Decomposed Collaborative Filtering: Modeling Explicit and Implicit Factors For Recommender Systems

Hao Chen
maxhao@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, Shanghai

Xin Xin
x.xin.1@research.gla.ac.uk
University of Glasgow
Glasgow, Scotland

Dong Wang*
wangdong@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, Shanghai

Yue Ding
dingyue@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, Shanghai

ABSTRACT

Representation learning is the keystone for collaborative filtering. The learned representations should reflect both explicit factors that are revealed by extrinsic attributes such as movies' genres, books' authors, and implicit factors that are implicated in the collaborative signal. Existing methods fail to decompose these two types of factors, making it difficult to infer the deep motivations behind user behaviors, and thus suffer from sub-optimal solutions. In this paper, we propose Decomposed Collaborative Filtering (DCF) to address the above problems. For the explicit representation learning, we devise a user-specific relation aggregator to aggregate the most important attributes. For the implicit part, we propose Decomposed Graph Convolutional Network (DGCN), which decomposes users and items into multiple factor-level representations, then utilizes factor-level attention and attentive relation aggregation to model implicit factors behind collaborative signals in fine-grained level. Moreover, to reflect more diverse implicit factors, we augment the model with disagreement regularization. We conduct experiments on three public accessible datasets and the results demonstrate the significant improvement of our method over several state-of-the-art baselines. Further studies verify the efficacy and interpretability benefits bought from the fine-grained implicit relation modeling. Our Code is available on https://github.com/cmaxhao/DCF.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Recommender systems, Graph neural networks, Knowledge Graph

*Corresponding author.

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441826

ACM Reference Format:
Hao Chen, Xin Xin, Dong Wang, and Yue Ding. 2021. Decomposed Collaborative Filtering: Modeling Explicit and Implicit Factors For Recommender Systems. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441826

1 INTRODUCTION

Recommender systems have been widely applied in web applications to overcome information overload and meet user's personal interests. One of the most successful techniques for recommender systems is collaborative filtering (CF), which assumes that the user preference can be aggregated from historical interacted items (item-based CF) or similar users (user-based CF). The keystone when training model-based CF methods is the construction of item and user representations. After that, the prediction score can be obtained through the interactions (e.g., inner product [13] or neural networks [8]) between user and item representations.

In real-world scenarios, the motivations behind user's decisions are complex and obscure. They can be motivated from extrinsic attributes. For example, a user may watch a fiction movie because he is a fan of this genre. They can also root from implicit factors implicated from the collaborative signal (co-interaction pattern) [30]. To better capture user preference, the learned representations should reflect both the explicit and implicit factors.

Conventional CF methods based on pure user-item interactions can only model the implicit factors. The co-interaction patterns revealed from implicit factors are coarse-grained and it's not convincing to provide user recommendations based on the explanation like "similar users also bought this". Recent works [25] attempt to model implicit factors in fine-grained level. However, it's still hard to find concrete semantic meaning for them, resulting in the difficulty of providing convincing and explainable recommendations.

Incorporating side information such as user profile and item description as explicit factors into the training of recommender systems becomes more common. Knowledge graphs (KGs) contain abundant semantic meaningful facts and relations, providing a natural solution to model explicit factors in CF. Plenty of researches [2, 23, 33] have been done to exploit KGs for recommendation. These works can be categorized into embedding-based methods and path-based methods [30]. Most of embedding-based methods [2, 23] introduce tasks from KG domain (e.g., relation

completion) as auxiliary tasks to guide the representation learning. However, they are not end-to-end fashions and whether the KG domain tasks conform to the nature of recommendation still needs to be investigated. Path-based methods [9, 21] extract paths or meta-paths from KGs. However, it's not scalable in practical usage because of the huge amount of items (users). Besides, most of these methods fail to consider fine-grained implicit factors which may still constitute the majority of motivations behind user behaviors. In fact, the explicit factors and implicit factors entangle with each other and affect the user's decision collaboratively.

Considering the limitations of existing methods, we believe that it's vital to develop a model that can capture fine-grained implicit and explicit factors behind user behaviors in a generalized and end-to-end manner. In this paper, we seamlessly unify user-item interaction graph and KG to construct a *Collaborative Relation-aware Graph* (CRG) which can be regarded as a kind of heterogeneous information network (HIN) [19]. CRG contains two types of relations between nodes: fine-grained implicit relations behind $Interaction$ and explicit relations, as illustrated in Figure 1. We propose Decomposed Collaborative Filtering (DCF) to model the fine-grained implicit relations behind $Interaction$ and explicit relations on CRGs. More precisely, for the explicit part, we devise a user-specific relation aggregator to mine the most important explicit motivations from user behaviors. To model implicit relations, we first decompose users and items into multiple fine-grained representations related to implicit relations, called factor-level representations. Then, we utilize factor-level attention to learn probability distribution over all implicit factors for each interacted item and then sum them up to formulate the representations of implicit factors. After that, we utilize attentive relation-wise aggregation to sum up these implicit factors as the final implicit representation. To avoid the issue that these implicit factors are reflecting similar motivations, we add disagreement regularization to guide them to learn more diverse representations. In addition, we unify fine-grained implicit relation aggregator and explicit relation aggregator as a generalized form to capture high-order connectivity for users and items. Generally speaking, our method can even infer potential semantic meaning for the implicit factor that drives user to choose the item. Our proposed method DCF is superior to existing methods in that: 1) we unveil the fine-grained implicit factors behind the interactions between users and items; 2) and we propose a generalized relation aggregator to aggregate neighbors for exploiting high-order connectivity, independent of tedious process of materializing meta-paths and specific formula term. The contributions of our paper are summarized as follows:

- We present Decomposed Graph Convolutional Network (DGCN), which can capture fine-grained implicit factors behind user behaviors based on factor-level attention and attentive relation aggregation.
- We propose a new method DCF, which unifies fine-grained implicit relations modeling and explicit relations modeling as a generalized graph convolutional operation, and it can be stacked in multiple layers to exploit high-order connectivity.
- We conduct extensive experiments on three public datasets to evaluate our proposed method. Experimental results show the effectiveness and interpretability of DCF.

2 RELATED WORK

Collaborative Filtering. The key of CF is to model user preferences based on similar users or items [17]. The most representative CF-based method is MF [13] which projects each user and item into latent representations and then utilizing inner product between them to obtain the predicted score. FISM [10] is an item-based CF model which characterizes the user representation as the mean aggregation of item embeddings which occur in his interaction history. Plenty of work has been done to enhance CF, such as incorporating user-specific information [4] and introducing local latent space [3]. Recently, utilizing the expressive deep learning approaches for CF has also become a hot research topic because of its effective feature extraction and end-to-end model training. NAIS [7] enhances FISM by replacing the mean aggregation with attention-based summation.

