal{R}^+| + |\mathcal{R}^-|} \left(\sum_{(u,i) \in \mathcal{R}^+} \log \sigma(\hat{r}_{ui}) + \sum_{(u,i) \in \mathcal{R}^-} \log(1 - \sigma(\hat{r}_{ui})) \right) + \lambda \|\Theta\|^2, \tag{21}$$

where $\mathcal{R}^+$ denotes the positive instances set and $\mathcal{R}^-$ denotes the negative one where each user-item instance is sampled from the non-interacted pairs; σ is a sigmoid function, which can convert the predicted score $\hat{r}_{ui}$ of user u and item i into a probability representation, constraining the result to (0,1); λ is the parameter to control the effect of ℓ_2 regularization, which is used to prevent overfitting; and Θ represents all the trainable parameters including $\mathbf{p}_i$, $\mathbf{q}_j$, $\mathbf{H}$, $\mathbf{W}$ and $\mathbf{b}$. In addition, FLA_{DICF} has a multi-layer perception behind the attention network to simulate high-level interactions of users and Θ also contains their weight parameters.

Model training. We adopt Adagrad [19] for optimize the prediction model and update the model parameters. Because the objective function is non-convex, the loss function might be trapped in a local minimum, resulting in sub-optimal performance. Previous work has demonstrated that *pre-training* is particularly useful in practice for accelerating the training process and achieving

Table 2. Basic statistics of the experimental datasets.

Dataset	#Users	#Items	#Ratings	Sparsity
Patio	1,686	962	13,272	99.18%
Music	5,541	3,568	64,706	99.67%
Grocery	14,681	8,713	151,254	99.88%
Beauty	22,363	12,101	198,502	99.93%
Clothing	39,387	23,033	278,677	99.97%
Home	66,519	28,237	551,682	99.97%

better performance [26, 28]. We report the results with and without the pre-training in experiments (see section 5.5).

5 EXPERIMENTS

We conducted extensive experiments on six publicly accessible datasets to evaluate the effectiveness of the proposed method. In particular, we mainly answer the following research questions.

- **RQ1:** Which design is better to integrate the item-level and factor-level attention, *Design 1* or *Design 2*?
- **RQ2:** Are our proposed factor-level attention methods useful for providing more accurate recommendations? How do our factor-level attention enhanced methods perform with comparison to the state-of-the-art item-based CF methods?
- **RQ3:** How do the hyper-parameters, i.e., the embedding size d and β, affect the performance of the factor-level attention enhanced methods?
- **RQ4:** Is the pre-training strategy useful for our factor-level attention enhanced methods?

In what follows, we first present the experimental settings, and then answer the above questions sequentially based on experimental results.

5.1 Experimental Setup

Datasets. We adopt the widely used benchmark dataset - the Amazon review dataset [46] for recommendation evaluation in our experiments. This dataset contains user interactions on items as well as item metadata from Amazon. In our experiments, we only use the interaction information. For each observed user-item interaction, we treated it as a positive instance; otherwise, it is negative. Six product categories from this dataset are used in evaluation, as shown in Table 2. The 5-core version of the dataset is used, which means that users and items in the dataset have at least 5 interactions. The basic statistics of the six categories are also shown in Table 2. As we can see, the selected datasets are of different sizes and sparsity levels, which can evaluate the performance of the proposed method for item recommendation under different scenarios.

Evaluation Protocols. As our main focus in this work is to study whether the factor-level attention can enhance the performance of item-based CF model's performance, We strictly follow the same evaluation protocol as the one used in NAIS [27] and DeepICF [72] to study the performance of item recommendation. Specifically, the latest interaction of each user is held-out as the testing data, the second latest interaction is reserved as the validation data, and the remaining interactions are used for training. For each user, we randomly sampled 99 items (negative instances) which are not interacted by this user for the testing item (positive instance). In the testing stage, each studied recommendation model predicts the preference scores for the 100 items (1 positive and 99 negative instances). The performance is evaluated by the widely used metrics - *Hit Ratio* (HR) [16] and *Normalized Discounted Cumulative Gain* (NDCG) [16]. For each metric, the performance of recommendation methods is often judged by the top n results. Particularly, *HR@n* is a recall-based metric, measuring whether the test item is in the top-n positions of the recommendation list. *NDCG@n* emphasizes the quality of ranking, which assigns higher score to the top-ranked items by taking the position of correctly recommended into considerations. The reported results are the average values across all the testing users based on the top 10 results (i.e., $n = 10$).

Compared Baselines. As the main contribution of this work is to advocate the importance of considering the factor-level attention in recommendation, especially for the item-based CF recommendation methods. Therefore, we mainly compared our factor-level attention enhanced methods with the state-of-the-art ICF models in the empirical studies. Specifically, we compared our methods with the following baselines.

- **BPR** [50] is a popular pair-wise learning method, which employs a Bayesian Personalized Ranking loss to optimize the matrix factorization model. This is a basic baseline with competitive performance on the top-n recommendation task and has been widely used in empirical studies to evaluate the newly developed method.
- **MLP** [28] uses a multi-layer perceptron above user and item embeddings to replace the inner product for recommendation. Following the setting in [27], we also use a three-layer MLP and optimize the point-wise log loss in experiments.
- **SLIM** [47] is the earliest learning-based item-based CF model. It learns an item-item similarity matrix to reconstruct the user-item interaction function as demonstrated in Eq. 3.
- **FISM** [33] is a pioneering learning-based ICF model by directly learning item embeddings as formulated in Eq. 4. In experiments, we carefully tuned α from 0 to 1 with a step size of 0.1 and reported the best result for each experimental dataset.
- **NAIS** [27] is a state-of-the-art item-based CF model. It considers the different effects of historical items to the target item and applies the attention mechanism to model the item-level attention in the modeling.
- **DeepICF** [72] is a recently proposed deep ICF method which can capture the high-order interactions between items. By stacking perceptron layer above the interactions between items, it adopts a linear regression for the final prediction.

