Multiplex Memory Network for Collaborative Filtering

Xunqiang Jiang *
skd621@bupt.edu.cn

Binbin Hu †
bin.hbb@antfin.com

Yuan Fang ‡
yfang@smu.edu.sg

Chuan Shi §
shichuan@bupt.edu.cn

Abstract

Recommender systems play an important role in helping users discover items of interest from a large resource collection in various online services. Although current deep neural network-based collaborative filtering methods have achieved state-of-the-art performance in recommender systems, they still face a few major weaknesses. Most importantly, such deep methods usually focus on the direct interaction between users and items only, without explicitly modeling high-order co-occurrence contexts. Furthermore, they treat the observed data uniformly, without fine-grained differentiation of importance or relevance in the user-item interactions and high-order co-occurrence contexts. Inspired by recent progress in memory networks, we propose a novel *multiplex memory network* for collaborative filtering (MMCF). More specifically, MMCF leverages a multiplex memory layer consisting of an interaction memory and two co-occurrence context memories simultaneously, in order to jointly capture and locate important and relevant information in both user-item interactions and co-occurrence contexts. Lastly, we conduct extensive experiments on four datasets, and the results show the superior performance of our model in comparison with a suite of state-of-the-art methods.

1 Introduction

In the era of information overload, recommender systems have been playing an increasingly important role in various online services [6], including E-commerce, online news and social media. Current recommender systems evolve around learning an effective preference prediction function based on historical user-item interaction records, known as collaborative filtering (CF) [14]. In particular, matrix factorization (MF) [7] is one of the most successful methods among CF techniques, which models the preference as the inner product of user and item latent factors. Since the interactions between users and items are often complex and may involve vastly different underlying intentions, such a shallow representation could be inadequate in expressing the preferences.

Due to the ability of modeling non-linear functions, deep neural networks have yielded state-of-the-art performance in many research areas such as computer vision and natural language processing. The recent integration of deep models into recommender systems has also revealed the remarkable strength of complex non-linear transformations of user-item interactions. Existing neural network-based recommendation fall into two broad categories. The first category replaces the traditional inner product with nonlinear neural networks to model more complex prediction functions [2, 15]. Recent efforts [12, 24] further utilize random walks to augment user-item interactions, which ultimately train a more effective deep model. The second category extends the matrix factorization by incorporating deep representations learned from additional side information such as review texts and videos [22, 25].

Unfortunately, these deep recommendation models still suffer from several limitations. Consider the scenario in Fig. 1(a1)—given that the user u_3 purchased items such as camera (i_2), pen (i_3) and book (i_4) in the past, the recommender system needs to determine whether u_3 is likely to purchase a memory card (i_1).

One limitation of the previous deep models, as sketched in Fig. 1(b), is that they mainly focus on the direct interaction between users and items, without explicitly accounting for the high-order *co-occurrence contexts* between users and items, respectively. Examples are the *co-purchase* contexts between users such as u_1 and u_2 both purchasing the same item camera (i_2) in Fig. 1(a2), as well as the *co-purchased* context between items such as the memory card (i_1) and camera (i_2) both purchased by the same user u_1 in Fig. 1(a3). Such high-order contexts are especially crucial in sparse datasets lacking enough user-item interactions. While some existing works [1, 4, 24] have attempted to utilize higher-order information, they focus on enriching user-item interactions, rather than explicitly modeling user-user and

*Beijing University of Posts and Telecommunications.
†Ant Financial Services Group.
‡Singapore Management University.
§Beijing University of Posts and Telecommunications.

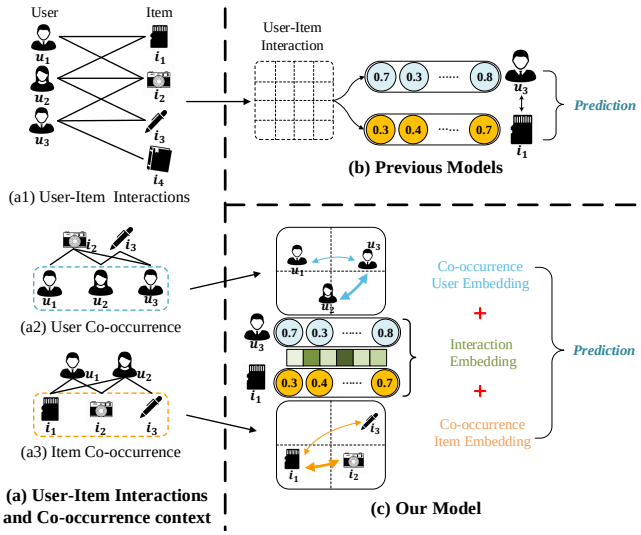


Figure 1: Illustration of our problem. (a) User-item interactions and co-occurrence contexts. (b) Previous models focusing on user-item interactions. (c) Our model with fine-grained interactions and co-occurrence contexts.

item-item contexts.

Second, most previous works treat the observed data (*i.e.,* user-item interactions and co-occurrence contexts) uniformly, without *fine-grained* differentiation of their underpinning preferences or importance. Intuitively, a user-item interaction could be influenced by diverse preferences, given users from different demographic, socioeconomic or cultural groups. Generally, such diverse preferences may be weighed differently in different interactions. While such differentiation has been explored to some extent for user-item interactions [1, 20, 27], it remains an open question for co-occurrence contexts, which are likewise not uniformly important to the recommendation. As shown in Fig. 1(c), given a memory card (i_1), the co-occurring camera (i_2) is a more important context than a pen (i_3), as memory card and camera are more complementary in nature. Similar scenarios exist in user co-occurrence contexts. Thus, in both user-item interactions and co-occurrence contexts, we need a fine-grained model that allows different preferences and contexts to receive varying attention.

To deal with the above considerations, two critical questions must be addressed. First, how to integrate user-item interactions and co-occurrence contexts in a unified and end-to-end manner? Second, how to enable fine-grained differentiation for both user-item interactions and co-occurrence contexts? Inspired by the recent progress in memory networks [11, 17], a few studies [1, 20, 21] have employed memory networks in recommender systems, which have the advantage of modeling fine-grained preferences with a memory module. However, their architecture only operates with a single type of memory centering around user-item interactions. Different from them, we firstly propose a novel **M**ultiplex **M**emory Network for **C**ollaborative **F**iltering (MMCF), which leverages *multiplex memory layers* to jointly capture user-item interactions and the co-occurrence contexts simultaneously in one framework. More specifically, the multiplex memory layer is designed to accommodate multiple types of memory concurrently, consisting of an interaction memory to model user-item interactions and two co-occurrence context memories to model user-user and item-item contexts, respectively. These memories are not independent and their interplay is also captured by our model. Furthermore, towards fine-grained interactions and contexts, for each type of memory, an attention mechanism is employed to locate the important and relevant information in the memory slots.

