Lifelong Sequential Modeling with Personalized Memorization for User Response Prediction

Kan Ren[†], Jiarui Qin[†], Yuchen Fang[†], Weinan Zhang[†], Lei Zheng[†],

Weijie Bian[‡], Guorui Zhou[‡], Jian Xu[‡], Yong Yu[†], Xiaoqiang Zhu[‡] and Kun Gai[‡*]

[†]Shanghai Jiao Tong University, [‡]Alibaba Inc.
{kren, qinjr, arthur_fyc, wnzhang}@apex.sjtu.edu.cn, {weijie.bwj, guorui.xgr, xiaoqiang.zxq}@alibaba-inc.com

ABSTRACT

User response prediction, which models the user preference w.r.t. the presented items, plays a key role in online services. With two-decade rapid development, nowadays the cumulated user behavior sequences on mature Internet service platforms have become extremely long since the user's first registration. Each user not only has intrinsic tastes, but also keeps changing her personal interests during lifetime. Hence, it is challenging to handle such *lifelong sequential modeling* for each individual user. Existing methodologies for sequential modeling are only capable of dealing with relatively recent user behaviors, which leaves huge space for modeling long-term especially lifelong sequential patterns to facilitate user modeling. Moreover, one user's behavior may be accounted for various previous behaviors within her whole online activity history, i.e., long-term dependency with multi-scale sequential patterns. In order to tackle these challenges, in this paper, we propose a Hierarchical Periodic Memory Network for lifelong sequential modeling with personalized memorization of sequential patterns for each user. The model also adopts a hierarchical and periodical updating mechanism to capture multi-scale sequential patterns of user interests while supporting the evolving user behavior logs. The experimental results over three large-scale real-world datasets have demonstrated the advantages of our proposed model with significant improvement in user response prediction performance against the state-of-the-arts.

1 INTRODUCTION

Nowadays, accurate prediction of user responses, e.g., clicks or conversions, has become the core part in personalized online systems, such as search engines [11], recommender systems [41] and computational advertising [20]. The goal of user response prediction is to estimate the probability that a user would respond to a specific item or a piece of content provided by the online service. The estimated probability may guide the subsequent decision making of the service provider, e.g., ranking the candidate items according to the predicted click-through rate [41] or performing ad bidding according to the estimated conversion rate [58].

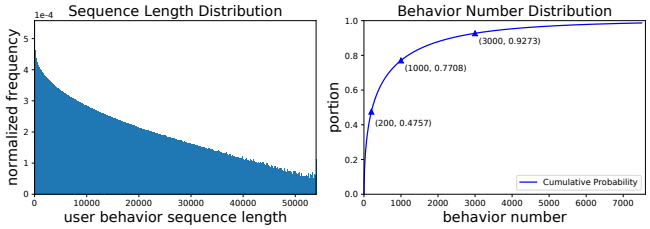

Figure 1: User behavior (click) statistics from Alibaba e-commerce platform during April to September in 2018. Left: the distribution of the user sequence lengths; Right: the number of user behaviors between add-to-cart event and the final conversion.

One key aspect of user response prediction is user modeling, which profiles each user through learning from her historical behavior data or other side information. Generally speaking, the user behavior data have three characteristics. First, the user behaviors not only reflect the intrinsic and multi-facet user interests [25, 28], but also reveal the temporal dynamics of user tastes [29]. Second, as is shown in Figure 1, the length of behavior sequences vary for different users because of diverse activeness or registration time. Third, there exist long-term dependencies in one's behavior history where some behaviors happened early may accounts for the final decision making of the user, as illustrated in the right plot of Figure 1. Moreover, the temporal dependency also shows multi-scale sequential patterns, i.e., various temporal behavior dependencies, of different users.

With two-decade of rapid development of Internet service platforms, there have been abundant user behavior sequences cumulated in online platforms. Many works have been proposed for user modeling [46, 62], especially with sequential modeling [21, 61]. Some of the existing methods for user modeling aggregate the historical user behaviors for the subsequent preference prediction [28, 30]. However, they ignore temporal dynamics of user behaviors [29]. Sequential modeling for user response prediction is to conduct a dynamic user profiling with sequential pattern mining. Some other works [21, 61] aim to deal with temporal dynamics with sequential pattern mining. Nevertheless, these sequential models

*K. Ren, J. Qin and Y. Fang share the co-first authorship. The work was done when they were working as internship in Alibaba Inc.

SIGIR'19, July 21-25, 2019, Paris, France

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

focus only on short-term sequences, e.g., several latest behaviors of the user [61] or the behavior sequence within recent period of time [21] but abandon previous user behaviors.

Considering the situation of recommending items in the manual way. Human may first take one's intrinsic tastes into consideration [56] and then consider her multi-facet interests [25, 28], e.g., various preferences over different item categories. Moreover, it is natural to combine one's long-term [55] and recent experience [22] so as to comprehensively recommend items.