Graph Learning for Recommendation. With the great achievements in Graph Neural Networks (GNNs) [6, 12, 20], recent works have tried to apply GNNs in recommender system to learn better representations of users and items, especially when incorporating with KGs. SpectralCF [34] proposes a spectral convolution operation to learn latent factors of users and items from the spectral domain. PinSage [32] adopts GraphSAGE [6] on item-item graph to generate large-scale image recommendations. GC-MC [28] employs a graph convolutional autoencoder to construct embeddings of users and items. NGCF [24] proposed to use element-wise product between two connected nodes for explicitly conserving collaborative signals with message passing.

However, the above approaches ignore to capture the fine-grained implicit factors implicated in the collaborative signal. Although side information like KGs help to improve the recommendation quality [22, 23], the implicit factors still account for an important part to affect user behavior. DisenGCN [15] performs neighborhood routing to infer the latent factor behind the connectivity between two nodes in the graph. However, DisenGCN fails to make the disentangled representation distinct with each other and it's not designed for recommendation tasks in HIN.

3 PROBLEM FORMULATION

Collaborative Relation-aware Graph. In a typical recommendation scenario, we have historical interactions (*e.g.*, clicks and purchases) of users and items. Then we construct historical interaction data as user-item bipartite graph $\mathcal{G}_1 = \{(u, y_{uv}, v)|u \in \mathcal{U}, v \in \mathcal{V}\}$, where $\mathcal{U}$ and $\mathcal{V}$ denote user and item set, respectively. If there exists an observed interaction between u and v then $y_{uv} = 1$, otherwise $y_{uv} = 0$. In addition to the user-item bipartite graph, we also have a knowledge graph $\mathcal{G}_2$ that describes the relationship between items and side information (e.g. item attributes). $\mathcal{G}_2$ comprises of triples (h, r, t), where $h, t \in \mathcal{E}'$ and $r \in \mathcal{R}^+$. Here h and t denote head entity and tail entity, respectively. $\mathcal{E}'$ is the Entity set, r denotes relation type. We define explicit relation set $\mathcal{R}^+$ to represent the relationships in KG. For example, the triplet (*The Million Pound Note*, *book.book.author*, *Mark Twain*) means *The Million Pound Note*'s author is *Mark Twain*. Considering that the user behavior can also be formulated as a triplet $(u, Interaction, v)$ and the item v can occur as the head entity in the KG, we integrate user-item graph and KG as

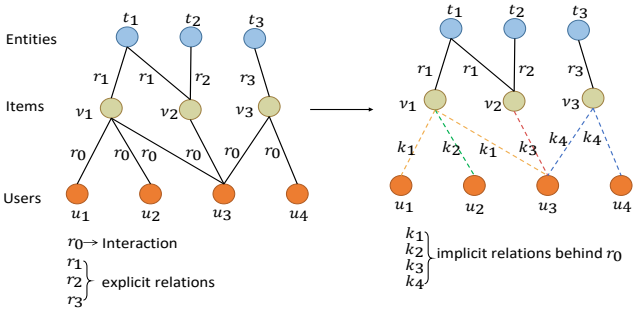

Figure 1: An example of collaborative relation-aware graph. $r_0 = 1$ represents the user-item *interaction*. In the right part we replace *Interaction* with fine-grained implicit relations (k_1, k_2, k_3, k_4) and the most important k_i factor determines the reason why the user choose a specific item.

a unified graph $\mathcal{G} = \{(h, r, t)|h, t \in \mathcal{E}, r \in \mathcal{R}\}$[1], where $\mathcal{E} = \mathcal{U} \cup \mathcal{E}'$ and $\mathcal{R} = \mathcal{R}^+ \cup \{Interaction\}$, here $Interaction = 1$ if u interacts with v. The left part of Figure 1 illustrates an example of CRG, the explicit relations $(r_1, r_2, r_3) \in \mathcal{R}^+$ and $r_0 = 1$ represents the user-item interaction. Since the motivation behind user's choice of a specific item is complex, we replace the *Interaction* set by defining an implicit relation set $\mathcal{R}^-$, then, the whole relation set of CRG is defined as $\mathcal{R} = \mathcal{R}^+ \cup \mathcal{R}^-$. The right part of Figure 1 illustrates the fine-grained implicit relations in CRG. $(k_1, k_2, k_3, k_4) \in \mathcal{R}^-$ represents four different underlying factors, and the most important k_i factor determines the reason why the user choose a specific item.

Task Description. Given the collaborative relation-aware graph $\mathcal{G}$, our purpose is to predict the probability that user u will interact item v via learning a prediction function

$$\hat{y}_{uv} = \mathcal{F}(u, v|\mathcal{G}, \Theta), \tag{1}$$

where Θ denotes the parameters of prediction function $\mathcal{F}$.

4 METHOD

In this section, we demonstrate the detail of our proposed method DCF which consists of two main components: 1) implicit relations modeling, which first decomposes users and items embeddings into multiple fine-grained factor-level representations, then employs factor-level attention and attentive relation-wise aggregation to model fine-grained implicit factors behind the interactions; 2) explicit relations modeling, which employ user-specific relation aggregator to model fine-grained explicit factors.

4.1 User-Item Implicit Relation Modeling

Recap the widely used relation-aware aggregator [31] which considers relation type when aggregating node's local neighborhood. Given embedding $\mathbf{e}_u$ of user u and embedding of nodes in its relation-ware neighborhood $\{\mathbf{e}_i|\forall i \in \mathcal{N}_u^r\}$, here $\mathcal{N}_u^r$ are neighbor nodes of u. Then relation-aware aggregator for relation r is calculated by $\sigma(\mathbf{W}_r(\mathbf{e}_u + \sum_{i \in \mathcal{N}_u^r} \alpha_{i,r}\mathbf{e}_i))$, where σ is a nonlinear activation function, $\mathbf{W}_r$ is the transformation weight, and $\alpha_{i,r} = \frac{1}{|\mathcal{N}_u^r|}$.Relation-aware aggregator is simple but effective on learning representations for nodes in HIN [31]. However, it heavily depends on the preset meta-paths to capture the high-order connectivity, which is not scalable in practical usage, and it's unable to handle fine-grained implicit relations behind the interactions between users and items. To model more fine-grained relationships behind *Interaction*, we design a Decomposed Graph Convolutional Network (DGCN), as shown in Figure 2.