BRP is a traditional and competitive CF model based on matrix factorization for the top-n recommendation task. MLP is a state-of-the-art CF method based on the neural network proposed in recent years. Both methods are widely used as baselines in many studies, and they are used as the basic references to show the performance of other methods. SLIM is the representation of the earliest learning-based ICF method. NAIS and DeepICF are the main baselines to compared with FLA_{NAIS} and FLA_{DICF} to study the effects of the factor-level attention in recommendation.

Parameter Settings. All the considered methods in experiments use the pair-wise learning strategy. For fair comparisons, we paired each positive instance in the training set with four randomly sampled negative instances to train all methods. Four embedding sizes ($d\ in \{8, 16, 32, 64\}$) are tested in experiments. The learning rate is searched in [0.01, 0.001, 0.0001, 0.00001]. The smoothing parameter β is tuned in the range of [0.1, 0.9] with a step size of 0.2 for NAIS, DeepICF, and our methods. The best results of each method on the test datasets are reported below. Without particular specifying the value of β, the reported results are obtained by setting $\beta = 0.7$ for all the models. It is also worth mentioning that we used the learned user/items' embeddings by FISM as model initialization for NAIS, FLA_{NAIS}, DeepICF, and FLA_{DICF}. The benefits of pre-training for

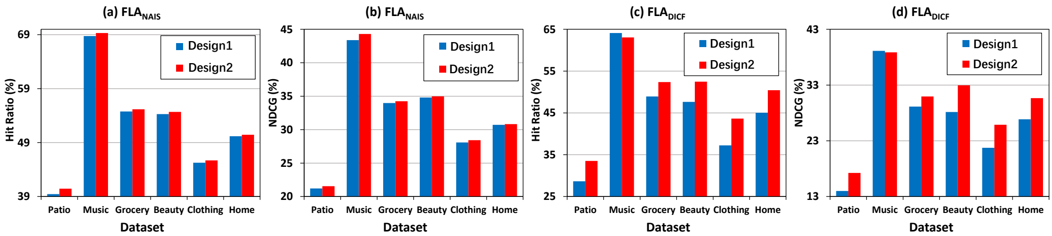

Fig. 3. Performance of different designs for combining item- and factor-level attentions in two ICF models: NAIS and DeepICF.

NAIS and DeepICF have been demonstrated in [27] and [72], respectively. We also study its effects to FLA$_{NAIS}$ and FLA$_{DICF}$ in Section 5.5.

5.2 Performance Comparisons of Different Designs (RQ1)

In this section, we report the performance of different designs for combining item-level and factor-level attentions in ICF models, namely, *Design 1* and *Design 2* as described in Section 3. Figure 3 shows the performance (in terms of HR and NDCG) of applying the two designs to NAIS and DeepICF (i.e., FLA$_{NAIS}$ and FLA$_{DICF}$) on the six evaluation datasets.

From the results, we can see that for the NAIS model, the *Design 2* can obtain consistently and slightly better results than the *Design 1*; for the DeepICF model, *Design 2* yields much better performance than the *Design 1* across the other five datasets besides the "Music" dataset. The better performance of *Design 2* is largely attributed to its simple design with less parameters, making the model easier to be trained and more resistant to overfitting. Because DeepICF has much more parameters than NAIS, the improvement of *Design 2* over *Design 1* in DeepICF is larger than it in NAIS. The marginally better performance of *Design 1* over *Design 2* in DeepICF on the Music dataset might because the relatively denser interactions between users and items, and this needs further validation. [6] Because of the better performance of *Design 2* in our experiments, in the following sections, all the reported results of FLA$_{NAIS}$ and FLA$_{DICF}$ are based on *Design 2*.

5.3 Model Comparison (RQ2)

In this section, we compare the performance of our factor-level attention (FLA) enhanced ICF models with all the adopted competitors. The results of all compared methods over all the test datasets are reported in Table 3 in terms of HR@10 and NDCG@10. For a fair comparison, the reported results are based on the same embedding size $d = 16$ for all the methods.

First, we would like to validate the effects of factor-level attentions on enhancing the performance of ICF models by comparing FLA$_{NAIS}$ and FLA$_{DICF}$ to NAIS and DeepICF, respectively. From the table, we can observe that with the consideration of factor-level attentions, the performance of NAIS and DeepICF can achieve better performance in most cases across the six datasets, which are of different scales and sparsity. Note that both NAIS and DeepICF already consider the different importance of items to the target item (i.e., item-level attention), the better performance of FLA$_{NAIS}$ and FLA$_{DICF}$ demonstrates that differentiating the contributions of different factors (i.e., factor-level attention) can further improve the performance. The results validate our main assumption that users attend to different factors of varied items and the incorporation of factor-level attention into ICF models are beneficial. Another interesting observation is that NAIS consistently outperforms

[6]Note that the "Patio" dataset is the densest one among all the datasets, however, this dataset is too small for training a deep model. This is also demonstrated by the results of MLP and DeepICF in Table 3.

Table 3. Performance of HR@10 and NDCG@10 of compared approaches at embedding size 16. Noticed that the values are reported by percentage with '%' omitted.

Methods	**Patio**		**Music**		**Grocery**		**Beauty**		**Clothing**		**Home**	
	HR	**NDCG**	**HR**	**NDCG**	**HR**	**NDCG**	**HR**	**NDCG**	**HR**	**NDCG**	**HR**	**NDCG**
BPR	36.71	19.91	66.40	40.69	50.00	29.80	48.65	30.60	39.50	22.99	43.02	25.18
MLP	33.39	16.99	59.00	35.14	47.75	28.18	42.42	24.23	34.40	19.49	45.34	26.89
SLIM	36.06	21.76	56.99	40.76	42.14	28.63	39.34	27.11	27.23	18.50	28.89	18.59
FISM	31.32	15.37	56.40	33.70	49.38	29.65	49.48	30.21	40.55	24.00	48.93	29.44
NAIS	40.21	21.68	68.76	43.96	54.58	33.77	54.02	34.75	45.03	27.95	50.02	30.71
FLA_{NAIS}	40.45	21.55	69.32	44.31	55.17	34.25	54.69	34.98	45.69	28.43	50.46	30.82
DeepICF	26.57	12.42	60.96	37.31	50.94	31.26	49.76	30.85	43.24	26.56	48.98	29.83
FLA_{DICF}	33.51	17.23	63.06	38.87	52.37	30.95	52.48	32.95	43.64	25.88	50.43	30.66

