Sequential Recommendation with Dual Side Neighbor-based Collaborative Relation Modeling

Jiarui Qin[1], Kan Ren[2], Yuchen Fang[1], Weinan Zhang[1], Yong Yu[1]
[1]Shanghai Jiao Tong University, Shanghai, China
[2]Microsoft Research Asia, Beijing, China
{qinjr, arthur_fyc, wnzhang, yyu}@apex.sjtu.edu.cn, kan.ren@microsoft.com

ABSTRACT

Sequential recommendation task aims to predict user preference over items in the future given user historical behaviors. The order of user behaviors implies that there are resourceful sequential patterns embedded in the behavior history which reveal the underlying dynamics of user interests. Various sequential recommendation methods are proposed to model the dynamic user behaviors. However, most of the models only consider the user's own behaviors and dynamics, while ignoring the collaborative relations among users and items, i.e., similar tastes of users or analogous properties of items. Without modeling collaborative relations, those methods suffer from the lack of recommendation diversity and thus may have worse performance. Worse still, most existing methods only consider the user-side sequence and ignore the temporal dynamics on the item side. To tackle the problems of the current sequential recommendation models, we propose **S**equential **Co**llaborative **Re**commender (SCoRe) which effectively mines high-order collaborative information using cross-neighbor relation modeling and, additionally utilizes both user-side and item-side historical sequences to better capture user and item dynamics. Experiments on three real-world yet large-scale datasets demonstrate the superiority of the proposed model over strong baselines.

KEYWORDS

Sequential Recommendation, Collaborative Filtering, Co-Attention

ACM Reference Format:

Jiarui Qin, Kan Ren, Yuchen Fang, Weinan Zhang, Yong Yu. 2020. Sequential Recommendation with Dual Side Neighbor-based Collaborative Relation Modeling. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining (WSDM '20), February 3-7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3336191.3371842

1 INTRODUCTION

With the emergence of large online information systems such as e-commerce platform, the amount of user behavioral data grows rapidly. Therefore, in recent years, the researchers in both academic and industrial fields have devoted many efforts on sequential recommendation task which aims to mine the resourceful yet complex temporal dynamics embedded in user behavior sequences.

As has been stated in many related works [1, 9, 13, 19], the temporal dynamics have high impacts on the future user behaviors, especially accounted for concept drifting [35], long-term behavior dependency [19], periodic patterns [26], etc.

Commonly, the current sequential recommendation models regard user's purchasing or browsing behaviors as "token"s in the natural language processing (NLP) field. And the mainstream models use sequential modeling techniques that are widely used in NLP such as recurrent neural networks (RNNs) [14, 43], convolutional neural networks (CNNs) [32] and Transformer [17]. These models make huge success on sequential recommendation task with many deployed real-world applications [39, 43].

Despite the success of current sequential recommendation methods, there are still some limitations of them. The first one is that most of the models [14, 17, 32] only consider user's (or item's) own interaction history, while ignoring similar users or items that have collaborative relations with itself. Therefore, each user (item) only know her (its) own behaviors, it is bad for the variety of recommendation and may hurt the recommendation performance.

We could regard the user-item interactions as a bipartite graph in which the nodes are users and items, and links are interaction records as illustrated in Figure 1. Traditional models [18, 19] only consider the directly interacted items (users) of the target user (item) which are the 1-hop neighbors of the target node. In this way, it is difficult to capture the collaborative relations among users and items. But when we make a step forward and consider the 2-hop neighbors, we find that the neighbors in 2-hop have collaborative relations with the target node because both the 2-hop neighbors and the target node have interacted with the same group of nodes which are the 1-hop neighbors. As these relations are found through 2-hops on the graph, we call them *high-order collaborative relations*.

By this means, we may find the corresponding collaborative information for the target user or item. Moreover, there are various patterns across the neighbors that can be utilized. By aggregating these collaborative relations to the representation of users and items, we could model more complex and various user interests (item attractions). And the collaborative modeling can be done in a sequential way to better handle the temporal dynamics.

Another key limitation of current models is that they only consider the user-side temporal dynamics while ignoring the ones on the item side. The user-side sequence consists of the items that are browsed by the user, and thus it could reveal the user's drifting interests. However, the item-side also contains sequential patterns:

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-1-4503-6822-3/20/02…$15.00
https://doi.org/10.1145/3336191.3371842

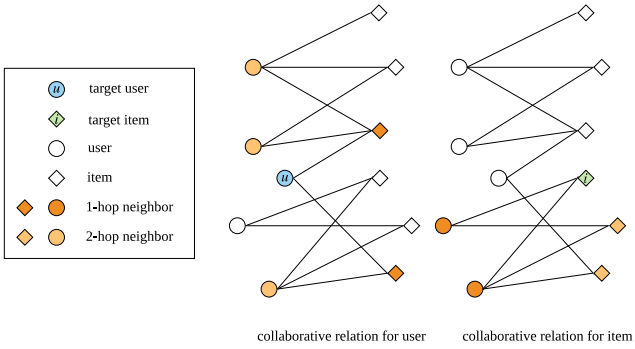

Figure 1: Graph illustration of user-item interactions.

an item attracts different users at different time which could reveal the item dynamics such as popularity trend or social topic drift. For example, the recommender system may present Christmas card to a user when the holiday is coming even *before* her interacting with any related items because the Christmas card has attracted the other users who share similar interests or collaborative relations to that specific user. The modeling of item-side sequence is similar with *information dissemination* [22, 29], which means the item information disseminates from users to users at different time and to predict which user will be the next one that the information disseminates to.

There are already some sequential recommendation models that have tried combining user-side and item-side sequences to perform dual sequence modeling [36, 37]. However, these works intend to consider the two sequences in a relatively independent manner and the sequential representations of both sequences have interactions only in the final prediction stage. Nevertheless, our work aims to model the dual sequences in a more interactive way which means the information of both sequences have interactions along the timeline. As we do collaborative relation capturing from both user-side and item-side, it is natural that we interact both sequences at synchronized time which is illustrated in Figure 2.

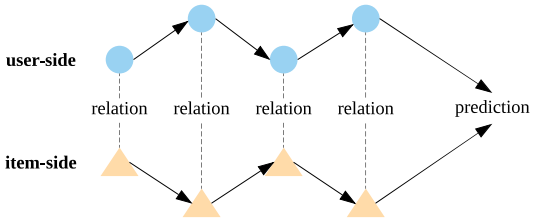

Figure 2: Illustration of interactive dual sequence modeling.