In summary, we make the following contributions. (1) We highlight the importance of explicitly modeling the user and item co-occurrence contexts in deep models especially when the user-item interactions are sparse, which is also observed empirically in our experiments. (2) We propose a novel model MMCF, a deep multiplex memory network to jointly capture fine-grained user-item interactions and co-occurrence contexts through multiple types of memory in an unified, end-to-end framework. To our best knowledge, MMCF is the first memory network concurrently employing multiple types of memory for collaborative filtering. (3) We conduct extensive experiments on four real-world benchmarks, validating the effectiveness of MMCF and its assumptions.

2 Related Work

Collaborative filtering (CF) aims to recommend a suitable list of items based on historical user-item interaction records. In particular, the matrix factorization (MF) method [5, 7] has shown its effectiveness in many applications, which decomposes the user-item interaction matrix to learn low-rank latent user and item factors. Furthermore, many studies [4, 10, 16] have been proposed to extend a MF-based framework to incorporate additional side information. In spite of the success of existing CF-based recommendation methods, they still suffer from the limited ability of modeling more complex user-item interactions.

Due to the ability of modeling arbitrary non-linear functions, deep neural networks endow recommender system with the potential of capturing more complex and intricate user-item interactions [2, 9, 15]. Sim-

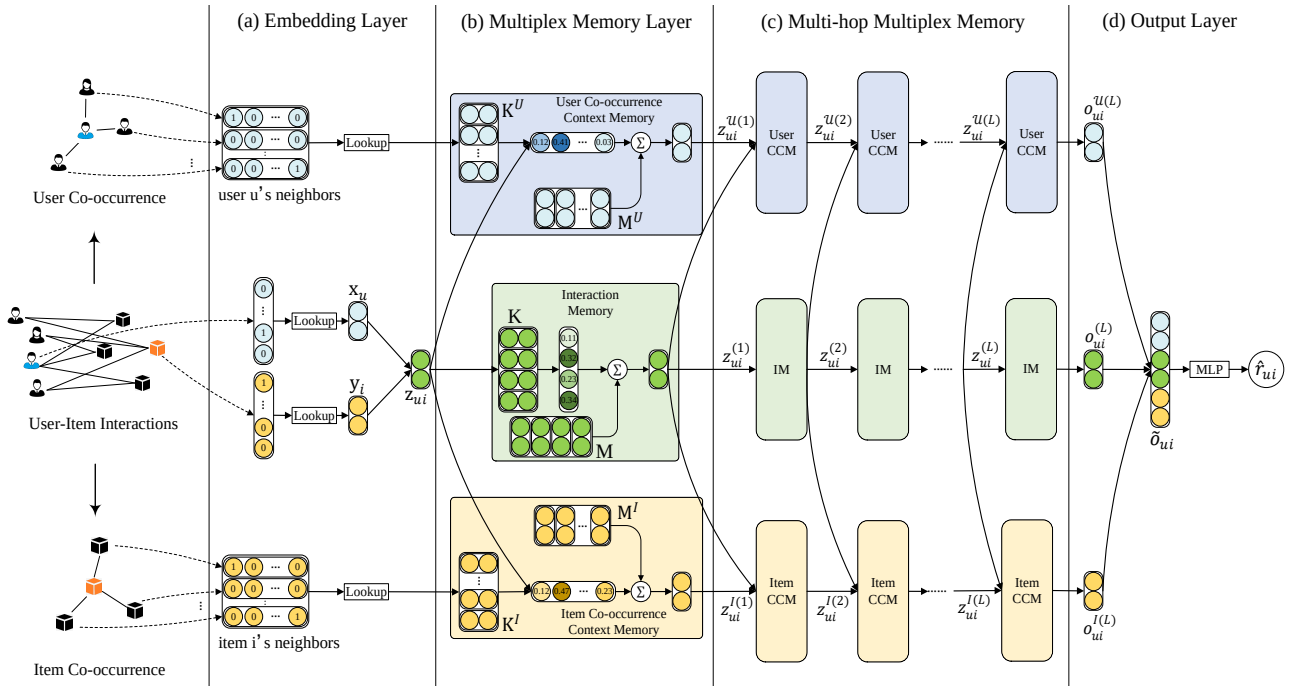


Figure 2: Overall architecture of the proposed model MMCF.

ilar to the traditional MF-based model, various attempts [4, 19] have been made to integrate different side information into the deep models. However, most of these deep models only concern with coarse-grained user-item interactions.

To enable fine-grained modeling, several studies attempt to employ memory networks to capture complex and fine-grained user-item interactions in collaborative filtering [1, 20, 21]. Additional side information such as texts [27] and heterogeneous relations [26] can be integrated into the memory network too. However, they still neglect high-order co-occurrence contexts. In contrast, our proposed model MMCF hinges on a novel multiplex memory layer, which jointly captures fine-grained user-item interactions as well as user-user and item-item co-occurrence contexts using multiple types of memory concurrently. To the best of our knowledge, MMCF is the first memory network concurrently employing multiple types of memory in a unifying, end-to-end framework for collaborative filtering.

3 The Proposed Model: MMCF

In this section, we present our model MMCF, a novel deep multiplex memory network for collaborative filtering.

3.1 Overview We assume a set of historical user-item interaction records (*e.g.,* product purchasing and movie watching), denoted as $\mathcal{R}$. $\mathcal{R}$ is comprised of $\{u, i, r_{ui} | u \in \mathcal{U}, i \in \mathcal{I}\}$, where $\mathcal{U}$ and $\mathcal{I}$ respectively represent the user and item sets, $r_{ui} \in \{0, 1\}$ is set to one if and only if there is an observed interaction between user u and item i. Formally, our goal is to learn a prediction function $\hat{r}_{ui} = \mathcal{F}(u, i; \Theta)$, such that $\hat{r}_{ui}$ represents the probability that user u will interact with (*e.g.,* purchase and watch) item i, and Θ represents the parameters of the prediction function $\mathcal{F}$.