4.1.1 *Implicit Factors Decomposition.*

Given a specific user u and his interacted items set $\mathcal{N}_u = \{i|y_{ui} = 1\}$, we can get their initial embedding $\mathbf{e}_u \in \mathbb{R}^d$ and $\{\mathbf{e}_i \in \mathbb{R}^d|i \in \mathcal{N}_u\}$ via simple embedding look-up operations. Supposed that there are K implicit relations behind the interaction, we can obtain user fine-grained representation over k-th relation through projecting $\mathbf{e}_u$ into corresponding relation space:

$$\mathbf{z}_{u,k} = \sigma\left(\mathbf{W}_k^\top \mathbf{e}_u + \mathbf{b}_k\right), k = 1, 2, .., K, \tag{2}$$

where $\mathbf{W}_k \in \mathbb{R}^{d \times d}$ is transformation weight, $\mathbf{b}_k \in \mathbb{R}^d$ is bias term, $\sigma(\cdot)$ is nonlinear activation function. Obviously, $\mathbf{z}_{u,k}$ is also the factor-level representation of user u over implicit factor k, which motivates user's decisions when choosing items. Analogously, we can also get k-th fine-grained representation for item i as $\mathbf{z}_{i,k}$ via similar operation:

$$\mathbf{z}_{i,k} = \sigma\left(\mathbf{W}_k^\top \mathbf{e}_i + \mathbf{b}_k\right), k = 1, 2, .., K. \tag{3}$$

Note that parameters $\mathbf{W}_k$ and $\mathbf{b}_k$ are shared for the user and item in same relation space.

4.1.2 *Factor-level attention network.*

After constructing fine-grained implicit representations over different factors for both the user and interacted items, we aim to aggregate the items that are interacted due to the same implicit relation. We can assume that the reason behind the interaction between user u and item i is that the item implicit representation pertinent to factor k "attracts" the user. It is obvious that different items may be interacted due to different factors. To this end, we calculate a relation distribution vector $\mathbf{p}_i^u = [p_{i,1}^u, p_{i,2}^u, p_{i,3}^u, ..., p_{i,K}^u]$ for each interacted item, each element of which is calculated by:

$$\tilde{p}_{i,k}^u = ReLU(\mathbf{W}_p^\top [\mathbf{z}_{i,k}, \mathbf{z}_{u,k}] + b_p), \tag{4}$$

$$p_{i,k}^u = \frac{\exp(\tilde{p}_{i,k}^u)}{\sum_{k'=1}^K \exp(\tilde{p}_{i,k'}^u)}, \forall i \in \mathcal{N}_u, \tag{5}$$

where $[\cdot, \cdot]$ denotes the concatenate operation, $\mathbf{W}_p$ and b_p are transformation weight matrix and bias term. $p_{i,k}^u$ is the probability that factor k is the reason why user u interacts item i, which only depends on the correlation between their factor-level representations. Obviously, the larger $p_{i,k}^u$ is, the more likely we choose item i to construct $\mathbf{o}_k^u$ which is the representation of implicit factor k for user u. Then we obtain $\mathbf{o}_k^u$ by aggregating the neighborhood representations related to factor k:

$$\mathbf{o}_k^u = ReLU(\mathbf{z}_{u,k} + \sum_{i \in \mathcal{N}_u} p_{i,k}^u \mathbf{z}_{i,k}), k = 1, 2, .., K \tag{6}$$

In order to balance the influence of different neighbor numbers and make the computation more efficient, $\mathcal{N}_u$ is constructed by sampling a fixed number of neighbors.

[1] CRG is treated undirected.

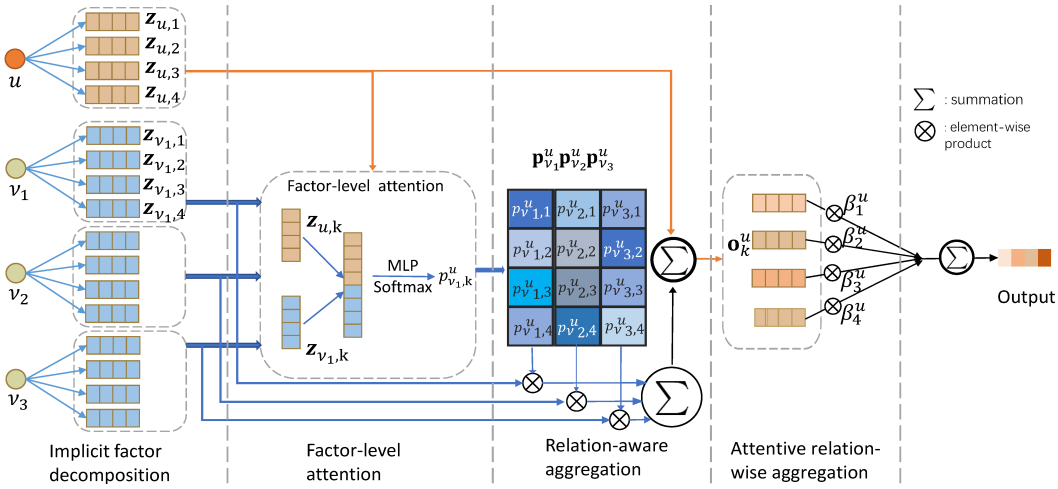

Figure 2: Illustration of the Decomposed Graph Convolutional Network (DGCN), taking user node as example ($K = 4$). It takes user embedding and embeddings of interacted items as input, and output updated user embedding.

It's technically difficult to discretely separate the interacted items according to different implicit factors. So we transform such problem to learn the probability distribution over each factor for each interacted item, and Section 5.5 bears out the feasibility of this method.