DeepICF with a large margin, especially on the smaller datasets. This is because DeepICF adopts deep networks which often require large-scale data in training for a good performance. As we can see, with the increasing of the data scale, the performance of DeepICF becomes closer to that of NAIS. Besides the scale of the data size, the sparse interactions of most users in the training datasets also negatively affect the performance of deep-learning based models (see the performance of MLP), because we adopted the 5-core version of those Amazon datasets in experiments. [7] NAIS is a direct extension from FISM by considering the item-level attention, we can see that a large improvement of NAIS over FISM; and with the additionally considering factor-level attention, FLA_{NAIS} can only slightly outperform NAIS. This is because NAIS itself is a very competitive ICF model (achieving a large improvement over the FISM). More importantly, FLA_{NAIS} attempts to capture the factor-level preference of users on items, which needs more interactions or side information to model such a fine-grained level preference on items. In this work, because only the interaction information is exploited and the interactions are fairly sparse for most users (most users have less than 10 interactions in the training data), it is difficult to model the fine-grained preference well, resulting in marginally improvement of FLA_{NAIS} over NAIS. Despite the limited information of the training data, we can still observe a consistent performance improvement by a small modification of the ICF models - replacing item-level attention (a scalar weight) with our proposed attention network (a weight vector), which is encouraging.

Second, we compare the performance of all the adopted methods. There are some interesting findings: 1) BPR is very competitive when the sizes of datasets are small, such as "Patio", "Music", and "Grocery". It performs the best besides NAIS and FLA_{NAIS} on "Patio" and "Music". SLIM is to learn a complete item-item similarity matrix and reconstruct the interaction matrix. It also performs well when the dataset is small. Reminder that the main drawbacks of SLIM are its scalability and generalization capability to un-interacted items. For larger datasets, FISM yields better performance than SLIM. 2) Because deep-based models often need large-scale training data for good performance, we can see that MLP and DeepICF do not perform well on small datasets. MLP surpasses BPR on the largest dataset "HOME", and DeepICF cannot compete BRP, MLP, SLIM, and FISM on the two smallest datasets (e.g., "Patio" and "Music"), even it considers both item- and factor-level attentions. 3) NAIS consistently preforms best among the baselines, which indicates the importance

[7]To validate this viewpoint, we conducted another experiment, in which the dataset is much larger and users/items with less than 20 interactions are removed. On this dataset, we observed a better performance of DeepICF over NAIS. Because this is not our focus in this study, we omitted the result here.

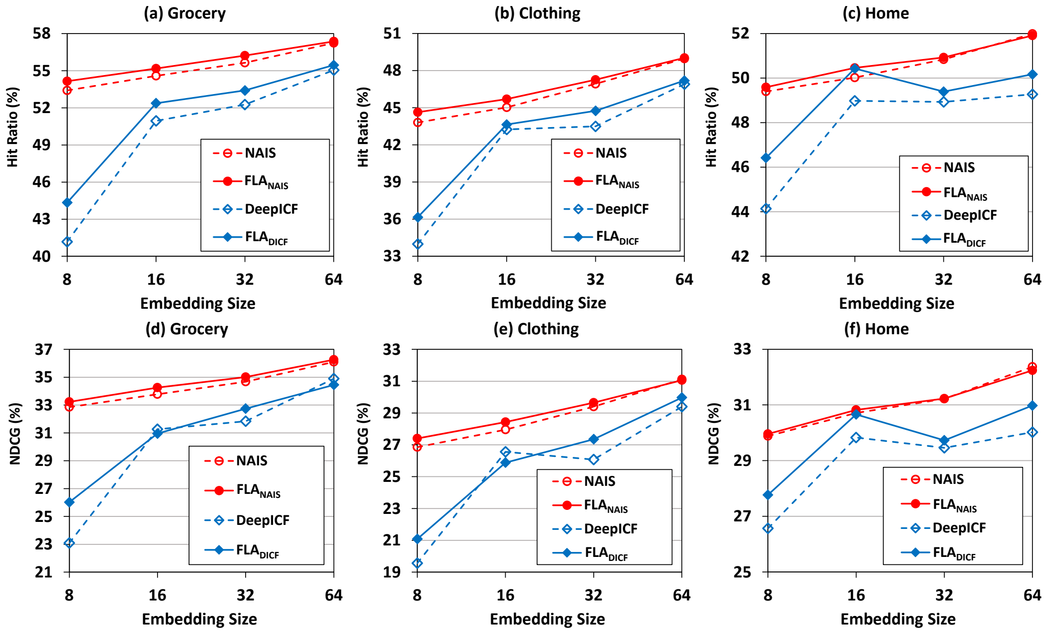

Fig. 4. Performance of HR@10 and NDCG@10 *w.r.t.* the number of embedding sizes on three datasets.

of differentiating the different contributions of items. Note that the performance of our factor-level attention enhanced models depends on performance of the backbone models. Although FLA_{DICF} obtains much better results than DeepICF, it is still inferior to NAIS in this experiment. Overall, the best performance is obtained by FLA_{NAIS}, which further enhances the performance of NAIS with the consideration of factor-level attention.

5.4 Hyper-parameter Analysis (RQ3)

In this section, we analyze the influence of two hyper-parameters, i.e., embedding size d and smoothing exponent β, on the performance of our factor-level attention enhanced ICF models.