To address the limitations mentioned above, we propose **S**equential **Co**llaborative **Re**commender (SCoRe) which considers high-order collaborative relations and models dual sequences in an interactive and more expressive manner. The contribution of the paper can be summarized in three-fold:

- We propose to aggregate high-order collaborative relations which could enrich the representation of users and items. More importantly, through cross neighbor relation modeling, our model can effectively capture the various and complex patterns in the neighbor-to-neighbor collaborative relations.
- We propose to model both user-side and item-side sequence. Dual sequences interactions are modeled in a more thorough way, which makes the modeling of the dual sequence more expressive.
- We conduct extensive experiments of evaluating and comparing our model with several strong baselines over three real-world yet large-scale recommendation datasets. The results have proved the efficacy of SCoRe model and the detailed analysis reveals some key principles of training our model.

The rest of the paper is organized as follows. Section 2 and Section 3 present the preliminaries and describe the SCoRe model in detail. We also make some discussions about the model efficiency. We conduct comprehensive experiments and present the experimental setups with the corresponding results in Section 4. In Section 5, we discuss about some related works. Finally we conclude the paper and point out some future works in Section 6.

2 PRELIMINARIES

In a recommender system, there are M users in $\mathcal{U} = \{u_1, \ldots, u_M\}$ and N items in $\mathcal{V} = \{v_1, \ldots, v_N\}$. In the history, any user may reveal interests on some items and the interaction behaviors would be tracked in the system as $\mathcal{Y} = \{y_{uv} | u \in \mathcal{U}, v \in \mathcal{V}\}$ and

$$y_{uv} = \begin{cases} 1, & u \text{ has interacted with } v; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The user preference y is either implicit feedback [1], e.g., clicks, or explicit user rating [19]. Without loss of generality, we focus on the implicit feedback which is more common in practice. For sequential recommendation, each user-item interaction has the corresponding timestamp ts, thus we use the triplet (u, v, ts) to denote one interaction. All the observed interaction records are denote as $\mathcal{T} = \{u, v, ts\}$.

2.1 Interaction Relations

For a target pair of (u, v) that we need to predict the interaction probability, we could extract some interaction relations of the target user u and target item v.

Definition 1. (Interaction Set): Given the target user u, we can conduct the *interaction set* of u as

$$G(u) = \{v_j | y_{uv_j} = 1\}. \tag{2}$$

The user's interaction set is the collection of all the items that the user has interacted with. Symmetrically, we can define the item v's interaction set as,

$$G(v) = \{u_i | y_{u_i v} = 1\}. \tag{3}$$

which contains all the users that have interaction with the item v.

Definition 2. (Co-Interaction Set): To explicitly capture the collaborative relations among users and items, i.e., similar users or similar items, it is natural to consider the users (items) that have similar tastes (attractions). Therefore we define the *co-interaction set* of u and v respectively as

$$G(v|u) = \{u_i | y_{u_i v_j} = 1, v_j \in G(u)\}, \tag{4}$$

$$G(u|v) = \{v_j | y_{u_i v_j} = 1, u_i \in G(v)\}. \tag{5}$$

Table 1: Notations and descriptions

Notation	Description.
u, v	The target user and the target item.
M, N	The number of users and items.
$y, \hat{y}$	The indicator and the predicted probability of the user-item interaction.
$\boldsymbol{u}, \boldsymbol{v}$	Dense representation of target user u and target item v.
$G^t(u), G^t(v)$	User u's/item v's interaction set at t-th time slice.
$G^t(v\|u), G^t(u\|v)$	User u's/item v's co-interaction set at t-th time slice.
G_u^t, G_v^t	User u's/item v's interaction relations at t-th time slice.
$\boldsymbol{u}_t^{agg}, \boldsymbol{v}_t^{agg}$	Aggregated user/item-side representation at t-th time slice.
U, V	User/item-side sequence.
ΔT	Time interval to split the whole timeline.
S	Size of (co-)interaction set.

For user u, the co-interaction set $G(v|u)$ actually consists of a group of users that shares similar behaviors with u, because they all interacted with the items in u's interaction set. Therefore they have collaborative relations in some extent. For item v, similarly, the co-interaction set $G(u|v)$ consists of the items that attract the same group of users (v's interaction set) with the target item v.

As shown in Figure 1, we can tell that the interaction set and co-interaction set are essentially the 1-hop and 2-hop neighbors of the rooted node (u or v) respectively.

2.2 Evolving Time-sliced Interaction Relations

Now that we have defined the local interaction relations of the user u (item v), we take one step forward and consider it in a temporal way. Specifically, the users (items) may conduct different interactions with different items (users) at different time. Thus, the interaction relations are evolving all the time and could be regarded as a series of time-sliced processes.

To better model the temporal patterns of the interaction relations, we slice the whole timeline into T time frames, each of which is constructed within a unified time interval ΔT. In this way, all observed interactions $\mathcal{T} = \bigcup_{t=1}^{T} \mathcal{T}^t$ and $\mathcal{T}^t$ contains the triplet (u, v, ts) that happens in the t-th time slice. Using the interaction records within $\mathcal{T}^t$, we could construct the user u's and item v's interaction relations. We denote them as,

$$G_u^t = \{G^t(u), G^t(v|u)\}, \tag{6}$$

$$G_v^t = \{G^t(v), G^t(u|v)\}. \tag{7}$$

2.3 Task Definition

The goal of the recommender system is to estimate the probability of interactions $\hat{y}$ between the target user $u \in \mathcal{U}$ and the given item $v \in \mathcal{V}$, with consideration of the user's interaction history G_u and the item's interaction history G_v as

$$\hat{y}_{uv} = f(u, v|G_u, G_v; \boldsymbol{\theta}) \tag{8}$$

through the learned function f with parameters $\boldsymbol{\theta}$ where $G_u = \bigcup_{t=1}^{T} G_u^t$ and $G_v = \bigcup_{t=1}^{T} G_v^t$. We conclude the notations and the corresponding descriptions in Table 1.

3 METHODOLOGY

In this section, we present our proposed model SCoRe (**S**equential **Co**llaborative **Re**commender) in detail. We first introduce the high-order collaborative relation mining through cross neighbor modeling, and then we describe the dual sequence modeling in an interactive manner. Furthermore, we analyze the time complexity of the proposed model.