Next, we present the overall architecture of MMCF in Fig. 2. After an initial embedding layer for users and items in Fig. 2(a), we propose a multiplex memory layer to jointly model user-item interactions and co-occurrence contexts as shown in Fig. 2(b). Different from existing memory network-based CF, we simultaneously accommodate multiple types of memory: an *interaction memory* (IM) sublayer for user-item interactions, and two *co-occurrence context memory* (CCM) sublayersfor user-user and item-item contexts. An attention mechanism is further employed for fine-grained modeling—IM weighs various underlying preferences for user-item interactions, whereas CCM locates the important and relevant users or items. Moreover, Fig. 2(c) extends the multiplex memory layer to multiple hops to model more complex interactions and co-occurrence contexts. Finally, the output layer in Fig. 2(d) makes a prediction $\hat{r}_{ui}$.

3.2 Embedding Layer Following existing works[2, 4], we transform the one-hot encoding of users and items into a low-dimensional dense vector via a lookup (*i.e.,* embedding) layer. Formally, each user-item pair $\langle u, i\rangle$ can be denoted as the one-hot representation $\mathbf{p}_u \in \mathbb{R}^{|\mathcal{U}|\times 1}$ and $\mathbf{q}_i \in \mathbb{R}^{|\mathcal{I}|\times 1}$, where $|\mathcal{U}|$ and $|\mathcal{I}|$ are the total number of users and items respectively. Thus, the lookup layer corresponds to two parameter matrices $\mathbf{P} \in \mathbb{R}^{|\mathcal{U}|\times d}$ and $\mathbf{Q} \in \mathbb{R}^{|\mathcal{I}|\times d}$, which store the d-dimensional latent factors for users and items, respectively. Thus, the embedding of user u and item i, or $\mathbf{x}_u$ and $\mathbf{y}_i$, can be looked up as follows:

$$\mathbf{x}_u = \mathbf{P}^T \cdot \mathbf{p}_u, \tag{3.1}$$

$$\mathbf{y}_i = \mathbf{Q}^T \cdot \mathbf{q}_i. \tag{3.2}$$

3.3 Multiplex Memory Layer The key difference in our multiplex memory layer is the joint modeling of multiple types of memory, including an interaction memory and two co-occurrence context memories.

3.3.1 Interaction Memory sublayer To start, given a user-item pair $\langle u, i\rangle$, we assume a joint user-item embedding $\mathbf{z}_{ui}$ as follows:

$$\mathbf{z}_{ui} = f(\mathbf{x}_u, \mathbf{y}_i), \tag{3.3}$$

where $f(\cdot)$ can be addition, element-wise product or other aggregation. Addition is chosen in our implementation, which performs the best empirically.

Since each user-item interaction is underpinned by many latent preferences, we utilize attention-based memory network to capture such fine-grained interactions. In particular, we propose the *Interaction Memory* (IM) sublayer to weigh the importance of latent preferences in each user-item interaction, as shown in Fig. 2(b). Inspired by the key-value memory network [11], given a joint user-item embedding, the IM leverages a key matrix $\mathbf{K} \in \mathbb{R}^{K\times d}$ and a memory matrix $\mathbf{M} \in \mathbb{R}^{K\times d}$, where K denotes the number of memory slots and d is the embedding dimension. To further differentiate the importance of these preferences, we learn an attention vector $\mathbf{a} \in \mathbb{R}^K$ based on the similarity between the user-item representation $\mathbf{z}_{ui}$ and the key matrix $\mathbf{K}$:

$$\mathbf{a}_m = \mathbf{z}_{ui}^T \mathbf{K}_m, \quad \forall m \in \{1, 2, \ldots, K\}, \tag{3.4}$$

where $\mathbf{K}_m$ refers to the m^{th} row of $\mathbf{K}$. Thus, the attention vector signifies the importance of different memory slots. Finally, we obtain the representation $\mathbf{o}_{ui}$ of the user-item pair $\langle u, i\rangle$ by aggregating the memory values based on the attention vector:

$$\mathbf{o}_{ui} = \sum_m \frac{\exp(\mathbf{a}_m)}{\sum_{m'=1}^K \exp(\mathbf{a}_{m'})} \mathbf{M}_m. \tag{3.5}$$

3.3.2 Co-occurrence Context Memory sublayers Beside the user-item interactions, as motivated in Sect. 1, it is crucial to consider high-order co-occurrence contexts, *i.e.,* user-user and item-item contexts, especially in datasets with sparse user-item interactions. In the following, we first introduce the construction of co-occurrence contexts, followed by the co-occurrence context memory for capturing the fine-grained influence of co-occurrence contexts.

Given a user u, his or her co-occurrence context $\mathcal{C}^{\mathcal{U}}(u)$ is defined as the set of users who have purchased at least one common item as u has. Similarly, given an item i, its co-occurrence context $\mathcal{C}^{\mathcal{I}}(i)$ is the set of items which have been purchased by at least one common user as i has. That is,

$$\mathcal{C}^{\mathcal{U}}(u) = \{u' \in \mathcal{U} \mid \exists i \in \mathcal{I} : r_{ui} = r_{u'i} = 1\}, \tag{3.6}$$

$$\mathcal{C}^{\mathcal{I}}(i) = \{i' \in \mathcal{I} \mid \exists u \in \mathcal{U} : r_{ui} = r_{ui'} = 1\}. \tag{3.7}$$

However, taking all of the user or item co-occurrences as contexts may not work well due to noises in the co-occurrences. Hence, as a common way to assess the strength of co-occurrences, we filter the co-occurrences using the pointwise mutual information (PMI) [8]:

$$PMI^{\mathcal{U}}(u, u') = \log \frac{P(u, u')}{P(u)P(u')} \approx \log \frac{\#(u, u') \cdot N^{\mathcal{U}}}{\#u \cdot \#u'}, \tag{3.8}$$

$$PMI^{\mathcal{I}}(i, i') = \log \frac{P(i, i')}{P(i)P(i')} \approx \log \frac{\#(i, i') \cdot N^{\mathcal{I}}}{\#i \cdot \#i'}. \tag{3.9}$$

Here, $\#(\triangle, \square)$ denotes the frequency of $\square$ appearing as the context of $\triangle$, $\#\triangle = \sum_{\square} \#(\triangle, \square)$ and $\#\square = \sum_{\triangle} \#(\triangle, \square)$. Moreover, $N^{\mathcal{U}}$ and $N^{\mathcal{I}}$ are the total number of co-occurrence contexts for users and items, respectively. Subsequently, we filter the original co-occurrence contexts to require their PMI to be more than $\log k$ (*i.e.,* shifted positive PMI):

$$\tilde{\mathcal{C}}^{\mathcal{U}}(u) = \{u' \in \mathcal{C}^{\mathcal{U}}(u) \mid PMI^{\mathcal{U}}(u, u') > \log k\}, \tag{3.10}$$

$$\tilde{\mathcal{C}}^{\mathcal{I}}(i) = \{i' \in \mathcal{C}^{\mathcal{I}}(i) \mid PMI^{\mathcal{I}}(i, i') > \log k\}. \tag{3.11}$$

Here, k is a hyper-parameter to control the sparsity of the co-occurrence contexts.