Effect of embedding size. For analyzing the effect of the embedding size for the performance improvement of the factor-level attention module, we test FLA_{NAIS} and FLA_{DICF} with their counterparts with respect to different embedding sizes.The results on three relatively larger datasets are reported in Figure 4. Firstly, we can have a clearly observation which is consistent with many previous studies: the performance (in terms of accuracy) of all models is increasing with a larger embedding size, which is attributed to the increasing representation capability of the larger embedding size. Note that when the embedding size continue increasing, there is a risk of overfitting, which has not been observed in this study because the largest embedding size in our experiments is 64. A more interesting observation is that our factor-level attention enhanced models obtain larger the performance gain with a smaller embedding size. This observation is more consistent for the improvement of FLA_{NAIS} over NAIS. The underlying reason is that when the embedding size, i.e., number of item factors, is smaller, it is relatively easier for the attention network to learn good factor-level attention weights (because of less factors). Therefore, our models can benefits more from the factor-level effects for user preference modeling, leading to better performance. Comparing to NAIS, DeepICF is more difficult to train. As a result, the performance gain of FLA_{DICF}

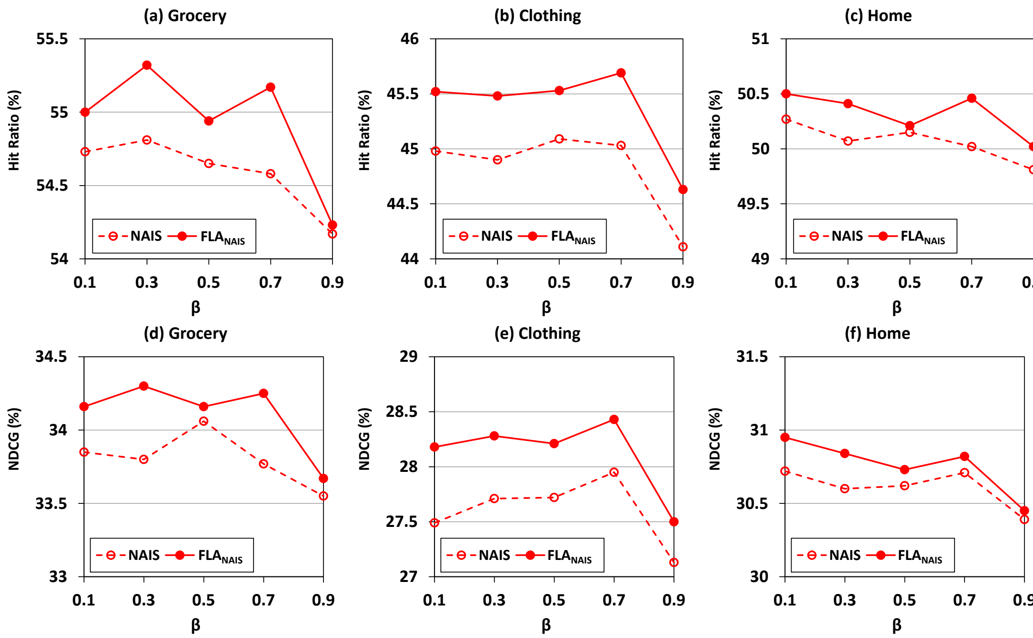

Fig. 5. Performance of FAMR-NAIS on different smoothing exponent β.

over DeepICF is not very stable, although the largest gain for the three datasets is also achieved when the embedding size is 8.

Effect of the smoothing exponent. Because of the different numbers of history items for users, using the standard softmax normalization can excessively penalize the weights of active users with a long history. We use the performance of NAIS and FLA$_{NAIS}$ to demonstrate the effects of the smoothing factor β. We omit results of DeepICF and FLA$_{NAIS}$, as they adopted the same smoothing strategy and similar results are observed. Figure 5 shows the performance of NAIS and FLA$_{NAIS}$ with different β. We can see FLA$_{NAIS}$ consistently outperforms NAIS; and the general trends of the performance change for both methods are similar, indicating the smoothing effects are the same to the two methods. The optimal value of β depends on the target datasets. It seems 0.7 is a good choice across all the datasets. Note that when $\beta = 1$, it means that a standard attention method is used to normalize the attention weights. As pointed out in [27], a standard stetting does not work well because of the large variance of the length of user histories. We can observe a dramatic performance degradation when $\beta = 0.9$, indicating it already becomes insufficient to reduce the punishment on the attention weights of active users. This also demonstrates the importance of smoothing the denominator in the softmax function for attention weight computation on user behavior data.

5.5 Effect of Pre-training (RQ4)

As pre-training has been widely used for model training and demonstrated good performance, we also employ this technique in our experiments. To demonstrate the effects of pre-training, we compare the factor-level attention enhanced models with (denoted by FLA$_{NAIS}$/w and FLA$_{DICF}$/w) and without pre-training (denoted by FLA$_{NAIS}/o$ and FLA$_{DICF}/o$). In our implementation, we used the learned user/items' embeddings by FISM as model initialization for both FLA$_{NAIS}$ and

Table 4. Performance of FLA$_{NAIS}$ and FLA$_{DICF}$ with (/w) and without (/o) pre-training at embedding size 16.

Methods	Patio		Music		Grocery		Beauty		Clothing		Home	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
FLA$_{NAIS}$/o	37.60	19.88	64.25	40.53	52.26	31.69	48.35	29.67	37.37	22.18	46.64	27.85
FLA$_{NAIS}$/w	**40.45**	**21.55**	**69.32**	**44.31**	**55.17**	**34.25**	**54.69**	**34.98**	**45.69**	**28.43**	**50.46**	**30.82**
FLA$_{DICF}$/o	23.96	11.64	34.31	19.46	33.04	18.74	33.06	19.79	18.05	9.07	40.30	24.07
FLA$_{DICF}$/w	**33.51**	**17.23**	**63.06**	**38.87**	**52.37**	**30.95**	**52.48**	**32.95**	**43.64**	**25.88**	**50.43**	**30.66**