3.1 High-Order Collaborative Relation Mining

In this section, we describe the proposed *Co-Attention Network* for handling the complex relations across the neighbors of interaction set and co-interaction set.

3.1.1 Cross Neighbor Co-Attention Network. At each time slice t, we use *Co-Attention Network* to capture the complex relations across neighbors in interaction and co-interaction sets.

One of the key parts of recent success of recommendation models [31, 38, 43] are the attention mechanism which attributes different credits to different item representations or temporal representations (e.g. hidden states of RNNs). The attentive weight of a user interacted item v_j w.r.t target item v is calculated following the paradigm as,

$$\alpha_{v_j, v} = Attn(r(v_j), v), v_j \in G(u), \tag{9}$$

where the $Attn(\cdot)$ function can be various, $r(\cdot)$ is representation of v_j which could be embedding or hidden states. The calculated $\alpha_{v_j, v}$ measures the correlation (e.g. similarity) between v_j and v. This paradigm only focuses on the relations between user interacted items and **single** target item v. But there are many neighboring items of v (those items in $G(u|v)$), so we could calculate neighbor-to-neighbor correlations between items in $G(u|v)$ and those in $G(u)$. In this way, the relation between target u and v can be modeled with more resourceful information.

To model this cross neighbor collaborative relations, we propose Co-Attention Network, which is illustrated in Figure 3. We not only consider the relatedness between the user interacted items and the target item, but also take the relatedness across user interacted items and the collaborative neighbors of the target item into account.

At each time slice, we calculate a co-attention relatedness matrix, $A_t^{\text{item}} \in R^{S \times S}$, each element of which is calculated as

$$\alpha_{i,j}^t = \sigma(\boldsymbol{w}_1^T[\boldsymbol{v}_i, \boldsymbol{v}_j, \boldsymbol{v}] + b), v_i \in G^t(u), v_j \in G^t(u|v) \tag{10}$$

where $\boldsymbol{v}$ is the embedding of the target item v, σ is the ReLU activation function $ReLU(x) = max(0, x)$ and S is the number of items in $G^t(u)$ and $G^t(u|v)$. As the objects in $G^t(u)$ and $G^t(u|v)$ are all items, we denote this relatedness matrix as A_t^{item}. Followed by a softmax operation, we get the co-attention matrix C_t^{item}, each element of which is calculated as,

$$c_{i,j}^t = \frac{\exp(\alpha_{i,j}^t)}{\sum_{m,n} \exp(\alpha_{m,n}^t)}. \tag{11}$$

Symmetrically, we could calculate each element of A_t^{user} as

$$\alpha_{i,j}^t = \sigma(\boldsymbol{w}_2^T[\boldsymbol{u}_i, \boldsymbol{u}_j, \boldsymbol{u}] + b), u_i \in G^t(v), u_j \in G^t(v|u) \tag{12}$$

and C_t^{user} using the softmax operation described in Eq. (11).

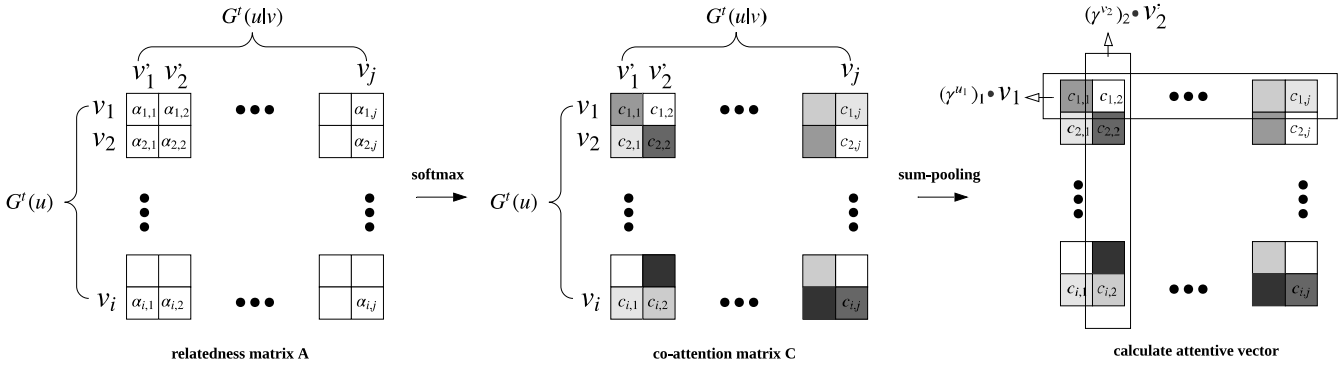

Figure 3: The processes of calculating the co-attention values. First, calculate relatedness matrix A; Second, softmax over the whole matrix A and get the co-attention matrix C; Lastly, use sum pooling along rows and columns to get attentive vector and use the vector to weighted sum over objects in (co-)interaction set.

In this way, we could capture cross neighbor relations which are more complex and resourceful than the original paradigm described in Eq. (9).

3.1.2 high-order Information Aggregation. In collaborative filtering (CF) models [6, 12, 18] and sequential recommendation models [14, 32, 43], we only use the user (item) directly interacted items (users) to represent a user (item) which may cause a narrow understanding of user or item properties.

However, in previous section, by incorporating the information of co-interaction sets, not only could we model the cross neighbor collaborative relations but integrate high-order information by summarizing the co-interaction set. By integrating these co-interaction objects, we could enrich the representation of target user u and item v.