4.1.3 ***Attentive implicit relation-wise aggregation.*** We need to fuse representations of implicit factors to formulate updated user representation, and a straightforward solution is to apply average pooling operation on them. Considering different implicit factors have different importance for users while choosing items, we utilize the attention mechanism to learn the importance of each implicit relation, which is defined as follows:

$$\tilde{\beta}_k^u = \mathbf{ReLU}(\mathbf{W}\mathbf{o}_k^u + b), \tag{7}$$

where $\mathbf{W}$ and b are transform weight matrix and bias term. Note that $\mathbf{W}$ and b are shared parameters among the all implicit relations. Then, the importance of each implicit relation is calculated by normalizing the attention value through the softmax function:

$$\beta_k^u = \frac{exp(\tilde{\beta}_k^u)}{\sum_{k' \in \mathcal{R}^-} exp(\tilde{\beta}_{k'}^u)}. \tag{8}$$

We combine all representations of implicit factors with attention weight to form the updated user representation, as follows:

$$\mathbf{e}_u' = \sigma(\sum_{k \in \mathcal{R}^-} \beta_k^u \mathbf{o}_k^u). \tag{9}$$

4.1.4 ***Disagreement regularization.*** To make the implicit factor distinct with each other and contain more diverse semantic information, we introduce disagreement regularization to guide the training of related fine-grained representations. We first utilize cosine distance [14] which is defined as the cosine similarity (i.e, $cos(\cdot)$) on pairwise implicit factors. Our training objective is to diminish the cosine distance among each pairwise implicit factors for all users and items, then the regularization term is defined as:

$$\mathcal{L}_{dr\_cos} = \sum_{x \in \mathcal{U} \cup \mathcal{V}} \sum_{k_1=1}^{K} \sum_{k_2=k_1+1}^{K} \frac{\mathbf{o}_{k_1}^x \odot \mathbf{o}_{k_2}^x}{||\mathbf{o}_{k_1}^x|| \cdot ||\mathbf{o}_{k_2}^x||}, \tag{10}$$

where $\odot$ denotes inner-product operation.

In fact, we can also keep the separated $\mathbf{o}_k$ to be orthogonal with each other (i.e., $\mathbf{o}_{k_1} \odot \mathbf{o}_{k_2} \approx 0$, where $1 \le k_1, k_2 \le K$ and $k_1 \ne k_2$) to achieve our goals. As a result, we define such strategy as the l_2 norm of the inner-product between pairwise $\mathbf{o}_k$:

$$\mathcal{L}_{dr\_inner} = \sum_{x \in \mathcal{U} \cup \mathcal{V}} \sum_{k_1=1}^{K} \sum_{k_2=k_1+1}^{K} \left\| \mathbf{o}_{k_1}^x \odot \mathbf{o}_{k_2}^x \right\|_2. \tag{11}$$

Moreover, the time complexity of calculating $\mathcal{L}_{dr\_inner}$ can be reduced from $O(K^2)$ to $O(K)$, as follows:

$$\mathcal{L}_{dr\_inner} = \frac{1}{2} \sum_{x \in \mathcal{U} \cup \mathcal{V}} \left\| \left(\sum_{k=1}^{K} \mathbf{o}_k^x \right)^2 - \sum_{k=1}^{K} (\mathbf{o}_k^x)^2 \right\|_2. \tag{12}$$

Therefore, $\mathcal{L}_{dr\_inner}$ is chosen as the disagreement regularization in our paper due to its linear computational complexity.

4.2 Explicit Relation Modeling

Despite the implicit factors revealed in the collaborative signals which can be modeled through DGCN, the explicit factors can be captured via modeling explicit relations. Knowledge Graph Embedding (KGE) is an effective approach to model item explicit representation from relations between entities. The most representative method is TransE [1] which learns the embeddings of entities and relations via assuming $\mathbf{e}_h + \mathbf{e}_r \approx \mathbf{e}_t$ when (h, r, t) holds. $\mathbf{e}_h$, $\mathbf{e}_r$ and $\mathbf{e}_t$ are corresponding embeddings. However, existing methods [23, 30] rely on constructing additional loss based on this formula, which is more suitable for KG completion task rather than recommendation. Taking the inspiration from recent works [18, 20], we devise a user-specific relation aggregator to model explicit relations. Given a head entity h (corresponding to an item v), we sample a set of tail-relation

pairs $\mathcal{N}_h^+ = \{(r,t)|(h,r,t) \in \mathcal{G} \wedge r \in \mathcal{R}^+\}$ and a set of implicit relation neighbors $\mathcal{N}_h^- = \{t|(h,r,t) \in \mathcal{G} \wedge r \in \{Interaction\}\}$. We use the user embedding as a context input to decide the importance of this tail-relation pair:

$$\tilde{\pi}(u,r,t) = \mathbf{e}_u^\top(\mathbf{e}_r + \mathbf{e}_t). \tag{13}$$

这一段表述的意思是这个权重，要同时考虑头实体、尾实体、关系；不能只考虑头实体和关系

Here, the embedding of tail (i.e., $\mathbf{e}_t$) serves as a fine-tuning part for the relation r because relying only on modeling relation r can not completely reflect explicit factors. For example, a user may be interested in a movie that shares the same actor with some movies in his historical records. However, the film has numerous actors, while the user prefers only one of them. In that case, the relation is "is acted by" and the tail is the specific actor. Then, we can get user-specific attention scores through a $softmax$ operation:

$$\pi(u,r,t) = \frac{\exp(\tilde{\pi}(u,r,t))}{\sum_{(r',t')\in\mathcal{N}_h^+}\exp(\tilde{\pi}(u,r',t'))}. \tag{14}$$

To encode the connectivity structure in the CRG for the head h (item), we perform weighted sum among its explicit relation neighbors:

$$\mathbf{e}_{\mathcal{N}_h^+} = \sum_{(r',t')\in\mathcal{N}_h^+} \pi(u,r',t')\mathbf{e}_{t'}. \tag{15}$$

We combine explicit relation modeling part and implicit relation modeling part to denote item representation, which can be formulated as:

$$\mathbf{e}'_v = \sigma(\mathbf{W}_{agg}(\mathbf{e}_h + \mathbf{e}_{\mathcal{N}_h^+}) + f(\mathbf{e}_h, \{\mathbf{e}_t|t \in \mathcal{N}_h^-\})), \tag{16}$$

where $\mathbf{W}_{agg}$ is transformation weight, σ is the nonlinear activation function, and $f(\cdot)$ is DGCN layer. Similar to $\mathcal{N}_h^-$, $\mathcal{N}_h^+$ is also constructed by sampling a fixed number of neighbors.