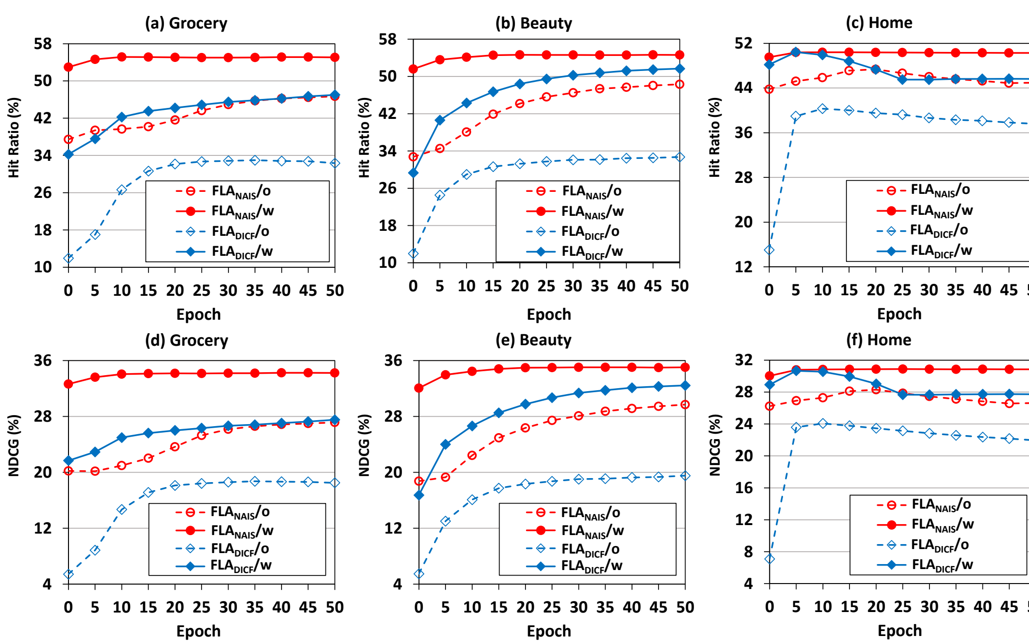

Fig. 6. Performance of FLA$_{NAIS}$ and FLA$_{DICF}$ with (/w) and without (/o) pre-training at embedding size 16 at each epoch.

FLA$_{DICF}$. For FLA$_{NAIS}/o$ and FLA$_{DICF}/o$, the hyper-parameters have been separately tuned. Note that we can also use the learned embeddings by NAIS and DeepICF as model initialization for FLA$_{NAIS}$ and FLA$_{DICF}$. Because NAIS and DeepICF themselves also need pre-training for faster convergence and better performance [27, 72], it is cumbersome to use their learned embedding in practice. Therefore, we used the embedding learned in FISM as pre-training results for simplicity and consistency. The comparison results with and without pre-training are shown in Table 4. It can be seen that with the pre-training, the performance of both methods have been significantly improved. By initializing the model randomly, it is easier to be trapped in local minimums, which hurts the performance of the model. Beyond performance improvements, pre-training can also accelerate the convergence speed. Figure 6 shows the convergence rate of FLA$_{NAIS}$ with (FAMR/w) and without (FAMR/o) pre-training. We find that in the three datasets *Grocery*, *Beauty*, and *Home*, there is a faster convergence speed with pre-training than without pre-training. The case with pre-training basically converges at the tenth epoch, while the case without pre-training takes longer. For the Home dataset, the situation without pre-training even drops after 20 epochs. This is because the random initialization is more difficult to find the optimal solution.

6 RELATED WORK

6.1 Collaborative Filtering

Collaborative filtering (CF) [31, 37, 48] has long been recognized as an effective approach in recommendation over the past decades. Based on the standpoint of the interacted instances, CF methods can be classified into two categories: user-based CF (UCF) and item-based CF (ICF). The former one recommends a user with the items favored by her similar users; and the latter one recommends a user with the items that are similar to the items she liked in the history. UCF has been extensively studied in both academia and industry. A typical UCF method is matrix factorization (MF) [38], which represents users and items as feature vectors in the same embedding space based on the user-item interactions, and then predicts the preference of a user to an item by an interaction function (i.e., inner product) between their embedding vectors. This simple idea has achieved great success in the Netflix contest and many variants have been developed later on, such as WRMF [31], SVD++ [36], BPR [50], NeuMF [28]. Although UCF has achieved significant progress, a big limitation is that the UCF models require to be re-trained when new interactions come in, which is unacceptable in real-time recommender systems [27, 72]. In contrast, ICF predicts user preference to a target item by estimating the similarity scores between the previously interacted items of this user and the target one, which enables ICF to easily incorporate new interactions into the preference modeling. Due to the nice property of ease online updating, ICF models are favored by industry and have been widely-adopted in real recommender systems [17, 24, 63].

Early ICF models leverage heuristic metrics, such as cosine similarity [51] or Pearson correlation coefficient [35] to calculate the similarity, which require quite a lot of manual tuning when adapting to another brand-new dataset. In order to tackle this limitation, several data-driven methods have been proposed [12, 70]. For example, SLIM [47] learns a complete item-item similarity matrix by minimizing the errors between the reconstructed rating matrix and the ground-truth. However, the transductive relations are omitted since only co-interacted items are considered. FISM [33] adopts the inner product of item embeddings between the historical items and the target item for prediction. To model the preference of like-minded usersets, Christakopoulou et al. [11] proposed a global and local SLIM (GLSLIM) method, which applies different SLIM models to capture the preference of different user subsets. An early neural network based ICF mdoel is the CADE model [70], which learns the item similarity by using nonlinear auto-encoder architecture. He et al. [27] claimed that the historically interacted items of a user contributed differently for the current user preference for the target item, they therefore developed an attention-based method NAIS to assign different weights to the historical items for better capturing the user preferences. Christakopoulou et al. [10] pointed out that high-order item relations also provide valuable information for user preference modeling. They proposed a higher-order sparse linear method (HOSLIM) which extends the SLIM model to learn the item-itemset similarity for capturing the higher-order relations. More recently, Xue et al. [72] proposed a DeepICF model, which captures the higher-order item relations by stacking multiple layer over the second-order item relations with a non-linear way. Despite great progress has been achieved by those ICF models, those models have not considered user diverse intents towards different items in an explicitly way. In this paper, we make an effort to model user diverse intents at the factor-level (i.e., each factor is considered as an intent dimension) in the ICF model and propose a factor-level attention method to enhance the performance of ICF models.