To model the above co-occurrence contexts, we propose the Co-occurrence Context Memory (CCM) sublayer , as shown in Fig. 2(b). In particular, there are two CCMs in a multiplex memory layer, for user-user and item-item contexts, respectively. We elaborate on the co-occurrence contexts for items first. Simlar to IM, the item CCM utilizes a key matrix $\mathbf{K}^{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}|\times d}$ and a memory matrix $\mathbf{M}^{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}|\times d}$. However, unlike

IM, the number of memory slots equals to the number of items, *i.e.,* each memory slot stands for an item, and we aim to capture the co-occurrence items for the item i involved in a given user-item interaction. Thus, the key matrix can just be the item embeddings matrix $\mathbf{Q}$, and we are only interested in its rows corresponding to the co-occurrence contexts $\tilde{\mathcal{C}}^{\mathcal{I}}(i)$. However, not all items in the co-occurrence contexts $\tilde{\mathcal{C}}^{\mathcal{I}}(i)$ are uniformly important. As motivated in Sect. 1, while a camera could co-occur with both a memory card and a pen, the memory card is likely to be a more important context for the camera. To enable fine-grained differentiation of the co-occurrence items, we employ the attention mechanism to focus on the important or relevant items. Specifically, we learn an attention vector $\mathbf{a}^{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}|}$ to weigh the co-occurrence items, based on the similarity between the given user-item interaction $\mathbf{z}_{ui}$ and each row of the key matrix $\mathbf{K}^{\mathcal{I}}$:

$$\mathbf{a}^{\mathcal{I}}_{i'} = \begin{cases} \mathbf{z}^T_{ui}\mathbf{K}^{\mathcal{I}}_{i'}, & \forall i' \in \tilde{\mathcal{C}}^{\mathcal{I}}(i), \\ 0, & \forall i' \notin \tilde{\mathcal{C}}^{\mathcal{I}}(i). \end{cases} \tag{3.12}$$

Subsequently, for a user-item pair $\langle u, i\rangle$, we formulate the embedding of its item co-occurrence contexts by weighing relevant information from the memory matrix $\mathbf{M}^{\mathcal{I}}$ with the attention vector:

$$\mathbf{o}^{\mathcal{I}}_{ui} = \sum_{i' \in \tilde{\mathcal{C}}^{(\mathcal{I})}(i)} \frac{\exp\left(\mathbf{a}^{\mathcal{I}}_{i'}\right)}{\sum_{i'' \in \tilde{\mathcal{C}}^{\mathcal{I}}(i)} \exp\left(\mathbf{a}^{\mathcal{I}}_{i''}\right)} \mathbf{M}^{\mathcal{I}}_{i'}. \tag{3.13}$$

Finally, the user co-occurrence contexts can be similarly modeled by the user CCM, which mirrors the item CCM above. In brief, it utilizes a key matrix $\mathbf{K}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|\times d}$ and a memory matrix $\mathbf{M}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|\times d}$, based on which we learn an attention vector $\mathbf{a}^{\mathcal{U}} \in \mathbb{R}^{|\mathcal{U}|}$ to weigh the importance of user contexts similarly as Eq. (3.12). Lastly, the embedding of the user co-occurrence contexts $\mathbf{o}^{\mathcal{U}}_{ui}$ can be obtained similarly as Eq. (3.13). We omit the equations due to space constraint.

3.4 Multi-hop Multiplex Memory A single memory layer may be inadequate in capturing more complex interactions and co-occurrence contexts. To increase the expressiveness, we stack multiple hops of the multiplex memory layer as shown in Fig. 2(c).

In the single-hop IM discussed earlier, for a given user-item pair $\langle u, i\rangle$ with input embedding $\mathbf{z}_{ui}$, our model query the key matrix $\mathbf{K}$ and aggregate the memory matrix $\mathbf{M}$ to derive the output embedding $\mathbf{o}_{ui}$. To extend to multiple hops, let $\mathbf{z}^{(l)}_{ui}$ and $\mathbf{o}^{(l)}_{ui}$ be the input and output of the l^{th} hop of the interaction memory. Following [1], we apply a nonlinear ReLU projection between hops, transforming the input of the $l+1^{\text{th}}$ hop as

$$\mathbf{z}^{(l+1)}_{ui} = \text{ReLU}(\mathbf{W}^{(l)}\mathbf{z}^{(l)}_{ui} + \mathbf{o}^{(l)}_{ui} + \mathbf{b}^{(l)}), \tag{3.14}$$

where $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weight matrix and bias vector, respectively. In this way, we are able to stack arbitrary hops to capture complex interactions between users and items. Note that the input of the first hop $\mathbf{z}^{(0)}_{ui}$ is obtained by Eq. (3.3).

Similarly, the input of the $l+1^{\text{th}}$ hop for user and item CCM can be calculated as follows:

$$\mathbf{z}^{\mathcal{U}(l+1)}_{ui} = \text{ReLU}(\mathbf{W}^{\mathcal{U}(l)}\mathbf{z}^{\mathcal{U}(l)}_{ui} + \mathbf{o}^{\mathcal{U}(l)}_{ui} + \mathbf{b}^{\mathcal{U}(l)}), \tag{3.15}$$

$$\mathbf{z}^{\mathcal{I}(l+1)}_{ui} = \text{ReLU}(\mathbf{W}^{\mathcal{I}(l)}\mathbf{z}^{\mathcal{I}(l)}_{ui} + \mathbf{o}^{\mathcal{I}(l)}_{ui} + \mathbf{b}^{\mathcal{I}(l)}), \tag{3.16}$$

which enable us to stack arbitrary hops to capture complex co-occurrence contexts of users and items. Again, the input of the first hop $\mathbf{z}^{\mathcal{U}(0)}_{ui}$ and $\mathbf{z}^{\mathcal{I}(0)}_{ui}$ is obtained by Eq. (3.3).