4.3 High-order Connectivity Modeling

In practice, mining information beyond local neighborhoods may be desirable to capture more influential signals. For example, $u_1 \rightarrow v_1 \rightarrow u_2$ may reflect the similar fine-grained implicit relation between (u_1, v_1) and (u_2, v_1); $v_1 \xrightarrow{r_1} t_1 \xrightarrow{r_1} v_2$ suggests the similar representations for item v_1 and item v_2. To this end, we unify fine-grained implicit relation aggregation and explicit relation aggregation in a generalized form to capture high-order connectivity for node h in CRG, and the l-th aggregation is formulated as:

$$\mathbf{e}_h^{(l)} = \sigma(\sum_{k\in\mathcal{R}^+\cup\mathcal{R}^-} \gamma_k^{(l)}(\sum_{j\in\mathcal{N}_h^k} \alpha_{j,k}^{(l)}\mathbf{W}_k^{(l)}\mathbf{e}_j^{(l-1)} + \mathbf{W}_k^{(l)}\mathbf{e}_h^{(l-1)})), \tag{17}$$

where $\mathbf{e}_h^{(l-1)}$ is the output of previous propagation layer for node h, and $\mathbf{e}_h^{(0)}$ is the initial embedding. For $k \in \mathcal{R}^-$, $\mathcal{N}_h^k$ is the set of implicit relation neighbors, $\alpha_{j,k}^{(l)}$ and $\gamma_k^{(l)}$ are calculated by Eq.(4) ~ Eq.(5) and Eq.(7) ~ Eq.(8), respectively. For $k \in \mathcal{R}^+$, $\mathcal{N}_h^k$ is the set of explicit relation neighbors and $\alpha_{j,k}^{(l)}$ is obtained by Eq.(13) ~ Eq.(14). We set $\gamma_k^{(l)} = 1$ and $\mathbf{W}_k^{(l)} = \mathbf{W}_{agg}^{(l)}$ for $\forall k \in \mathcal{R}^+$.

4.4 Model learning

After stacking L layers for capturing high-order connectivity for users and items, we can obtain representations for user u in different layers, namely $\{\mathbf{e}_u^{(0)}, \mathbf{e}_u^{(1)}, ..., \mathbf{e}_u^{(L)}\}$; analogous to item v, $\{\mathbf{e}_v^{(0)}, \mathbf{e}_v^{(1)}, ..., \mathbf{e}_v^{(L)}\}$ are obtained. To make full use of connectivity information of different orders, we adopt summation operation among each layer of outputs of user u and item v to represent their embeddings:

$$\mathbf{u} = \mathbf{e}_u^{(0)} + \mathbf{e}_u^{(1)} + ... + \mathbf{e}_u^{(L)}, \mathbf{v} = \mathbf{e}_v^{(0)} + \mathbf{e}_v^{(1)} + ... + \mathbf{e}_v^{(L)}, \tag{18}$$

After constructing the representations of user and item, we use inner-product between them to predict the probability:

$$\hat{y}_{uv} = \sigma(\mathbf{u}^\mathrm{T}\mathbf{v}) \tag{19}$$

Then, the complete objective function of our proposed method is defined as follows:

$$\begin{aligned}\mathcal{L} =& \mathcal{L}_{rs} + \lambda_1\mathcal{L}_{dr\_inner} + \lambda_2\mathcal{L}_{reg} \\ =& \sum_{u\in\mathcal{U},v\in\mathcal{V}} \mathcal{J}(\hat{y}_{uv}, y_{uv}) \\ &+ \frac{\lambda_1}{2}\sum_{x\in\mathcal{U}\cup\mathcal{V}} \left\| \left(\sum_{k=1}^{K}\mathbf{o}_k^x\right)^2 - \sum_{k=1}^{K}(\mathbf{o}_k^x)^2 \right\|_2 + \lambda_2\|\Theta\|_2\end{aligned} \tag{20}$$

where $\mathcal{J}$ is the cross-entropy loss function, Θ is the total learning space which contains all embeddings and network parameters. λ_1 and λ_2 are hyper-parameters that control the importance of disagreement regularization and l_2 regularization, respectively. DCF is optimized by mini-batch Adam [11] optimizer, which is able to adaptively control the learning rate.

5 EXPERIMENTS

In this section, we present the details of the experiment setups and the corresponding results on three datasets. We start with four research questions (RQ) to lead the experiments:

RQ 1: Does our proposed method outperforms other state-of-the-art methods on two recommendation scenarios (top-N recommendation and CTR prediction)?

RQ 2: How do some hyper-parameters affect model performance, such as the number of implicit relations, the neighbor sampling sizes and high-order connectivity modeling?

RQ 3: How do different components affect our proposed method, such as fine-grained implicit relations modeling, disagreement regularization and user-specific relation aggregator?

RQ 4: Can DCF provide qualitative analyses of learned representations with regard to fine-grained implicit factors and explainable recommendations for users?

5.1 Experimental Setting

5.1.1 Datasets. We utilize the following three datasets in our experiments to verify DCF's effectiveness:

- MovieLens-20M[2] is a widely used benchmark dataset in movie recommendations, which contains approximately 20 million explicit ratings from 138000 users on 27000 movies. The rating ranges from 1 to 5.
- Book-Crossing [3] contains 1,149,780 ratings from 278,858 users in Book-Crossing community.

[2]https://grouplens.org/datasets/movielens/
[3]http://www.informatik.uni-freiburg.de/~cziegler/BX/

Table 1: Statistics and hyper-parameter settings for the three datasets (d : dimension of embeddings).

	Movie	Book	Music
# users	138159	17860	1872
# items	16954	14967	3846
# interactions	13501622	139746	42346
# entities	102569	77903	9366
# explicit relations	32	25	60
# KG triples	499474	151500	15518
d	32	64	64
batch size	16384	256	256
learning rate	10^{-3}	5×10^{-4}	10^{-3}
λ_1	5×10^{-6}	2×10^{-5}	10^{-4}
λ_2	10^{-6}	2×10^{-5}	10^{-4}

- Last.FM[4] is a widely used benchmark in music recommendations, which contains 1892 users' artists listening information from Last.fm online website.

Since three datasets are originally in rating format, we follow the procedure of [22] to convert explicit ratings to binary feedback (positive and negative). For each user, we randomly sample same amount of negative samples as their positive samples from unobserved interactions. We also filter low-frequency users and items. Similar to [21], we use Microsoft Satori[5] to build the KG for each dataset. We first find the triples with relation name contains "item" from the KG and filter out the triples whose confidence level is lower than 0.9. We collect IDs of all valid movies/books/musicians from sub-KG by matching their names with tail of triples (*head, film.film.name, tail*), (*head, book.book.title, tail*) or (*head, type.object.name, tail*). Then, we select all triples whose head entities match with the item IDs from sub-KG. The detailed statistics of the three datasets are shown in Table 1.

5.1.2 Evaluation Protocols. In this paper, we conduct experiments in two recommendation tasks. For top-N recommendation task, we adopt two widely-used evaluation protocols to evaluate the effectiveness of our proposed method: Recall@N and NDCG@N, and we set N={5, 10, 20}. For CTR prediction task, we use AUC for evaluation protocol. For each dataset, the ratio of training, validation, and test set is $6:2:2$. We report the average performance of five runs.