6.2 Attention-based Recommendation

The attention mechanism has been widely-used in deep learning methods and achieved great success in many tasks in computer vision and natural language processing. With the widespread application of deep learning in recommendation, this technique has also been used in various ways in recommender systems in order to model user preference more accurately. Many attention-based recommender systems have been developed. A comprehensive survey of attention-based recommender system is out-of-the-scope of this paper. In this section, we briefly review the three paradigms of using attention-mechanism used in recommender systems.

Item-level attention. As discussed, historically interacted items have different contributions to model users' preference. Therefore, it is important to assign different weights to the items for more accurate recommendation [20, 75, 76]. NAIS [27] and DeepICF [72] are typical examples of this paradigm. Besides the ICF models, the item-level attention has also been used in graph convolution network (GCN) based recommender systems. The core of GCN-based recommendation models is that the embeddings of users/items are iteratively updated by aggregating information from their local neighbors (i.e., interacted items/users) [25, 32, 73]. The attention mechanism is introduced to differentiate the different contributions of neighboring nodes in the user/item embedding learning process [42, 65, 67]. Another widely applied task for item-level attention is the the session-based recommendation task. Because interacted items in a session are typically sparse, it becomes very crucial to identify important items for user intent inference [60]. A general framework is to use a recurrent neural network to learn the hidden states of items inside a session, followed by an attention model on the items' hidden representations to capture the main purpose of users [39, 57]. Recently, the self-attention blocks, such as Transformer [58] and BERT [18] have also been applied to the session-based recommendation [1, 34, 56].

Feature-level attention in side information. The attention mechanism has become a standard component in the side information enriched recommender system, in order to extract effective features from the side information to represent item features or user preference. The most widely used side information is review and user/item attributes. At the beginning, the attention mechanism is only used to assign different weights on the review-level for learning user and item embeddings [3, 43, 69]. Later on, the review-aware recommender systems exploit the reviews at a more fine-grained level by applying the attention mechanism in a hierarchical manner [13, 44]: 1) first attending important words of a review (i.e., word-level) to learn better review representations, and then 2) assigning different weights to review representations for user and item embedding learning. Beyond the two-layer of hierarchical attention network design, Wu et al. [68] proposed to additionally encode the sentence-level attention in the review and developed a three-tier attention network for recommendation. Besides, there are also aspect-aware attention-based recommendation models [9, 22], which extract aspects from the reviews and then assign weights to different aspects in the user preference modeling.

Attribute information is often used in factorization machine [49] and graph-based models [42], especially knowledge-graph (KG) based recommendation models [23, 30, 53, 59, 61]. A representative attention-based FM based model is the AFM model [71], which learns the importance of each feature interactions from data via a neural attention network. In the KG-based recommender systems, the attributes of items/users are taken as node entities in the graph. There are two typical ways of applying KGs: embedding-based and meta-path based. In the embedding-based models, such as KGAT [61], RippleNet [59], AKGE [52], and A^2-GCN [42], the attention mechanism is often used to learn the importance of neighbor nodes during the embedding propagation. For meta-path approaches, the attention mechanism can be applied inside a meta-path to learn representation of the meta-paths or directly attends to different meta-paths. A typical meta-path based recommendation approach is MCRec [30], which first uses the attention mechanism to learn the representation of meta-paths and then applies it to assign the weights of different meta-paths for final user representation learning.

The attention mechanism is also used in visual-aware and multimedia recommendation. For example, Chen et al. [4] proposed an ACF model for multimedia recommendation, in which a component-level attention model is used to capture the user's different preferences on different components, e.g., certain actions in a video; and an item-level attention model is leveraged to treat historically interacted items differently. In [5], a visually explainable recommendation model is presented to capture use attention on different regions of images based on attention neural networks.

Factor-level attention. Different from the above methods, we use the attention mechanism to attend each factor of an item embedding, aiming to capture users' diverse intents towards various items. In other words, the attention weights are assigned to different factors of the target item embedding to capture the user's specific preference on this item. From this perspective, the most similar method is A^3NCF [7]. In this method, for each user-item pair, it learns the attentive weights for each factor by taking the user's and item's embedding, as well as their text-based representations learned from review into an attentive neural network. Note that there is a big difference between the A^3NCF and the method presented in this work. The A^3NCF is a user-based CF model which learns the attentive weights based on the target user and item embeddings; and our method here is designed for item-based CF methods, which do not modeling user embeddings.

6.3 Diverse Preference Modeling

The underlying rationale of factor-level attention is that a user's intent to different items could be diverse. Traditional recommender systems often represent a user preference with a fix embedding vector, which is then used to match the vectors of different items for preference prediction. This process does not differentiate user intents on different items. In recent few years, researchers start to pay attention to model the diverse preferences of users towards different items and proposed several methods. Cheng et al. [6–8] proposed to model user intents on different aspects of items. They first applied topic models on side information (e.g., reviews and images) to analyze user interests on different aspects of items. These aspects are then linked to the factors of (user/item's) embeddings (learned by matrix factorization [6, 8] or neural networks [7]). For a target user-item pair, a unique weight vector is learned to represent this user's attention on different factors of the target item. This unique weight vector is expected to capture the user's intent (e.g., on which aspects) to the target item. Following this idea, Chin et al. [9] presented an end-to-end neural recommendation model called ANR, which exploits the review information to model user diverse preference on different aspects of items. Later on, Liu et al. [41] presented a metric-learning based recommendation model, which uses an attentive neural network to estimate user attention on different aspects of the target item by exploiting the item's multimodal features (e.g., review and image).