By sum pooling (SP) along the rows or columns of the two co-attention matrix C_t^{item} and C_t^{user}, we could get four attentive vector,

$$\boldsymbol{\gamma}_t^{u_1} = SP(\{C_{ij}^{\text{item}}\}_{j=1}^{S}), \tag{13}$$

$$\boldsymbol{\gamma}_t^{v_2} = SP(\{C_{ij}^{\text{item}}\}_{i=1}^{S}), \tag{14}$$

$$\boldsymbol{\gamma}_t^{u_2} = SP(\{C_{ij}^{\text{user}}\}_{j=1}^{S}), \tag{15}$$

$$\boldsymbol{\gamma}_t^{v_1} = SP(\{C_{ij}^{\text{user}}\}_{i=1}^{S}). \tag{16}$$

We denote the embeddings of interaction set and co-interaction set for u and v at t-th time slice as $\boldsymbol{U}_t^1 \in R^{S\times d}$ (interaction), $\boldsymbol{U}_t^2 \in R^{S\times d}$ (co-interaction), $\boldsymbol{V}_t^1 \in R^{S\times d}$, $\boldsymbol{V}_t^2 \in R^{S\times d}$ respectively, where d is the dimension of the embedding and S is the size of (co-)interaction set. The aggregated representation of u and v at t-th time slice are

$$\boldsymbol{u}_t^{agg} = [\boldsymbol{U}_t^{1\,T}\boldsymbol{\gamma}_t^{u_1}, \boldsymbol{U}_t^{2\,T}\boldsymbol{\gamma}_t^{u_2}] \tag{17}$$

$$\boldsymbol{v}_t^{agg} = [\boldsymbol{V}_t^{1\,T}\boldsymbol{\gamma}_t^{v_1}, \boldsymbol{V}_t^{2\,T}\boldsymbol{\gamma}_t^{v_2}]. \tag{18}$$

3.2 Interactive Dual Sequence Modeling

In this section, we describe our approach on temporal dynamics modeling. We conduct a dual sequence modeling method which considers both user-side and item-side sequences and interactively models the relations among two sequences of synchronized time slice.

3.2.1 Temporal Dynamics Modeling. At each time slice, we get the aggregated representations of target user u and target item v as $\boldsymbol{u}_t^{agg}$ and $\boldsymbol{v}_t^{agg}$ respectively following the co-attention mechanism in Section 3.1. After that we get two sequences U and V, which are the sequences of target user's and item's aggregated representation at different time respectively. For simplicity, we denote $\boldsymbol{u}_t = \boldsymbol{u}_t^{agg}$ and $\boldsymbol{v}_t = \boldsymbol{v}_t^{agg}$ thus

$$U = \{\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_T\}, V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_T\}. \tag{19}$$

We use two recurrent neural network models to model the temporal dynamics for user-side and item-side respectively. And we implement each recurrent cell as Gated Recurrent Unit [7] (GRU). Each GRU unit takes the corresponding representation $\boldsymbol{u}_t$ (or $\boldsymbol{v}_t$) at each time step and the hidden state $\boldsymbol{h}_{t-1}$ from the last time step, and then calculates as

$$\begin{aligned}
\boldsymbol{z}_t^u &= \sigma(\overline{\boldsymbol{W}}_z^u \boldsymbol{u}_t + \overline{\boldsymbol{U}}_z \boldsymbol{h}_{t-1}^u + \overline{\boldsymbol{b}}_z^u) \\
\boldsymbol{r}_t^u &= \sigma(\overline{\boldsymbol{W}}_r^u \boldsymbol{u}_t + \overline{\boldsymbol{U}}_r \boldsymbol{h}_{t-1}^u + \overline{\boldsymbol{b}}_r^u) \\
\boldsymbol{h}_t^u &= (1 - \boldsymbol{z}_t^u) \odot \boldsymbol{h}_{t-1}^u \\
&\quad + \boldsymbol{z}_t^u \odot \tanh(\overline{\boldsymbol{W}}_h^u \boldsymbol{u}_t + \overline{\boldsymbol{U}}_h^u (\boldsymbol{r}_t^u \odot \boldsymbol{h}_{t-1}^u) + \overline{\boldsymbol{b}}_h^u),
\end{aligned} \tag{20}$$

where $\odot$ is the element-wise product operator.

The item-side temporal dynamics are modeled in the same way. Till now, we've got user-side and item-side sequence of temporal representations: $\{\boldsymbol{h}_1^u, \boldsymbol{h}_2^u, ..., \boldsymbol{h}_T^u\}$ and $\{\boldsymbol{h}_1^v, \boldsymbol{h}_2^v, ..., \boldsymbol{h}_T^v\}$.

3.2.2 Interactive Attention Mechanism of Dual Sequence. As illustrated in Figure 4, different time slice has different impact on the final prediction at $(T+1)$. And hereby we introduce our *Interactive Attention Mechanism*. Unlike the attention mechanism in [43] and [44] which uses the target item to query the interacted items sequence, we utilize dual sequences information at the same time interactively to weigh across different time slice. The attention value of each time slice β_t is calculated as,

$$relation_t = R([\boldsymbol{h}_t^u, \boldsymbol{h}_t^v, \boldsymbol{\gamma}_t^{u_1}, \boldsymbol{\gamma}_t^{v_2}, \boldsymbol{\gamma}_t^{u_2}, \boldsymbol{\gamma}_t^{v_1}], [\boldsymbol{u}, \boldsymbol{v}]) \tag{21}$$

$$\beta_t = \frac{\exp(relation_t)}{\sum_{i=1}^{T}\exp(relation_i)} \tag{22}$$

where R is a three-layer MLP with ReLU activation function.

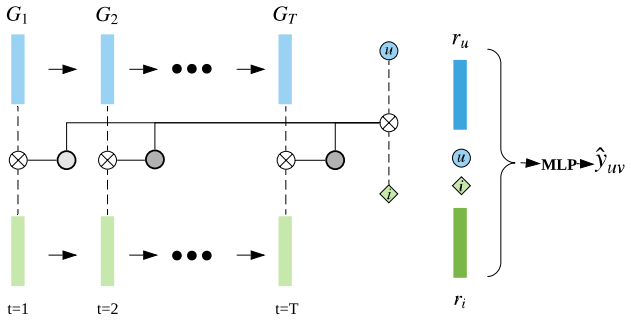

Figure 4: Temporal Interactive Dual Sequence Modeling of SCoRe.

And the final representations of user-side and item-side sequences are

$$\boldsymbol{r}_u = \sum_{t=1}^{T} \beta_t \boldsymbol{h}_t^u, \ \boldsymbol{r}_v = \sum_{t=1}^{T} \beta_t \boldsymbol{h}_t^v \tag{23}$$

It is natural that we consider both side information to calculate attention, because the cross neighbor relations modeling described in Section 3.1 utilizes both user-side neighbors and item-side neighbors. As a result, we consider using both sides representations of synchronized time slice to interactively calculate attentive value. In this way, the modeling of two sequences are highly correlated.

3.3 Final Prediction and Loss Functions

The predicted probability of interaction between the target user and the target item is calculated as

$$\hat{y} = f(\boldsymbol{r}_u, \boldsymbol{r}_v, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\Theta}), \tag{24}$$

where f is implemented as a multi-layer perceptron with the ReLU activation function. The parameters set of the MLP is $\boldsymbol{\Theta}$. The inference procedure is illustrated in Figure 4.