5.1.3 Baselines. To illustrate the effectiveness of our model, we choose one traditional CF model, three KG based models and three GCN based models, as follows:

- **MF** [13] is the standard matrix factorization which uses inner product to model user-item interactions.
- **CKE** [33] incorporates structural, textual, and visual knowledge to enhance item embeddings. In this paper, we combine CF with a structural knowledge module for original method since the unstructured data is unavailable in our datasets.
- **RippleNet** [21] models user representation as plenty of entities related to user's historical interacted items.
- **KGCN** [22] is a state-of-the-art KG-based method for recommendation, which effectively captures user-specific preferences for items in the KG.
- **PinSage** [32] is designed to apply GraphSAGE on item-item graph. In our work, we employ it on user-item bipartite graph for generating item recommendation lists.
- **NGCF** [24] is a state-of-the-art GCN-based method, which refines user and item representations via high-order connectivity modeling.
- **DisenGCN** [15] is a state-of-the-art GCN-based method, which utilizes neighborhood routing to learn fine-grained latent factors behind the edges.

5.1.4 Experiments Setup. In our work, we use *tanh* as the activation function. Hyper-parameters setting on three datasets for DCF are shown in Table 1. To fairly compare the performance of all models, we train them by optimizing cross-entropy loss with Adam optimizer and Xavier initializer [5]. We train all models for 30 epochs, and early stopping strategy is performed to prevent overfitting. The embedding size, batch size and learning rate for baselines on each dataset are the same as DCF, as shown in Table 1. We apply grid search for hyper-parameters tuning: the learning rate is tuned amongst $\{2 \times 10^{-5}, 5 \times 10^{-5}, 2 \times 10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$, the disagreement regularization weight λ_1 and l_2 normalization weight λ_2 are searched in $\{10^{-7}, 10^{-6}, 10^{-5}, 2 \times 10^{-5}, 10^{-4}, 10^{-3}\}$. For RippleNet, we set the number of hops and the memory size as 2 and 8, respectively. Other hyper-parameters are consistent with those mentioned in the original paper or their source codes.

5.2 Performance Comparison (RQ 1)

From the results of top-N recommendation and CTR prediction in Tables 2, 3 and 4, we have the following observations:

- DCF consistently yields the best performance on all datasets for top-N recommendation. In particular, DCF improves over the best baselines with regard to recall@20 by 6.8%, 19.57% and 4.1% in MovieLens-20M, Book-Crossing and Last.FM, respectively. Compared to state-of-the-art KG-based method KGCN, DCF still achieves better performance. It exactly shows that relying on external information is unable to characterize implicit fine-grained preference because even KGs that contain rich information about items can only represent an item from a limited perspective. Compared with GCN-based baselines, DCF still achieves significant improvement. We argue that this is because the fine-grained implicit relations modeling helps DCF to better capture user preference.
- DCF achieves impressive improvement upon DisenGCN, demonstrating that DCF is more effective to model implicit factors in fine-grained level.
- CKE is inferior to RippleNet and KGCN, indicating the embedding-based methods might not make full use of item knowledge and loosely couple with CF framework. Moreover, KGCN achieves the performance next to DCF for top-N recommendation in Book-Crossing dataset. It indicates that KGCN guarantees better performance even in sparse scenarios.

[4] https://grouplens.org/datasets/hetrec-2011/
[5] https://searchengineland.com/library/bing/bing-satori

Table 2: The results of Recall@N in top-N recommendation. Boldface denotes best baseline, and * denotes the significance p-value < 0.01 compared with the best baseline.

Methods	MovieLens-20M			Book-Crossing			Last.FM		
	Recall@5	Recall@10	Recall@20	Recall@5	Recall@10	Recall@20	Recall@5	Recall@10	Recall@20
MF	0.0879	0.1433	0.1843	0.0442	0.0568	0.0736	0.0718	0.1236	0.1766
CKE	0.0764	0.1303	0.1721	0.0452	0.0575	0.0733	0.0768	0.1346	0.2034
RippleNet	0.0878	0.1324	0.1798	0.0546	0.0711	0.0837	0.0732	0.1199	0.1501
KGCN	**0.0901**	0.1425	0.1995	**0.0631**	**0.0792**	0.0847	0.0735	0.1289	0.1826
PinSage	0.0893	**0.1492**	0.2124	0.0549	0.0751	0.0780	0.0816	0.1453	0.2113
NGCF	0.0842	0.1439	**0.2149**	0.0588	0.0734	0.0837	**0.0896**	**0.1459**	**0.2142**
DisenGCN	0.0832	0.1389	0.2116	0.0572	0.0704	**0.0874**	0.0893	0.1419	0.2114
DCF	**0.0927***	**0.1572***	**0.2295***	**0.0745***	**0.0927***	**0.1018***	**0.0975***	**0.1536***	**0.2230***
% improve.	2.89%	5.36%	6.79%	18.07%	17.05%	19.57%	8.81%	5.28%	4.11%

Table 3: The results of NDCG@N in top-N recommendation. Boldface denotes best baseline, and * denotes the significance p-value < 0.01 compared with the best baseline.

Methods	MovieLens-20M			Book-Crossing			Last.FM		
	NDCG@5	NDCG@10	NDCG@20	NDCG@5	NDCG@10	NDCG@20	NDCG@5	NDCG@10	NDCG@20
MF	0.0895	0.1179	0.1279	0.0375	0.0465	0.0579	0.0602	0.0763	0.0807
CKE	0.0812	0.1016	0.1195	0.0402	0.0447	0.0494	0.0612	0.0801	0.0977
RippleNet	0.0905	0.1115	0.1212	0.0478	0.0535	0.0578	0.0621	0.0798	0.0819
KGCN	0.0936	0.1126	0.1373	**0.0584**	**0.0627**	**0.0654**	0.0619	0.0832	0.1013
PinSage	**0.0917**	**0.1310**	**0.1586**	0.0516	0.0587	0.0599	0.0688	0.0951	0.1103
NGCF	0.0904	0.1249	0.1568	0.0535	0.0592	0.0616	0.0711	**0.0991**	**0.1232**
DisenGCN	0.0786	0.1096	0.1475	0.0453	0.0499	0.0642	**0.0759**	0.0947	0.1198
DCF	**0.0992***	**0.1341***	**0.1690***	**0.0682***	**0.0749***	**0.0790***	**0.0789***	**0.1028***	**0.1289***
% improve.	8.18%	2.37%	6.56%	16.78%	19.46%	20.80%	10.97%	3.73%	4.62%

Table 4: The results of AUC in CTR prediction.