3.5 Output Layer and Final Loss After L hops of the multiplex memory layer, for a user-item pair $\langle u, i\rangle$, we obtain the interaction memory embedding $\mathbf{o}^{(L)}_{ui}$ and co-occurrence memory embeddings $\mathbf{o}^{\mathcal{U}(L)}_{ui}$ and $\mathbf{o}^{\mathcal{I}(L)}_{ui}$. We integrate them into a unified representation

$$\widetilde{\mathbf{o}}_{ui} = \mathbf{o}^{\mathcal{U}(L)}_{ui} \oplus \mathbf{o}^{(L)}_{ui} \oplus \mathbf{o}^{\mathcal{I}(L)}_{ui}, \tag{3.17}$$

where $\oplus$ is the concatenation operator. We feed the unified representation $\widetilde{\mathbf{o}}_{ui}$ into a multi-layer perceptron as the final prediction function:

$$\hat{r}_{ui} = \text{ReLU}(\mathbf{W}_N \ldots \text{ReLU}(\mathbf{W}_1\widetilde{\mathbf{o}}_{ui} + \mathbf{b}_1) + \ldots + \mathbf{b}_N). \tag{3.18}$$

Here, $\mathbf{W}_*$ and $\mathbf{b}_*$ denote the weight and bias in each layer. Following [2, 4], we learn the parameters of our model based on the cross entropy loss equipped with negative sampling [18], as follows.

$$\sum_{\langle u,i\rangle \in \mathcal{R}^+ \bigcup \mathcal{R}^-} r_{ui} \log \hat{r}_{ui} + (1 - r_{ui}) \log(1 - \hat{r}_{ui}) + \lambda||\Theta||, \tag{3.19}$$

where $\mathcal{R}^+$ and $\mathcal{R}^-$ denote the positive and negative user-item interactions, respectively. Moreover, r_{ui} is the ground truth whereas $\hat{r}_{ui}$ is the predicted score. Θ represents the parameter set of our model.

Finally, we examine the time complexity of MMCF. In a single-hop multiplex memory layer, a user-item pair (*i.e.,* $\langle u, i\rangle$) requires $O(d(|\tilde{\mathcal{C}}^{\mathcal{U}}(u)| + |\tilde{\mathcal{C}}^{\mathcal{I}}(i)| + K))$ time, where d is the embedding dimension, $|\tilde{\mathcal{C}}^{\mathcal{U}}(u)|$ (or $|\tilde{\mathcal{C}}^{\mathcal{I}}(i)|$) denotes the number of co-occurring users (or

Dataset	#User	#Item	#Rating	Density
Movielens-100k	943	1,682	100,000	6.30%
Delicious	1,050	1,196	7,698	0.61%
Ciao	6,760	11,166	146,996	0.19%
BookCross	19,571	39,702	605,178	0.08%

Table 1: Summary of datasets. The last column reports the percentage of non-zero cells in the user-item matrix.

items) based on the shifted positive PMI, and K is the number of memory slots. Note that these quantities are all relatively small numbers. In particular, d and K are constants, whereas $|\tilde{\mathcal{C}}^{\mathcal{U}}(u)|$ (or $|\tilde{\mathcal{C}}^{\mathcal{I}}(i)|$) is less than 64 in 92.47% (or 89.73%) of the cases with an average value of 57.62 (or 61.27) on the Movielens dataset (other datasets show similar statistics). Given multiple hops, the total time is also linear to the number of hops L. In practice, L is typically a small constant between 2 and 5, which is expressive enough for modeling complex interactions.

4 Experiments

In this section, we conduct extensive experiments on several real-world datasets, with the aim of answering the following main research questions:

- **RQ1**: Does our MMCF model outperform state-of-art methods on the recommendation task?
- **RQ2**: How does the multiplex memory network work to improve recommendation?
- **RQ3**: How do the co-occurrence contexts benefit the recommendation task?

4.1 Experimental Setup

We first describe the settings of our experiments.

Datasets. We adopt four real-world datasets from different domains, namely, Movielens-100k[1], Delicious[2], Ciao[3] and BookCross[4]. For the Movielens dataset, we follow previous studies [2, 4] to treat a rating as an interaction record, indicating whether a user has rated an item. The other three datasets only contains the interaction records of users. Table 1 summarizes the descriptions of the four datasets.

Evaluation protocol. To evaluate the recommendation performance, we adopt the *leave-one-out* evaluation, which holds out the most recent interaction as the test set and utilizes the remaining data for training. Following the strategy in existing works [2, 4, 5], we randomly sample 100 items that are not interacted by the user, and further rank the test item among the sampled items. We subsequently evaluate the performance of the top-N ranked list using *Hit Ratio* (HR) and *Normalized Discounted Cumulative Gain* (NDCG).

Baselines. We consider three categories of representative recommendation methods: traditional CF methods (*i.e.,* BPR, MF and eALS), neural network-based methods (*i.e.,* NeuMF, DMF and LRML) and methods considering high-order interactions (*i.e.,* CMN and HOP-Rec).

- **BPR** [13]: Bayesian Personalized Ranking, which optimizes the pairwise ranking.
- **MF** [7]: The standard latent factor model with the cross entropy loss [2].
- **eALS** [3]: A strong matrix factorization method, which optimizes the squared loss.
- **NeuMF** [2]: It is a neural network-based method, consisting of an MF module and an MLP model.
- **DMF** [23]: Another neural network-based method with MF and MLP.
- **LRML** [20]: A state-of-the-art memory-based attention model for recommendation, which models the relational translation for each user-item interaction.
- **CMN** [1]: Another state-of-the-art memory network model, which additionally considers similar users who also interacted with the current item. Note that it only focus on the local item in each interaction, which is different from our user co-occurrence contexts derived from the global item distribution.
- **HOP-Rec** [24]: A state-of-the-art graph-based model, which exploits random walk to gather high-order user-item interactions to enrich the interaction data. In particular, they do not consider the high-order user-user or item-item co-occurrences.

We implement the proposed MMCF model[5] in Tensorflow. We employ Xavier initialization to initialize model parameters, and use a pre-trained MF model to initialize user and item embeddings. We perform Adaptive Moment Estimation to optimize MMCF using a learning rate of 0.01. Unless otherwise stated, we show the results of two-hop multiplex memory layer, with an embedding size $d = 64$, memory slots $K = 64$,

[1] https://grouplens.org/datasets/movielens/100k/
[2] https://grouplens.org/datasets/hetrec-2011/
[3] https://www.cse.msu.edu/ tangjili/datasetcode/truststudy.htm
[4] https://grouplens.org/datasets/book-crossing/
[5] We will release the code and data with the final version.