As for the loss function, we take an end-to-end training and introduce (i) the widely used cross entropy loss $\mathcal{L}_{\text{ce}}$ [25, 43, 44] over the whole training dataset and (ii) the parameter regularization $\mathcal{L}_{\text{r}}$. We utilize Adam algorithm for optimization. Thus the final loss function is

$$\begin{aligned} \arg\min_{\boldsymbol{\Theta},\boldsymbol{\Phi}} &= \mathcal{L}_{\text{ce}} + \lambda \mathcal{L}_r \\ &= -\sum_s \left[y_s \log \hat{y}_s + (1 - y_s) \log(1 - \hat{y}_s) \right] \\ &+ \frac{1}{2} \lambda \left(\|\boldsymbol{\Theta}\|_2^2 + \|\boldsymbol{\Phi}\|_2^2 \right), \end{aligned} \tag{25}$$

where $\boldsymbol{\Phi}$ includes the parameters in GRUs, $\boldsymbol{w}_1$, $\boldsymbol{w}_2$ in Co-Attention Network and the parameters of the three-layer MLP R in the Interactive Attention Mechanism.

3.4 Model Efficiency

In this section, we analyze the computational complexity of our SCoRe model. From the previous sections, we can tell that the forward inference of SCoRe can be regarded as two relatively separate parts. The first part is the cross neighbor collaborative relations modeling, which can be paralleled conducted for each time slice. The cost of it can be viewed as a constant $O(C_{co-atten})$ as the co-attention network conduct single layer non-linear transformation, softmax and sum-pooling operations. The second part is the GRU temporal modeling. We assume the average time performance of the GRU is a constant $O(C_{GRU})$ which is related to the implementation of the GRU module yet can be parallelly executed through GPU processor. Recall that we have T time slices, thus the time complexity of temporal inference is $O(T \cdot C_{GRU})$. Therefore the overall time complexity of SCoRe is $O(C_{co-atten}) + O(T \cdot C_{GRU}) = O(T \cdot C_{GRU})$ which is the time complexity of ordinary recurrent neural networks.

The inference time complexity is acceptable since several implementations sharing similar execution complexity have been adopted online [39, 43], which indicates online inference efficiency for our SCoRe model in some extent.

4 EXPERIMENTS

In this section, we present the details of the experiment setups and the corresponding results. To illustrate the effectiveness of our proposed model, we compare it with some strong baselines on sequential recommendation task. Moreover, we have published our reproductive code[1].

We start with three research questions (RQ) to lead the experiments and the following discussions.

RQ1 Compared to the baseline models, does SCoRe achieve state-of-the-art performance in sequential recommendation task?

RQ2 What is the influence of different components in SCoRe? Are the proposed co-attention network and interactive attention necessary for improving performance?

RQ3 What patterns does the proposed model capture for the final recommendation decision?

4.1 Experimental Setups

In this part, we describe our experiment setups including datasets with preprocessing method, some important implementation details, evaluation metrics and the compared baselines.

4.1.1 Datasets. We use three real-world large-scale datasets to evaluate all the compared models. The dataset statistics have been shown in Table 2.

CCMR [4] is a dataset consists of movie rating (from integer score 1 to 5) logs collected from Douban, which is one of China's largest movie review websites. The data is collected and dumped in May 2015.

Taobao [45] is a dataset consisting of user behavior data collected from Taobao[2], one e-commerce platform in China. It contains user behaviors from November 25 to December 3, 2017 of several behavior types including click, purchase, add to cart and item favoring.

Tmall [3] is provided by Alibaba Group which contains user behavior history on Tmall e-commerce platform from May 2015 to November 2015.

Dataset Preprocessing. We cut the time line into total T time slices with the specific time interval as shown in Table 2. And for each time slice, we use the interaction records within it to construct interaction and co-interaction set for both user and item. Here we

[1] https://github.com/qinjr/SCoRe
[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=649
[3] https://tianchi.aliyun.com/dataset/dataDetail?dataId=42

Table 2: The dataset statistics.

Dataset	Users #	Items #	Interaction #	Time slices T	Time interval ΔT
CCMR	4,920,695	190,129	283,775,314	41	90 days
Taobao	987,994	4,162,024	100,150,807	9	1 day
Tmall	424,170	1,090,390	54,925,331	13	15 days

use a simple way to do time slicing, we leave finer segmentation strategy in future work.

Positive & Negative Samples. To evaluate the recommendation performance, we use one positive item and sample 100 negative items at the prediction time $(T+1)$ for each user in all three datasets. For Tmall and Taobao datasets, as we only have the positive user feedbacks (click, buying, etc.), we have to randomly sample the negative items. As for CCMR datasets, we regard items whose ratings are 5 or 4 as positive items and those whose ratings are 1,2 or 3 as negative items. If a user does not have enough negative items, we use random sampling to generate negative items for her. The positive items in CCMR form the behavior sequence.

Train & Test Splitting. The training set contains the sequential behaviors from the first to the $(T-2)$th time slice, we use the interactions history from 1 to $(T-3)$ to predict in $(T-2)$. For the validation set, we use the interactions data from 1 to $(T-2)$ to predict in $(T-1)$. In testing set, interactions data from 1 to $(T-1)$ are used to predict in T.

Implementation Details. It is common that the target user doesn't have any interaction record in a time slice, and similarly, the target item may be not visited by any user in a time slice. To handle this issue, we use a unified embedding vector to represent the situation.

We set the size of interaction set to S which can be regarded as a hyperparameter. For simplicity, the size of co-interaction set is S too. If there are more than S objects in a set, we use random sampling. If there are less than S objects (say k ($< S$) objects), we random sample $(S-k)$ times among the original set.

4.1.2 Evaluation Metrics. Three evaluation metrics are used and all of them are widely used in recommendation tasks.

HR@k (*Hit Ratio@k*) measures the proportion of samples that the positive item is among the top-k in all test cases which is computed as,

$$HR@k = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{I}(R_{u,v} \leq k), \tag{26}$$

where $R_{u,v}$ is the ranking position of the user u's interacting with item v, and $\mathbb{I}$ is the indicator function.