	MovieLens-20M	Book-Crossing	Last.FM
MF	0.961(-2.1%)	0.692(-6.7%)	0.809(-2.5%)
CKE	0.925(-6.1%)	0.678(-8.8%)	0.802(-3.4%)
RippleNet	0.972(-0.9%)	0.715(-3.2%)	0.787(-5.3%)
KGCN	0.978(-0.3%)	0.735(-0.4%)	0.798(-3.9%)
PinSage	0.980(-0.1%)	0.719(-2.6%)	0.808(-2.6%)
NGCF	0.979(-0.2%)	0.717(-2.9%)	0.817(-1.5%)
DisenGCN	0.980(-0.1%)	0.717(-2.9%)	0.816(-1.6%)
DCF	**0.981**	**0.738**	**0.829**

Table 5: Recall@20 results of different number of implicit relations for three datasets.

K	1	2	3	4	5	6
Movie	0.1985	0.2063	0.2245	**0.2295**	0.2146	0.2132
Book	0.0896	0.1031	0.0977	**0.1074**	0.1024	0.1018
Music	0.201	0.2036	0.2098	**0.2210**	0.2079	0.2068

- PinSage and NGCF achieve satisfactory performance for two recommendation scenarios, demonstrating the effectiveness of GCN-based methods on capturing high-order connectivity to enhance the embeddings of users and items. Moreover, NGCF outperforms PinSage in most cases, which points the usefulness of the explicitly preserving collaborative signals.

Table 6: Impact of high-order propagation. R is short for Recall, and NG is short for NDCG.

	Movie		Book		Music	
	R@20	NG@20	R@20	NG@20	R@20	NG@20
DCF-1	0.2187	0.1644	**0.1018**	**0.0790**	**0.2230**	**0.1289**
DCF-2	**0.2295**	**0.1690**	0.0942	0.0701	0.2176	0.1185
DCF-3	0.2104	0.1602	0.0876	0.0679	0.1892	0.0965

5.3 Hyper-parameters Study (RQ 2)

5.3.1 ***Impact of sampling size of two types of relation neighbors.*** The sampling number of explicit relation neighbors and implicit relation neighbors should be different to verify the influence of two factors separately. We vary the size of implicit relation neighbors to investigate the efficacy of the usage of implicit relationships between users and items by fixing the size of explicit relation neighbors. Analogously, we can evaluate the efficacy of the usage of explicit relations. n_1 denotes sampling size of explicit relation neighbors, and n_2 denotes sampling size of implicit relation neighbors. Results in Figure 3 show that DCF achieves best performance when $n_1 = 20$ and $n_2 = 20$ for Last.FM, $n_1 = 10$ and $n_2 = 20$ for Book-Crossing. In addition, we find DCF achieves better performance in MovieLens-20M when the neighbor sampling sizes increase. However, we choose $n_1 = 10$ and $n_2 = 20$ for MovieLens-20M for trade-off between performance and computational cost.

5.3.2 ***Impact of number of fine-grained implicit relations.*** We vary K from 1 to 6 to investigate the influence of the number of implicit relations. From the results in Table 5, we find the performance of DCF gets better when K increases. However, the

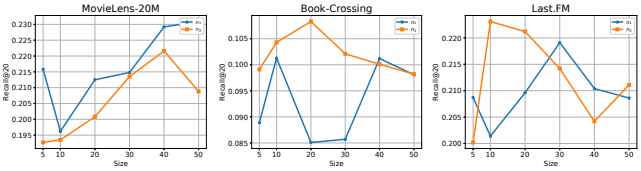

Figure 3: Recall@20 results of different sizes of the set of relation neighbors.

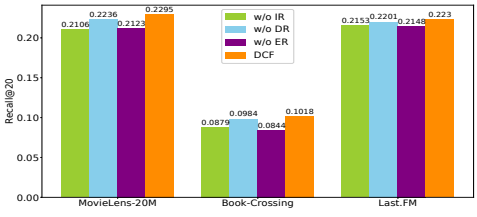

Figure 4: Performance comparison of three different variants of DCF.

performance of DCF tends to become saturated as K continues to grow, possibly due to overfitting problem. We make assumption that DCF achieves the optimal performance when the preset K is closed to the number of implicit factors that actually affect user decision-making. Results in Table 5 demonstrate that $K = 4$ is the optimal value on three datasets.

5.3.3 ***Impact of high-order connectivity modeling.*** We investigate the impact of high-order propagation via varying model depth L in $\{1, 2, 3\}$, getting corresponding variants of DCF: DCF-1, DCF-2 and DCF-3. From the experimental results in Table 6, we can find the performance of DCF decays when the depth of the model is saturated, demonstrating too deep of DCF may bring noise. Therefore, we set $L = 2$ for MovieLens-20M and $L = 1$ for Book-Crossing and Last.FM.

5.4 Ablation Study (RQ 3)

To verify the effectiveness of the specific designs in our model, we do ablation study via generating three different variants of DCF. In particular, we generate $\text{DCF}_{w/o\ IR}$ by setting the number of implicit relations to 1 (i.e., $K = 1$) which denotes that we don't model fine-grained implicit relations between users and items. We disable disagreement regularization, termed as $\text{DCF}_{w/o\ DR}$. Moreover, we disable the explicit relation modeling part and employee DGCN to model fine-grained implicit relations, namely $\text{DCF}_{w/o\ ER}$. From the experimental results in Figure 4, we have following findings:

- From the comparison between $\text{DCF}_{w/o\ IR}$ and DCF, we can find that fine-grained modeling of implicit factors is necessary to better capture user preference. As illustrated in Table 2, $\text{DCF}_{w/o\ IR}$ achieves massive improvement upon KG based recommendation methods (e.g. CKE, RippleNet, KGCN), which demonstrates the effectiveness of user-specific relation aggregation of DCF in modeling explicit relations.

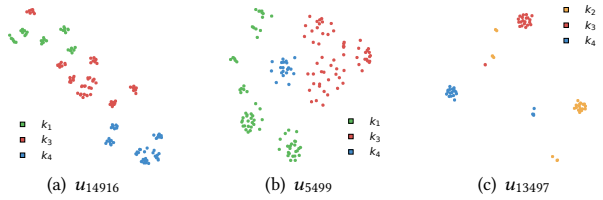

Figure 5: Interacted item embedding visualization using t-SNE of three users on Book-Crossing dataset. Items that are interacted by the user due to the same fine-grained implicit factor k_i tend to cluster together.