Datasets	Metrics	BPR	MF	eALS	NeuMF	DMF	LRML	CMN	HOP-Rec	MMCF
Movielens-100k	HR@10	0.6766	0.6702	0.6638	0.6723	0.6458	0.6363	0.6792	0.6734	**0.6845**
	NDCG@10	0.3853	0.3869	0.3819	0.3816	0.3608	0.3665	0.3872	0.3842	**0.3934**
	HR@20	0.8388	0.8246	0.8155	0.8176	0.8091	0.8038	0.8357	0.8343	**0.8411**
	NDCG@20	0.4265	0.4250	0.4204	0.4088	0.3839	0.4088	0.4276	0.4267	**0.4342**
Delicious	HR@10	0.1629	0.1733	0.1679	0.2010	0.2323	0.1086	0.2124	0.2317	**0.2357**
	NDCG@10	0.0903	0.1011	0.1009	0.1124	**0.1284**	0.0531	0.1252	0.1261	0.1273
	HR@20	0.2524	0.2790	0.2724	0.2695	0.3238	0.1876	0.3152	0.3457	**0.3706**
	NDCG@20	0.1127	0.1285	0.1273	0.1138	0.1437	0.0728	0.1397	0.1479	**0.1602**
Ciao	HR@10	0.1709	0.2058	0.2173	0.1908	0.2381	0.1802	0.2410	0.2386	**0.2565**
	NDCG@10	0.0839	0.1070	0.1072	0.0967	0.1228	0.0879	0.1240	0.1216	**0.1264**
	HR@20	0.3025	0.3194	0.3543	0.3101	0.3837	0.3151	0.4080	0.4171	**0.4306**
	NDCG@20	0.1168	0.1304	0.1416	0.1190	0.1457	0.1207	0.1531	0.1621	**0.1699**
BookCross	HR@10	0.2284	0.2568	0.2642	0.2608	0.2930	0.2716	0.3076	0.3014	**0.3252**
	NDCG@10	0.1243	0.1352	0.1496	0.1368	0.1540	0.1425	0.1594	0.1617	**0.1734**
	HR@20	0.3576	0.4233	0.3947	0.4032	0.4386	0.4244	0.4512	0.4423	**0.4858**
	NDCG@20	0.1567	0.1951	0.1824	0.1752	0.1926	0.1804	0.1985	0.1974	**0.2134**

Table 2: Performance comparison on four datasets. The best method appears in boldface for each metric, and the best performing baseline appears underlined.

batch size of 512, PMI sparsity parameter $k = 20$ and regularization parameter $\lambda = 0.01$.

The key hyper-parameter settings for baseline as follows. The embedding size for all models are tuned among {16, 32, 64, 128} and the learning rate is tuned among {0.001, 0.005, 0.01}. For CMN, we tune the number of memory hop L among {1,2,3,4} for four datasets. For HOP-Rec, the parameter K is tuned from 1 to 3 for each datasets. For LRML, the memory size for the four datasets are set as: $N = 16$ for MovieLens-100k, Delicious and Ciao; $N = 32$ for BookCross. For MF, eALS and NeuMF, we follow the optimal configuration and architecture reported in [2, 3].

4.2 Performance Comparison (RQ1) The empirical results of our proposed model MMCF and the baselines are reported in Table 2. The main findings can be summarized as follows.

First, our proposed MMCF achieves the best performance in majority of the cases. Our model is particularly advantageous on sparse datasets such as Ciao, where outperforms the baselines by 6.43∼50.08% in HR@10 and 1.93∼50.06% in NDCG@10, compared to 0.78∼7.5% in HR@10 and 1.60∼9.04% in NDCG@10 on the denser Movielens dataset. This observation demonstrates the effectiveness of MMCF for the recommendation task in two aspects. For one, our model provides for a better mechanism through a multiplex memory network to jointly model the fine-grained attentive effects on user-item interaction and co-occurrence contexts. Furthermore, co-occurrence contexts are very helpful in enriching sparse user-item interactions.

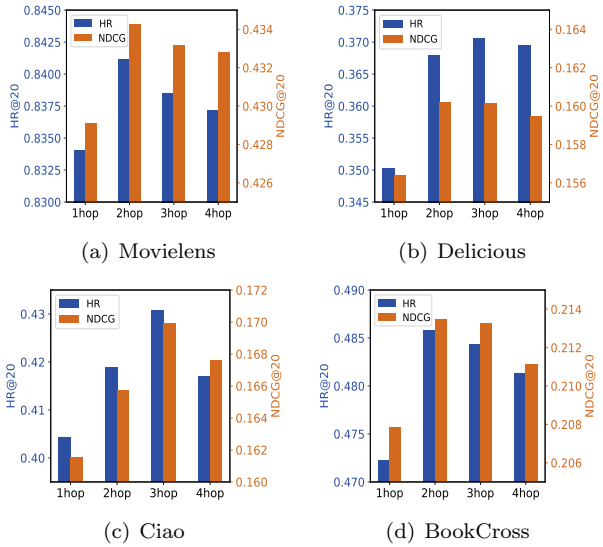
Second, neural network-based methods (*i.e.,* NeuMF, DMF and LRML) generally outperform traditional CF-based methods (*i.e.,* BPR, MF and eALS). It implies the potential ability of neural networks for modeling complex interactions in recommender systems. Moreover, Hop-Rec and CMN achieve competitive performance especially on sparse datasets, which indicates the usefulness of high-order interactions. However, CMN only focuses on users associated with the local items in each interaction, and Hop-Rec only explores high-order user-item interactions.

Datasets	Metrics	$K = 1$	$K = 16$	$K = 32$	$K = 64$
Movielens-100k	HR@20	0.8341	0.8382	0.8377	**0.8411**
	NDCG@20	0.4269	0.4301	0.4321	**0.4342**
Delicious	HR@20	0.3510	0.3590	0.3610	**0.3706**
	NDCG@20	0.1556	0.1567	0.1581	**0.1602**
Ciao	HR@20	0.4210	**0.4326**	0.4274	0.4306
	NDCG@20	0.1636	**0.1710**	0.1672	0.1699
BookCross	HR@20	0.4821	**0.4903**	0.4863	0.4858
	NDCG@20	0.2074	**0.2174**	0.2145	0.2134

Table 3: Impact of the number of memory slots K.

4.3 Analysis of Multiplex Memory Network (RQ2) Next, we investigate how our multiplex memory network affects the recommendation performance.

First, we study how the number of memory slots K in the interaction memory impacts the performance, by varying it among {1, 16, 32, 64}. Note that we employ a neural attention mechanism to weigh memory slots differently in different interactions, in order to model fine-grained latent preferences. Intuitively, each memory slot stands for one latent preference. Thus, $K = 1$ is a special case where we do not learn fine-grained preferences. As we can observe, when no fine-grained preferences are learned (*i.e.,* $K = 1$), the performance is the worst on all of the datasets. On the other hand, with



(a) Movielens (b) Delicious

(c) Ciao (d) BookCross

Figure 3: Impact of the number of hops L.