NDCG@k (Normalized Discounted Cumulative Gain) is a position-aware metric which assigns larger weights on higher ranks of the positive item, which is calculated as,

$$NDCG@k = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{2^{\mathbb{I}(R_{u,v} \leq k)} - 1}{log(R_{u,v} + 1)}. \tag{27}$$

MRR (Mean Reciprocal Rank) is another position-aware metric that is calculated as,

$$MRR = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{R_{u,v}}. \tag{28}$$

As HR@1 is equal to NDCG@1, so in this work, we report HR@{1,5,10}, NDCG@{5,10} and MRR in detail.

4.1.3 Compared Baselines. To illustrate the effectiveness of our model, we compare SCoRe with two CF models, three single sequence recommendation models and two dual sequence models. We follow [43] that all the models take the input sparse features and feed them through an embedding layer for the subsequent inference.

The first group of models are CF models:

SVD++ [18] is a hybrid method of latent factor model and neighbor-based model which is the fundamental approach of collaborative filtering recommendation. It regards all the sequential behaviors as a whole and ignores the temporal dynamics.

DELF [6] is the state-of-the-art CF method which utilizes deep neural networks to capture complex non-linear interaction patterns from both user-side and item-side.

The second group contains sequential recommendation methods that utilize single user-side sequence, which are based on RNNs, CNNs, or Transformer architecture:

GRU4Rec [14] bases on GRU and it is the first work using the recurrent cell to model sequential user behaviors.

Caser [32] is based on CNNs that uses horizontal and vertical convolutional filters to capture user behavior patterns at different scales.

SASRec [17] bases on Transformer [33], it only uses self-attention mechanism without recurrent architecture. It achieves very competitive performance in sequential recommendation task.

The third group is dual sequence recommendation models.

RRN [36] is the first RNN-based model that considers both the user- and item-side sequence. It uses sum-pooling to aggregate the information inside a time slice.

DEEMS [37] feed the user-side and item-side sequence respectively into two identical sequential models, and let the two models play a game with each other where one model will use the predicted score of the other model as feedback to guide the training.

SCoRe is our proposed model which is described in Section 3.

4.2 Evaluation Results: RQ1

The experimental results are shown in Table 3, we find several observations as below.

By comparing the performance of SCoRe and other baseline models, it outperforms baselines by 28.9% to 3.1%, 58.8% to 2.7% and 18.5% to 2.0% on MRR in CCMR, Taobao and Tmall dataset, respectively. And it also shows significant improvements on the other metrics so SCoRe achieves the state-of-the-art performance in sequential recommendation task.

For the models in Group 1, they do not consider the temporal dynamics of user behaviors thus perform not so good as models in Group 2 and 3. DELF uses both user-side and item-side interaction information so it achieves better performance than SVD++ which only utilizes user-side information.

By comparing the performance of Group 2 and 3, we find in Tmall and Taobao dataset that, Group 3 outperforms Group 2. But over CCMR dataset, SASRec and Caser are better than Group 3.

Table 3: Performance comparison against baseline models. Bold values are the best in each row, while the second best values are underlined. Improvements over baselines are statistically significant with $p < 0.01$. (HR, NDCG, MRR: the higher, the better)

Dataset	Metric	Group 1		Group 2			Group 3		
		SVD++	DELF	GRU4Rec	Caser	SASRec	RRN	DEEMS	SCoRe
CCMR	HR@1	0.0797	0.0755	0.0739	0.0845	0.0817	0.0739	0.0968	**0.1035**
	HR@5	0.1865	0.2255	0.2477	0.2469	0.2480	0.2214	0.2444	**0.2518**
	HR@10	0.2686	0.3422	0.3494	0.3663	0.3613	0.3431	0.3599	**0.3688**
	NDCG@5	0.1340	0.1638	0.1689	0.1736	0.1779	0.1733	0.1776	**0.1891**
	NDCG@10	0.1604	0.2051	0.1985	0.2113	0.2128	0.2060	0.2115	**0.2167**
	MRR	0.1516	0.1750	0.1706	0.1829	0.1893	0.1799	0.1896	**0.1954**
Taobao	HR@1	0.1947	0.3381	0.3439	0.3562	0.3510	0.3204	0.3255	**0.3688**
	HR@5	0.4489	0.6077	0.6035	0.6085	0.6159	0.6220	0.6478	**0.6816**
	HR@10	0.5933	0.7084	0.7189	0.7224	0.7371	0.7620	0.7517	**0.8068**
	NDCG@5	0.3256	0.4731	0.4866	0.5005	0.5101	0.4779	0.4814	**0.5339**
	NDCG@10	0.3723	0.5089	0.5139	0.5174	0.5199	0.5233	0.5476	**0.5745**
	MRR	0.3224	0.4405	0.4617	0.4744	0.4818	0.4615	0.4988	**0.5121**
Tmall	HR@1	0.3447	0.3386	0.3501	0.3588	0.3622	0.3634	0.3669	**0.3770**
	HR@5	0.5594	0.5636	0.5727	0.5712	0.5819	0.7310	0.7331	**0.7381**
	HR@10	0.6554	0.6562	0.6646	0.6662	0.6686	0.8378	0.8373	**0.8479**
	NDCG@5	0.4589	0.4654	0.4784	0.4768	0.4843	0.5594	0.5565	**0.5693**
	NDCG@10	0.4901	0.4986	0.5080	0.5074	0.5124	0.5942	0.5951	**0.6051**
	MRR	0.4527	0.4669	0.4741	0.4730	0.4778	0.5256	0.5259	**0.5363**

As shown in Table 2, Tmall and Taobao has a lot more items than CCMR, which makes ranking items more difficult in Tmall and Taobao. So it is more important on these two datasets to take item-side sequence into consideration because it gives the models more information than just using single user-side sequence.

DEEMS performs better than RRN in many cases which means the two player game of dual sequence in DEEMS is effective and show the potential of finer design on dual modeling.

Influence from the size of (co-)interaction set. We vary the size of (co-)interaction set to further investigate the robustness of SCoRe. For simplicity, we set interaction and co-interaction set to have same size in dual side. The results on Tmall and Taobao dataset are shown in Figure 5. We find that when size increases, the performance is improved at first because that the larger the size is, the more information it contains. And when the size continues to increase, the performances begin to drop which indicates that too much noise and useless information is introduced.