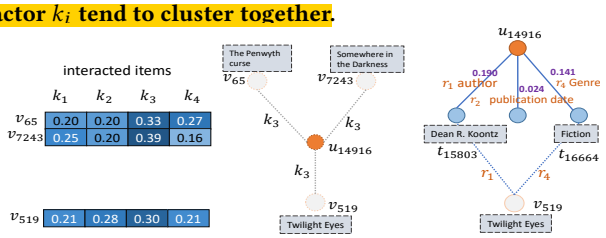

Figure 6: Real example from Book-Crossing dataset. Left figure is the factor-level attention weight of interacted items and recommended item, center figure reveals the relationship from implicit perspective, and right figure indicates their association from explicit perspective.

- From the comparison between $\text{DCF}_{w/o\ DR}$ and DCF, we can see that the proposed disagreement regularization successfully drives the model to learn more distinct fine-grained implicit relations, resulting in performance improvement of DCF.
- Compared with $\text{DCF}_{w/o\ ER}$, DCF achieves better performance, demonstrating the effectiveness and necessity of our method on modeling explicit relations. In addition, $\text{DCF}_{w/o\ ER}$ is superior to DisenGCN in modeling fine-grained implicit factors behind interaction, as illustrated in Table 2 and Figure 4.

5.5 Case Study (RQ 4)

5.5.1 ***In-depth analyses of the learned embeddings.*** In order to analyze the learned representations in depth, we randomly select three users in Book-Crossing dataset and project the learned embeddings of interacted items into 2-D space with t-SNE algorithm [16]. From the distribution of items in Figure 5, we can observe that items that are interacted by the user due to the same fine-grained implicit factor tend to cluster with each other. To conclude, the visualization shows DCF can effectively distinguish the fine-grained implicit factors behind the interactions between users and items, leading to finer reflecting deep motivations behind user behaviors.

5.5.2 ***Analysis of user behaviors.*** We analyze the user behaviors through the visualization of some components in our model, as illustrated in Figure 6. We randomly select a user u_{14916} in the Book-Crossing dataset. The recommended book for this user is *Twilight Eyes* From implicit factors, the user may be interested in *Twilight Eyes* due to factor k_3 and there exist two interacted items

in the user's historical records, which are *The Penwyth Curse* and *Somewhere in the Darkness*. They are also interacted by user u_{14916} through the same factor. To this end, we can make an assumption that the factor k_3 in that case may be related with horror novel. For explicit factors, the user pays the most attention to "genre" (0.190) and "author" (0.141). Hence, from explicit perspective, the explanation for the recommendation can be that its author (Dean R. Koontz) and genre (fiction) are what the user prefers.

6 CONCLUSION

In this work, we take the user-item bipartite graph and knowledge graph as a unified CRG and propose a generalized relation aggregator based method DCF to unveil the fine-grained factors behind the interactions for enhancing users and items embeddings on CRG. To capture implicit factors, we utilize factor-level attention and attentive implicit relation aggregation to model collaborative signals in fine-grained level. Then we apply disagreement regularization to avoid the issue that the separated factors reflect similar motivations. To capture explicit factors, we devise a user-specific relation aggregator to aggregate the most import entities. We stack multiple layers to capture high-order connectivity to enhance users and items representation. We conduct extensive experiments on three datasets for two recommendation tasks, and results demonstrate the effectiveness and interpretability of our method.

In future work, we want to extend DCF for enhancing user preferences via leveraging social networks [27]. Another promising direction of future work is to explore a more effective sampling policy [26, 29] to replace uniformly sampling rule.

ACKNOWLEDGMENT

This work was supported by CERNET Innovation Project (NGII20190904).

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*. 2787–2795.

[2] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *WWW*. 151–161.

[3] 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 & Data Mining*. 1235–1243.

[4] Asmaa Elbadrawy and George Karypis. 2015. User-specific feature-based similarity models for top-n recommendation of new items. *ACM Transactions on Intelligent Systems and Technology (TIST)* 6, 3 (2015), 1–20.

[5] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010*. 249–256.

[6] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*. 1024–1034.

[7] 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 Trans. Knowl. Data Eng.* 30, 12 (2018), 2354–2366.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. 173–182.

[9] 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, KDD 2018*. 1531–1540.

[10] Santosh Kabbur, Xia Ning, and George Karypis. 2013. FISM: factored item similarity models for top-N recommender systems. In *KDD*. 659–667.

[11] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[12] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[13] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer* 42, 8 (2009), 30–37.

[14] Jian Li, Zhaopeng Tu, Baosong Yang, Michael R. Lyu, and Tong Zhang. 2018. Multi-Head Attention with Disagreement Regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018*. 2897–2903.

[15] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled Graph Convolutional Networks. In *ICML*. 4212–4221.

[16] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[17] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10*. 285–295.

[18] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web - 15th International Conference, ESWC 2018*. 593–607.

[19] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. *Proc. VLDB Endow.* 4, 11 (2011), 992–1003.

[20] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[21] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems. In *CIKM*. 417–426.

[22] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2019*. 3307–3313.

[23] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *KDD*. 950–958.

[24] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. 165–174.

[25] Xiao Wang, Ruijia Wang, Chuan Shi, Guojie Song, and Qingyong Li. 2020. Multi-Component Graph Convolutional Collaborative Filtering. In *AAAI 2020*.

[26] Xiang Wang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. 2020. Reinforced Negative Sampling over Knowledge Graph for Recommendation. *CoRR* abs/2003.05753 (2020). arXiv:2003.05753

[27] Le Wu, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. 2019. A Neural Influence Diffusion Model for Social Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*. 235–244.

[28] Yuexin Wu, Hanxiao Liu, and Yiming Yang. 2018. Graph Convolutional Matrix Completion for Bipartite Edge Prediction. In *IC3K*. 49–58.

[29] Yikun Xian, Zuohui Fu, S. Muthukrishnan, Gerard de Melo, and Yongfeng Zhang. 2019. Reinforcement Knowledge Graph Reasoning for Explainable Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*. 285–294.

[30] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon M. Jose. 2019. Relational Collaborative Filtering: Modeling Multiple Item Relations for Recommendation. In *SIGIR*. 125–134.

[31] Fengli Xu, Jianxun Lian, Zhenyu Han, Yong Li, Yujian Xu, and Xing Xie. 2019. Relation-Aware Graph Convolutional Networks for Agent-Initiated Social E-Commerce Recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019*. 529–538.

[32] 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 *KDD*. 974–983.

[33] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *KDD*. 353–362.

[34] Lei Zheng, Chun-Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S. Yu. 2018. Spectral collaborative filtering. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018*. 311–319.