To take the user diverse preference on items into consideration, another line of work is to dynamically adapt the target user's or item's embedding to accurately predict the user preference to the target item. For example, CMN [20] adapts the target user embedding based on the selected most influential neighbor users, whose influential scores are computed according to the target item. MARS [75] adopts a different strategy, which adapts the user vector embedding based on the most influential item vectors of the target item. In contrast, DIN [76] adapts the target item embedding based on the user's previously purchased items. More recently, the disentangled representation learning approach has been applied in recommendation for disentangled embedding learning. A representative method is the disentangled graph collaborative filtering (DGCF) method proposed by Wang et al. [64]. In this method, different intents are represented as different chunks in the embedding vector and a distance correlation regularization is applied to make those chunked representations independent. Different from this method, DisenHAN [66] learns the disentangled representations by aggregating aspect features from different meta relations in a heterogeneous information network (HIN).

Apparently, the method presented in this work is fundamentally different from the above method. Our method predicts user preference on the target item by attending each factor in the item embedding vectors of the historical items. All the above methods fall into the user-based CF approach, and they use the learned user embedding to analyze the user intent to the target item.

7 CONCLUSION

In this work, we advocate the importance of modeling user diverse intents to items in recommendation and present a factor-level attention model for ICF models. The proposed model distinguishes the contribution of different factors of a historical item to the target item for prediction. In this way, our model captures user intents at the factor-level of item embeddings. In addition, we design a light attention neural network to combine the item- and factor-level attentions for neural ICF models. It is model-agnostic and easy-to-implement in ICF models. To show its effectiveness, we apply it to the recently proposed NAIS and DeepICF models and evaluate its effectiveness on six Amazon datasets. The superior performance over several competitive baselines demonstrates the benefits of modeling the impact of different factors (in item embeddings) for recommendation.

We hope this work can shed light on modeling user preference at a fine-grained level (like the factor-level) to capture user diverse intents on adopting items for recommendation, and can motivate more researches in this direction in the future. Because it typically needs more data to model user preference on such a fine-grained level, an interesting future study is to exploit the rich side information, such as reviews and knowledge-graphs, in the modeling. In addition, how to leverage the fine-grained preference modeling to provide better interpretation for recommendations is also worth studying.

REFERENCES

[1] Pham Hoang Anh, Ngo Xuan Bach, and Tu Minh Phuong. 2019. Session-Based Recommendation with Self-Attention. In *Proceedings of the Tenth International Symposium on Information and Communication Technology*. ACM, 1–8.

[2] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. 2017. A generic coordinate descent framework for learning from implicit feedback. In *Proceedings of the 26th International Conference on World Wide Web*. 1341–1350.

[3] Chong Chen, Min Zhang, Yiqun Liu, and Shaoping Ma. 2018. Neural Attentional Rating Regression with Review-level Explanations. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. ACM, 1583–1592.

[4] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 335–344.

[5] Xu Chen, Hanxiong Chen, Hongteng Xu, Yongfeng Zhang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2019. Personalized Fashion Recommendation with Visual Explanations based on Multimodal Attention Network: Towards Visually Explainable Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 765–774.

[6] Zhiyong Cheng, Xiaojun Chang, Lei Zhu, Rose Catherine Kanjirathinkal, and Mohan S. Kankanhalli. 2019. MMALFM: Explainable Recommendation by Leveraging Reviews and Images. *ACM Trans. Inf. Syst.* 37, 2 (2019), 16:1–16:28.

[7] Zhiyong Cheng, Ying Ding, Xiangnan He, Lei Zhu, Xuemeng Song, and Mohan S Kankanhalli. 2018. A3NCF: An Adaptive Aspect Attention Model for Rating Prediction.. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 3748–3754.

[8] Zhiyong Cheng, Ying Ding, Lei Zhu, and Kankanhalli Mohan. 2018. Aspect-aware latent factor model: Rating prediction with ratings and reviews. In *Proceedings of the 27th International Conference on World Wide Web Companion*. IW3C2, 639–648.

[9] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based neural recommender. In *Proceedings of the 2018 ACM on Conference on Information and Knowledge Management*. ACM, 147–156.
[10] Evangelia Christakopoulou and George Karypis. 2014. Hoslim: Higher-order sparse linear method for top-n recommender systems. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 38–49.
[11] Evangelia Christakopoulou and George Karypis. 2016. Local item-item models for top-n recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 67–74.
[12] Evangelia Christakopoulou and George Karypis. 2018. Local latent space models for top-n recommendation. In *Proceedings of the 24th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1235–1243.
[13] Dawei Cong, Yanyan Zhao, Bing Qin, Yu Han, Murray Zhang, Alden Liu, and Nat Chen. 2019. Hierarchical attention based neural network for explainable recommendation. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*. ACM, 373–381.
[14] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. ACM, 191–198.
[15] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. 2010. The YouTube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*. 293–296.
[16] Mukund Deshpande and George Karypis. 2004. Item-based top-N Recommendation Algorithms. In *ACM Transactions on Information Systems*, Vol. 22. 143–177.
[17] Mukund Deshpande and George Karypis. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems* 22 (2004), 143–177.
[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL, 4171–4186.
[19] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12 (2011), 2121–2159.
[20] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative memory network for recommendation systems. In *Proceedings of the 41st international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 515–524.
[21] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 27th International Conference on World Wide Web Companion*. ACM, 1775–1784.
[22] Xinyu Guan, Zhiyong Cheng, Xiangnan He, Yongfeng Zhang, Zhibo Zhu, Qinke Peng, and Tat-Seng Chua. 2019. Attentive Aspect Modeling for Review-Aware Recommendation. *ACM Trans. Inf. Syst.* 37, 3 (2019).
[23] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A Survey on Knowledge Graph-Based Recommender Systems. *CoRR* abs/2003.00911 (2020).
[24] Taolin Guo, Junzhou Luo, Kai Dong, and Ming Yang. 2019. Locally differentially private item-based collaborativeItem-based collaborative filtering recommendation algorithms filtering. *Information Sciences* 502 (2019), 229–246.
[25] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd international ACM SIGIR conference on Research and development in Information Retrieval*. 639–648.
[26] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial personalized ranking for recommendation. In *Proceedings of the 41st international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 355–364.
[27] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 30 (2018), 2354–2366.
[28] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. ACM, 173–182.
[29] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. 1999. An Algorithmic Framework for Performing Collaborative Filtering. In *Proceedings of the 22nd international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 230–237.
[30] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. 2018. Leveraging Meta-path based Context for Top- N Recommendation with A Neural Co-Attention Model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1531–1540.
[31] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 263–272.