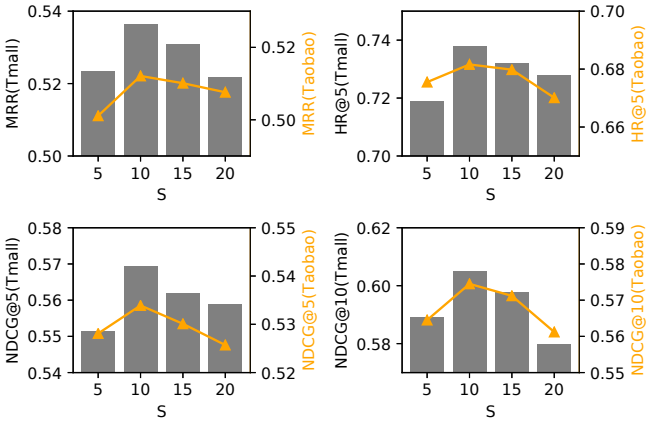

Figure 5: Performance comparision on different size S of (co-)interaction set on Tmall and Taobao dataset.

4.3 Ablation Study: RQ2

In this section, we conduct some ablation studies to investigate the effectiveness of three important components of SCoRe : (1) Interactive Attention Mechanism in dual sequence modeling; (2) Co-Attention Network for cross neighbor relation mining and aggregation; (3) The consideration of using both user-side and item-side.

We set four comparative settings, and the performances of them have been shown in Table 4. The details of the four settings are listed as below.

Table 4: Performance comparison of ablation study.

Dataset	Metric	models				
		RIA	RCA	User	Item	SCoRe
CCMR	HR@10	0.3667	0.3461	0.3491	0.3218	**0.3688**
	NDCG@10	0.2098	0.2077	0.2012	0.1892	**0.2167**
	MRR	0.1872	0.1795	0.1782	0.1567	**0.1954**
Taobao	HR@10	0.7888	0.7702	0.7678	0.7453	**0.8068**
	NDCG@10	0.5436	0.5286	0.5192	0.5001	**0.5745**
	MRR	0.4785	0.4669	0.4652	0.4495	**0.5121**
Tmall	HR@10	0.8467	0.8406	0.8355	0.8122	**0.8479**
	NDCG@10	0.6030	0.5903	0.5843	0.5671	**0.6051**
	MRR	0.5352	0.5289	0.5192	0.5085	**0.5363**

- **RIA** (Remove Interactive Attention) removes the attention part described in 3.2.2 and set the final sequential representations of target user and item as $\boldsymbol{r}_u = h_T^u$ and $\boldsymbol{r}_v = h_T^v$ of Eq. (23).
- **RCA** (Remove Co-Attention) removes the co-attention network and uses simply sum pooling to aggregate neighbor information in interaction and co-interaction set.

- **User** only uses the user-side sequence to do final prediction, as $\hat{y} = f(\boldsymbol{r}_u, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\Theta})$.
- **Item** only uses the item-side sequence to do final prediction, as $\hat{y} = f(\boldsymbol{r}_v, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\Theta})$.

Except for the changes mentioned above, the other parts of the models and experimental settings remain identical to ensure the fairness of comparison.

From Table 4 we can find that (1) SCoRe performs the best indicating the efficacy of different components of the model. (2) the performance decreases more when removing co-attention than interactive attention which means the cross neighbor relation modeling is more important and fundamental to SCoRe's performance. (3) Using single sequence hurt the performance badly and item-side is harder to model compared to user-side thus have worse performance.

4.4 Case Study: RQ3

In this section, we further investigate what patterns SCoRe captures by studying and visualizing a specific case sampled from the CCMR dataset. In Figure 6, we plot the prediction of user-item pair (u36, m1911) where m1911 is the movie *American Sniper*. The ground truth is $y = 1$, we plot the Caser and SCoRe predictions which are $\hat{y}_{\text{caser}} = 0.312$ and $\hat{y}_{\text{SCoRe}} = 0.891$ respectively. By looking into the user behavior sequence, we find the reason of SCoRe's better prediction result.

The user's recent behaviors are favouring comedy, cartoon or fiction, as illustrated in the upper part of Figure 6 which are not very relevant to the target item *American Sniper* which is a biographical and action movie. So it is natural that models like Caser tend to give lower prediction score.

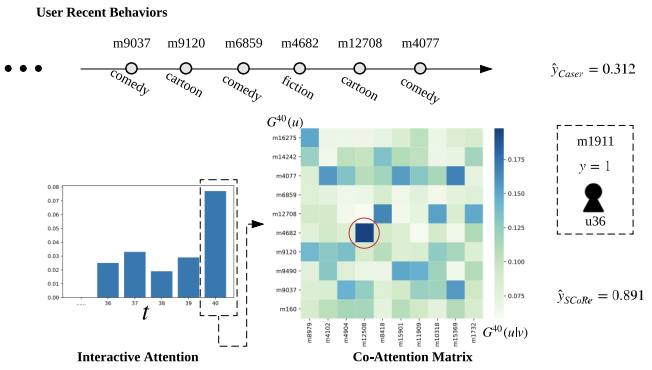

Figure 6: Case study of Caser and SCoRe. We plot user u36's recent behaviors, interactive attention values and co-attention matrix C_{40}^{item} of time slice 40 of SCoRe.

However, we plot the interactive attention value of recent time slices of SCoRe and find the 40-th slice have higher attention value. So we further plot the co-attention matrix of this time slice C_{40}^{item}. We find m4682 (*Interstellar*, fiction and adventure) of $\boldsymbol{G}^{40}(u)$ has high relation with m12508 (*007:Casino Royal*, action and fiction) of $\boldsymbol{G}^{40}(u|v)$. Movie m12508 is more relevant to the user's interest, and its representation is aggregated to the item-side, so it is reasonable that SCoRe has the ability to give the target item higher predicted probability score, which has precisely been interacted by the target user in the test data.

5 RELATED WORK

5.1 Collaborative Filtering

In recommender system literatures, the most widely used method is collaborative filtering [8], which learns from the historical user-item interactions without exogenous information about items or users. It recommends according to the modeled user preferences, e,g., clicks [1, 23] and ratings [19], over the items. Many works [20, 30, 40, 42] have been proposed based on collaborative filtering. Among them, latent factor models play a key role in CF methods, ranging from pLSA [15] and Latent Dirichlet Allocation [3] to SVD-based models [5, 18] and factorization machines [27]. Nowadays, deep neural network (DNN) has attracted huge attentions in recommender systems because of its effective feature extraction and end-to-end model training with satisfying generalization [41]. Some DNN methods [11, 12, 23] are proposed for latent factor collaborative filtering. However, almost all of these approaches, either conventional matrix factorization methods or deep models, lack of temporal pattern mining.