Datasets	Metrics	MMCF$_{no}$	MMCF$_U$	MMCF$_I$	MMCF$_{eq}$	MMCF
Movielens-100k	HR@20	0.8314	0.8346	0.8367	0.8336	**0.8411**
	NDCG@20	0.4265	0.4293	0.4321	0.4301	**0.4342**
Delicious	HR@20	0.3171	**0.4229**	0.3343	0.3376	0.3706
	NDCG@20	0.1517	**0.1770**	0.1555	0.1542	0.1602
Ciao	HR@20	0.4150	0.4154	0.4174	0.4123	**0.4306**
	NDCG@20	0.1656	0.1623	0.1575	0.1613	**0.1699**
BookCross	HR@20	0.4610	0.4697	0.4704	0.4633	**0.4858**
	NDCG@20	0.1954	0.2017	0.2003	0.2022	**0.2134**

Table 4: Effects of user and item co-occurrence contexts.

Datasets	Metrics	$k=1$	$k=5$	$k=10$	$k=20$	$k=30$	$k=40$
Movielens-100k	HR@20	0.8314	0.8350	0.8406	**0.8411**	0.8385	0.8396
	NDCG@20	0.4244	0.4281	**0.4363**	0.4342	0.4320	0.4334
Delicious	HR@20	0.3212	0.3514	0.3628	0.3706	**0.3728**	0.3686
	NDCG@20	0.1521	0.1550	0.1553	0.1601	**0.1634**	0.1609
Ciao	HR@20	0.4093	0.4170	0.4301	**0.4306**	0.4240	0.4240
	NDCG@20	0.1568	0.1627	**0.1715**	0.1699	0.1679	0.1677
BookCross	HR@20	0.4653	0.4723	0.4801	**0.4858**	0.4845	0.4820
	NDCG@20	0.2054	0.2080	0.2100	**0.2134**	0.2133	0.2103

Table 5: Impact of shifted positive PMI.

fine-grained learning (*i.e.,* $K > 1$), on dense datasets (*i.e.,* Movielens and Delicious) our proposed MMCF model can benefit from more memory slots, whereas fewer memory slots are adequate on sparse datasets (*i.e.,* Ciao and BookCross). Next, we vary the number of hops L among $\{1, 2, 3, 4\}$, and report the results in Fig. 3. Generally, we observe that multiple hops of the multiplex memory layer often outperform a single hop. It demonstrates that stacking multiple memory layers can better model complex interactions by leveraging additional memory components. However, it does not always yield a performance improvement with more hops due to overfitting.

4.4 Analysis of Co-occurrence Contexts (RQ3)

To examine the effect of the co-occurrence contexts, we study four variants of MMCF: (i) MMCF$_{no}$: MMCF without any co-occurrence contexts; (ii) MMCF$_U$: MMCF with only user co-occurrence contexts; (iii) MMCF$_I$: MMCF with only item co-occurrence contexts; (iv) MMCF$_{eq}$: MMCF with both user and item co-occurrence contexts, but without the attention mechanism such that each user or item receives an equal weight.

From Table 4, we observe that the general performance order among MMCF and its first three variants is as follows (except on the Delicious dataset): MMCF$>$ MMCF$_U \approx$ MMCF$_I >$ MMCF$_{no}$, which can reveal three major implications. First, user and item co-occurrence contexts are valuable high-order information that can complement user-item interactions. Thus, ignoring them altogether (*i.e.,* MMCF$_{no}$) is not ideal. Second, the attention mechanism is crucial to effectively utilizing the co-occurrence contexts, since the model without attention (*i.e.,* MMCF$_{eq}$) does not perform as well as MMCF. Third, the user and item co-occurrence contexts are complementary to rather than repetitive of each other, since integrating both of them (*i.e.,* MMCF) can further enhance the recommendation performance.

It is worth noting that on the Delicious dataset, we have MMCF$_U >$ MMCF. One possible explanation might be that the item co-occurrence contexts on the Delicious dataset are noisy, which would harm the overall model. We find out that most user and distributions follow the power law, except the item distribution on the Delicious dataset which significantly deviates from the others (see Supplementary 6.1). This deviation may indicate potential noises in item co-occurrence contexts, and thus may negatively impact the model.

Even when the users and items in co-occurrence contexts follow the power law, there may still exist considerable noises, which could be filtered using the shifted positive PMI by the value of $\log k$ in Eq. 3.10 and Eq. 3.11. We examine the impact of such filtering by varying $k \in \{1, 5, 10, 20, 30, 40\}$ and present the performance change in Table 5. We observe that the performance initially increases with the increase of k, and the optimal performance is generally obtained around $10 \leq k \leq 30$. This implies that noises may exist when not enough filtering is done. Subsequently, the performance drops if we further increase k, as we have filtered some useful co-occurrence contexts.

4.5 Additional experiments. Apart from the above main research questions, we further conducted experiments on parameter sensitivity on the embedding dimension d and the regularization parameter λ. Moreover, we showcase a visualization of the learned atten-

tion, which demonstrates that our multiplex memory network can model fine-grained semantics. Due to the space constraint, they are included in Supplementary 6.2 and 6.3, respectively.

5 Conclusion

In this paper, we firstly proposed a novel multiplex memory network for collaborative filtering (MMCF). The key contribution of MMCF is the concurrent modeling of multiple types of memory, to jointly capture user-item interactions and high-order co-occurrence contexts in a fine-grained manner. Specifically, using a neural attention mechanism, the interaction memory learn the weights for various latent preferences underpinning each user-item interaction, whereas the co-occurrence context memories learn to locate the important and relevant user or item. Finally, we conduct extensive experiments on four public datasets, and our model MMCF outperforms a range of state-of-the-art recommender models.

References

[1] T. Ebesu, B. Shen, and Y. Fang, *Collaborative memory network for recommendation systems*, in SIGIR, 2018, pp. 515–524.

[2] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, *Neural collaborative filtering*, in WWW, 2017, pp. 173–182.

[3] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, *Fast matrix factorization for online recommendation with implicit feedback*, in SIGIR, 2016, pp. 549–558.

[4] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, *Leveraging meta-path based context for top-n recommendation with a neural co-attention model*, in SIGKDD, 2018, pp. 1531–1540.

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

[6] Y. Koren and R. Bell, *Advances in collaborative filtering*, in Recommender systems handbook, 2015, pp. 77–118.

[7] Y. Koren, R. Bell, and C. Volinsky, *Matrix factorization techniques for recommender systems*, Computer, (2009), pp. 30–37.

[8] O. Levy and Y. Goldberg, *Neural word embedding as implicit matrix factorization*, in NIPS, 2014, pp. 2177–2185.