[32] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20, 4 (2002), 422–446.
[33] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 659–667.
[34] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining*. IEEE, 197–206.
[35] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.
[36] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 426–434.
[37] Yehuda Koren. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data* 4 (2010), 1–24.
[38] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42 (2009), 30–37.
[39] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1419–1428.
[40] David C Liu, Stephanie Rogers, Raymond Shiau, Dmitry Kislyuk, Kevin C Ma, Zhigang Zhong, Jenny Liu, and Yushi Jing. 2017. Related pins at pinterest: The evolution of a real-world recommender system. In *Proceedings of the 26th International Conference on World Wide Web Companion*. 583–592.
[41] Fan Liu, Zhiyong Cheng, Changchang Sun, Yinglong Wang, Liqiang Nie, and Mohan S. Kankanhalli. 2019. User Diverse Preference Modeling by Multimodal Attentive Metric Learning. In *Proceedings of the 27th ACM International Conference on Multimedia*. ACM, 1526–1534.
[42] Fan Liu, Zhiyong Cheng, Lei Zhu, Chenghao Liu, and Liqiang Nie. 2021. Aˆ 2-GCN: An Attribute-aware Attentive GCN Model for Recommendation. *IEEE Trans. Knowl. Data Eng.* (2021), to appear.
[43] Hongtao Liu, Wenjun Wang, Hongyan Xu, Qiyao Peng, and Pengfei Jiao. 2020. Neural Unified Review Recommendation with Cross Attention. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. ACM, 1789–1792.
[44] Hongtao Liu, Fangzhao Wu, Wenjun Wang, Xianchen Wang, Pengfei Jiao, Chuhan Wu, and Xing Xie. 2019. NRPA: Neural Recommendation with Personalized Attention. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1233–1236.
[45] Andrew L. Maas, Awni Y. Hannum, and Andrew Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning*.
[46] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*. ACM, 165–172.
[47] Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th International Conference on Data Mining*. IEEE, 497–506.
[48] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 502–511.
[49] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.
[50] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI, 452–461.
[51] Badrul Munir Sarwar, George Karypis, Joseph A Konstan, John Riedl, et al. 2001. Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web Companion* 1 (2001), 285–295.
[52] Xiao Sha, Zhu Sun, and Jie Zhang. 2019. Attentive Knowledge Graph Embedding for Personalized Recommendation. *CoRR* abs/1910.08288 (2019). http://arxiv.org/abs/1910.08288
[53] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. 2019. Heterogeneous Information Network Embedding for Recommendation. *IEEE Trans. Knowl. Data Eng.* 31, 2 (2019), 357–370.
[54] Brent Smith and Greg Linden. 2017. Two decades of recommender systems at Amazon. com. *Ieee internet computing* 21 (2017), 12–18.
[55] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).
[56] Shiming Sun, Yuanhe Tang, Zemei Dai, and Fu Zhou. 2019. Self-Attention Network for Session-Based Recommendation With Streaming Data Input. *IEEE Access* 7 (2019), 110499–110509.

[57] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. ACM, 17–22.
[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. MIT Press, 5998–6008.
[59] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Exploring High-Order User Preference on the Knowledge Graph for Recommender Systems. *ACM Trans. Inf. Syst.* 37, 3 (2019), 32:1–32:26.
[60] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Menmet Orgun, and Defu Lian. 2019. A Survey on Session-based Recommender Systems. *CoRR* abs/1902.04864 (2019). arXiv:1902.04864 http://arxiv.org/abs/1902.04864
[61] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 950–958.
[62] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.
[63] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 165–174.
[64] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled Graph Collaborative Filtering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. ACM, 1001–1010.
[65] Xiao Wang, Ruijia Wang, Chuan Shi, Guojie Song, and Qingyong Li. 2020. Multi-Component Graph Convolutional Collaborative Filtering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*. AAAI Press, 6267–6274.
[66] Yifan Wang, Suyao Tang, Yuntong Lei, Weiping Song, Sheng Wang, and Ming Zhang. 2020. DisenHAN: Disentangled Heterogeneous Graph Attention Network for Recommendation. In *The 29th ACM International Conference on Information and Knowledge Management*. ACM, 1605–1614.
[67] Yinwei Wei, Zhiyong Cheng, Xuzheng Yu, Zhou Zhao, Lei Zhu, and Liqiang Nie. 2019. Personalized Hashtag Recommendation for Micro-videos. In *Proceedings of the 27th ACM International Conference on Multimedia*. ACM, 1446–1454.
[68] Chuhan Wu, Fangzhao Wu, Junxin Liu, and Yongfeng Huang. 2019. Hierarchical User and Item Representation with Three-Tier Attention for Recommendation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*. Association for Computational Linguistics, 1818–1826.
[69] Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo. 2019. A Context-Aware User-Item Representation Learning for Item Recommendation. *ACM Trans. Inf. Syst.* 37, 2 (2019), 22:1–22:29.
[70] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 153–162.
[71] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. ijcai.org, 3119–3125.
[72] Feng Xue, Xiangnan He, Xiang Wang, Jiandong Xu, Kai Liu, and Richang Hong. 2019. Deep Item-based Collaborative Filtering for Top-N Recommendation. *ACM Transactions on Information Systems* 37 (2019), 33.
[73] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 974–983.
[74] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. 2016. Discrete collaborative filtering. In *Proceedings of the 39th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 325–334.
[75] Lei Zheng, Chun-Ta Lu, Lifang He, Sihong Xie, Huang He, Chaozhuo Li, Vahid Noroozi, Bowen Dong, and S Yu Philip. 2019. MARS: Memory attention-aware recommender system. In *2019 IEEE International Conference on Data Science and Advanced Analytics*. IEEE, 11–20.
[76] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1059–1068.