5.2 Sequential Recommendation

Recently, sequential recommendation has drawn huge attention since the sequences of user behaviors have rich information for the user interests, especially for concept drifting [35], long-term behavior dependency [19, 24], periodic patterns [26], etc.

There are three categories for sequential recommendation. The first one is from the view of temporal collaborative filtering [19] with the consideration of drifting user preferences. The second stream is based on Markov-chain methodology [9, 10, 28] which implicitly models the user state dynamics and derive the outcome behaviors. The third school is based on deep neural networks, such as recurrent neural networks (RNNs) [2, 13, 14, 16, 21, 34, 36] and convolutional neural networks (CNNs) regarding the behavior history as an image [17, 32]. However, most of these methods only care about user's interest drifting and do not consider the sequential patterns of items, which also deliver rich information for user-item matching. Models like [36, 37] considers both sequences but in a relatively independent way, which leaves space for finer design of dual sequence modeling. Furthermore, most of these sequential models only care about user's own interaction history while ignoring the information that could be found in similar users or items. And thus the sequential models may suffer from narrowness of recommendation.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose SCoRe, a model that utilizes and aggregates high-order collaborative information using cross neighbor modeling to improve representation learning and collaborative relation mining. Furthermore, we propose an interactive attention mechanism to model the user-side and item-side sequences. In this way, dual sequence modeling captures temporal dynamics from both user and item-side and significantly facilitate final recommendation performance.

For the future work, we plan to further investigate on the time segmentation strategy of the evolving sequential interactions and its influence to the recommendation performance. We also seek to deploy our method on the real-world recommender systems.

Acknowledgement. The corresponding author Weinan Zhang thanks the support of National Natural Science Foundation of China (61702327, 61772333, 61632017) and Shanghai Sailing Program (17YF1428200).

REFERENCES

[1] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. 2009. Spatio-temporal models for estimating click-through rate. In *WWW*.

[2] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H Chi. 2018. Latent Cross: Making Use of Context in Recurrent Recommender Systems. In *WSDM*.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[4] Xuezhi Cao, Weiyue Huang, and Yong Yu. 2016. A Complete & Comprehensive Movie Review Dataset (CCMR). In *SIGIR*. ACM, 661–664.

[5] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. 2012. SVDFeature: a toolkit for feature-based collaborative filtering. *Journal of Machine Learning Research* 13, Dec (2012), 3619–3622.

[6] Weiyu Cheng, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. 2018. DELF: A Dual-Embedding based Deep Latent Factor Model for Recommendation.. In *IJCAI*. 3329–3335.

[7] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*.

[8] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[9] Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley. 2016. Vista: a visually, socially, and temporally-aware model for artistic recommendation. In *RecSys*.

[10] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *ICDM*.

[11] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer product-based neural collaborative filtering. *arXiv preprint arXiv:1808.03912* (2018).

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

[13] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. *CIKM* (2018).

[14] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. *ICLR* (2016).

[15] Thomas Hofmann. 2004. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 89–115.

[16] How Jing and Alexander J Smola. 2017. Neural survival recommender. In *WSDM*.

[17] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. *ICDM* (2018).

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

[19] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *KDD*.

[20] Yehuda Koren, Robert Bell, Chris Volinsky, et al. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[21] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-Aware Sequential Recommendation. In *ICDM*.

[22] Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2011. Rt to win! predicting message propagation in twitter. In *Fifth International AAAI Conference on Weblogs and Social Media*.

[23] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *ICDM*.

[24] Kan Ren, Jiarui Qin, Yuchen Fang, Weinan Zhang, Lei Zheng, Weijie Bian, Guorui Zhou, Jian Xu, Yong Yu, Xiaoqiang Zhu, et al. 2019. Lifelong Sequential Modeling with Personalized Memorization for User Response Prediction. *SIGIR*.

[25] Kan Ren, Weinan Zhang, Ke Chang, Yifei Rong, Yong Yu, and Jun Wang. 2018. Bidding Machine: Learning to Bid for Directly Optimizing Profits in Display Advertising. *TKDE* (2018).

[26] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2018. RepeatNet: A Repeat Aware Neural Recommendation Machine for Session-based Recommendation. *arXiv preprint arXiv:1812.02646* (2018).

[27] Steffen Rendle. 2010. Factorization machines. In *ICDM*.

[28] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*.

[29] Marian-Andrei Rizoiu, Swapnil Mishra, Quyu Kong, Mark Carman, and Lexing Xie. 2018. SIR-Hawkes: Linking epidemic models and Hawkes processes to model diffusions in finite populations. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 419–428.

[30] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization.. In *Nips*, Vol. 1. 2–1.

[31] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. 2019. Session-based social recommendation via dynamic graph attention networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 555–563.

[32] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[34] Kiewan Villatel, Elena Smirnova, Jérémie Mary, and Philippe Preux. 2018. Recurrent Neural Networks for Long and Short-Term Sequential Recommendation. *KDD* (2018).

[35] Gerhard Widmer and Miroslav Kubat. 1996. Learning in the presence of concept drift and hidden contexts. *Machine learning* 23, 1 (1996), 69–101.

[36] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *WSDM*.

[37] Qitian Wu, Yirui Gao, Xiaofeng Gao, Paul Weng, and Guihai Chen. 2019. Dual Sequential Prediction Models Linking Sequential Recommendation and Information Dissemination. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 447–457.

[38] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. 2019. Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems. In *The World Wide Web Conference*. ACM, 2091–2102.

[39] Sai Wu, Weichao Ren, Chengchao Yu, Gang Chen, Dongxiang Zhang, and Jingbo Zhu. 2016. Personal recommendation using deep recurrent neural networks in NetEase. In *ICDE*.

[40] Diyi Yang, Tianqi Chen, Weinan Zhang, Qiuxia Lu, and Yong Yu. 2012. Local implicit feedback mining for music recommendation. In *RecSys*. ACM, 91–98.

[41] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep learning based recommender system: A survey and new perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[42] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *SIGIR*. ACM, 785–788.

[43] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2018. Deep Interest Evolution Network for Click-Through Rate Prediction. *arXiv preprint arXiv:1809.03672* (2018).

[44] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *KDD*.

[45] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *KDD*.