[9] S. Li, J. Kawale, and Y. Fu, *Deep collaborative filtering via marginalized denoising auto-encoder*, in CIKM, ACM, 2015, pp. 811–820.

[10] H. Ma, H. Yang, M. R. Lyu, and I. King, *Sorec: social recommendation using probabilistic matrix factorization*, in CIKM, 2008, pp. 931–940.

[11] A. Miller, A. Fisch, J. Dodge, A.-H. Karimi, A. Bordes, and J. Weston, *Key-value memory networks for directly reading documents*, in EMNLP, 2016, pp. 1400–1409.

[12] A. N. Nikolakopoulos and G. Karypis, *Recwalk: Nearly uncoupled random walks for top-n recommendation*, in WSDM, 2019, pp. 150–158.

[13] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, *Bpr: Bayesian personalized ranking from implicit feedback*, in UAI, 2009, pp. 452–461.

[14] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, et al., *Item-based collaborative filtering recommendation algorithms*, in WWW, 2001, pp. 285–295.

[15] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, *Autorec: Autoencoders meet collaborative filtering*, in WWW, 2015, pp. 111–112.

[16] C. Shi, B. Hu, W. X. Zhao, and S. Y. Philip, *Heterogeneous information network embedding for recommendation*, IEEE Transactions on Knowledge and Data Engineering, 31 (2018), pp. 357–370.

[17] S. Sukhbaatar, J. Weston, R. Fergus, et al., *End-to-end memory networks*, in NIPS, 2015, pp. 2440–2448.

[18] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, *Line: Large-scale information network embedding*, in WWW, 2015, pp. 1067–1077.

[19] J. Tang and K. Wang, *Personalized top-n sequential recommendation via convolutional sequence embedding*, in WSDM, 2018, pp. 565–573.

[20] Y. Tay, L. Anh Tuan, and S. C. Hui, *Latent relational metric learning via memory-based attention for collaborative ranking*, in WWW, 2018, pp. 729–739.

[21] T. Tran, X. Liu, K. Lee, and X. Kong, *Signed distance-based deep memory recommender*, in WWW, 2019, pp. 1841–1852.

[22] H. Wang, N. Wang, and D.-Y. Yeung, *Collaborative deep learning for recommender systems*, in SIGKDD, 2015, pp. 1235–1244.

[23] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, *Deep matrix factorization models for recommender systems.*, in IJCAI, 2017, pp. 3203–3209.

[24] J.-H. Yang, C.-M. Chen, C.-J. Wang, and M.-F. Tsai, *Hop-rec: high-order proximity for implicit recommendation*, in RecSys, 2018, pp. 140–144.

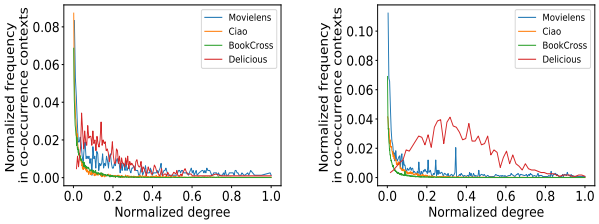
[25] L. Zheng, V. Noroozi, and P. S. Yu, *Joint deep modeling of users and items using reviews for recommendation*, in WSDM, 2017, pp. 425–434.

[26] X. Zhou, D. Liu, J. Lian, and X. Xie, *Collaborative metric learning with memory network for multi-relational recommender systems*, in IJCAI, 2019, pp. 4454–4460.

[27] X. Zhou, C. Mascolo, and Z. Zhao, *Topic-enhanced memory networks for personalised point-of-interest recommendation*, in SIGKDD, 2019, pp. 3018–3028.

6 Supplementary

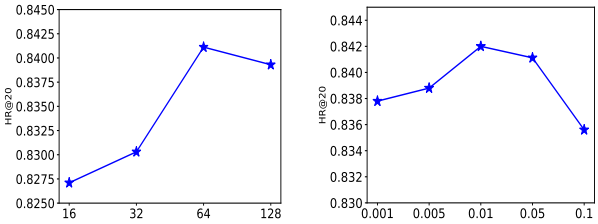
6.1 Frequency distribution of users and items in co-occurrence contexts To better understand the item irregularity on Delicious, we plot the frequency distribution of users and items in co-occurrence contexts with respect to their degrees, on the four datasets in Fig. 4.



(a) User distribution (b) Item distribution

Figure 4: Distribution of users and items in co-occurrence contexts. We normalize the degree and frequency between [0, 1] for better visualization.

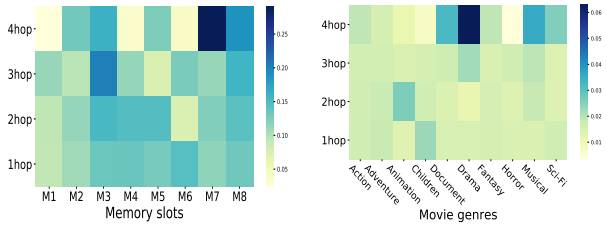
6.2 Parameter Sensitivity In addition, we examine the impact of the embedding size d and the regularization parameter λ. For brevity we only present the results on the Movielens datasets using HR@20 as the metric. Similar findings have been observed on other datasets and metrics. First, we vary the embedding size $d \in \{16, 32, 64, 128\}$ in Fig 5(a), where the performance is stable at around 64 or 128. A very small d may lack the expressiveness to model complex interactions. Second, we vary the regularization parameter $\lambda \in \{0.001, 0.005, 0.01, 0.05, 0.1\}$ in Fig. 5(b), where the performance is largely stable between 0.005 and 0.05.



(a) Embedding size d (b) Regularization parameter λ

Figure 5: Parameter study for MMCF on Movielens.

6.3 Attention Visualization Finally, to visualize how the multiplex memory layer with the multi-hop design works, we present a case study on user $u344$ and item $i1082$ in the Movielens dataset. The attentive weights learned in the IM and item CCM are shown as heatmaps in Fig. 6. As the number of hops increases, the IM attention becomes more concentrated in fewer memory slots in Fig. 6(a). Meanwhile, each cell in Fig. 6(b) represents the mean attention weights of the movies in a genre in the co-occurrence contexts of movie $i1082$, and we showcase the ten most relevant genres. We observe that the genre "drama" attains the highest weight after 3 hops of the item CCM, which is intuitive since movie $i1082$ also belongs to the same genre, demonstrating that the CCM can capture the most relevant context.



(a) Latent semantic information (b) Item co-occurrence memory

Figure 6: Attention visualization on Movielens.