In order to tackle these challenges, also to overcome the shortcomings of the related works, we formulate the *lifelong sequential modeling* framework and propose a novel Hierarchical Periodic Memory Network (HPMN) to maintain user-specific behavior memories to solve it. Specifically, we build a personalized memorization for each user, which remembers both intrinsic user tastes and multi-facet user interests with the learned while compressed memory. Then the model maintains hierarchical memories to retain long-term knowledge for user behaviors. The HPMN model also updates memorization from newly coming user behaviors with different periods at different layers so as to capture multi-scale sequential patterns during her lifetime. The extensive experiments over three large-scale real-world datasets show significant improvements of our proposed model against several strong baselines including state-of-the-art.

This paper has three main contributions listed as follows.

- To the best of our knowledge, it is the first work to propose the lifelong sequential modeling framework, which conducts a unified, comprehensive and personalized user profiling, for user response prediction with extremely long user behavior sequence data.
- In lifelong sequential modeling framework, we propose a memory network with incremental updating mechanism to learn from the retained knowledge of user lifelong data and the evolving user behavior sequences.
- We further design a hierarchical architecture with multiple update periods to effectively mine and utilize the multi-scale sequential patterns in users' lifelong behavior sequences.

The rest of our paper is organized as below. Section 2 presents a comprehensive survey of user response prediction works. Section 3 introduces the motivation and model design of our methodology in detail. The experimental setups with the corresponding results are illustrated in Section 4. We finally conclude this paper and discuss the future work in Section 5.

2 RELATED WORKS

2.1 User Response Prediction

User response prediction is to model the interest of the user on the content from the provider and estimate the probability of the corresponding user event [43], e.g., clicks and conversions. It has become a crucial part of the online services, such as search engines [11, 59], recommender systems [17, 41] and online advertising [15, 20, 62]. Typically, user response prediction is formulated as a binary classification problem with user response likelihood as the training objective [1, 15, 40, 47].

From the view of methodology, linear models such as logistic regression [14, 33] and non-linear models such as tree-based models [20] and factorization machines [38, 40] have been well studied. Recently, neural network models [41, 62] have attracted huge attention.

2.2 Sequential User Modeling

User modeling, i.e. to capture the latent interests of the user and derive the adaptive representation for each user, is the key component in user response prediction [60, 62]. The researchers have proposed many methodologies ranging from latent factor methods [30, 45] to deep representation learning methods [41, 62]. These models aggregate all historical behaviors as a whole while ignoring the temporal and drifting user interests.

Nowadays, sequential user modeling has drawn great attention since the sequences of user behaviors have rich information for the user interests, especially with drifting trends. It has been a research hotspot for sequential modeling in online systems [42, 51, 61]. From the perspective of modeling, there are three categories for sequential user modeling. The first is from the view of temporal matrix factorization [29] with the consideration of drifting user preferences but it heuristically made some assumptions about the behavior patterns. The second stream is based on Markov-chain methodology [18, 19, 46] which implicitly models the user state dynamics and derive the outcome behaviors. The third school is based on deep neural network for its stronger capacity of feature extraction, such as recurrent neural network (RNN) [3, 21, 22, 26, 34, 51, 54] and convolutional neural network (CNN) regarding the behavior history as an image [27, 50].

However, these methods mainly focus on short-term user modeling which has been constrained in the most recent behaviors. Zhang et al. [56] additionally utilized a static user representation for user intrinsic interests along with short-term intent representation. Ying et al. [55] proposed a hierarchical attentional method over a list of user behavior features for modeling long-term interests. But they can only capture simple sequential patterns lacking of considering long-term and multi-scale behavior dependencies. Moreover, few of the existing works consider modeling lifelong user behavior history thus cannot properly establish a comprehensive user profiling.

2.3 Memory-augmented Networks

Memory-augmented networks [16, 23, 32, 49, 53] have been proposed in natural language processing (NLP) tasks for explicitly remembering the extracted knowledge by maintaining that in an external memory component. Several works [5, 12, 24, 52] utilize memory network for recommendation tasks. However, these methods directly use the structure of memory network from NLP tasks, which does not consider practical issues in user response prediction. Specifically, they fail to consider multi-scale knowledge memorization or long-term dependencies. There is one work of recurrent model with multi-scale pattern mining [9] in the NLP field. The essential difference is that their model was designed for natural language sentence modeling with fixed length, while our model supports lifelong sequential modeling through the maintained user memory and additionally consider long-term dependencies within user behavior sequences with extremely large length.

3 METHODOLOGY

In this part, with discussions about the notations and preliminaries of user response prediction, we make a definition of lifelong sequential modeling and discuss some characteristics of it. Then we

Table 1: Notations and descriptions

Notation	Description.
u, v	The target user and the target item.
$y, \hat{y}$	The true label and the predicted probability of user response.
$\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{c}$	Feature of user u, item v and the context information.
$\bar{\boldsymbol{u}}$	The side information of the user.
$\boldsymbol{v}_i$	Feature of the i-th interacted item in user's behavior history.
$\boldsymbol{r}$	The inferred sequential representation of the user.
T, D	The total behavior sequence length and the layer number of HPMN.
i, j	The index of sequential behavior and network layer ($i \in [1, T], j \in [1, D]$).
$\boldsymbol{m}_i^j$	The maintained memory content in the j-th layer at the i-th time step.
w^j, t^j	The reading weight and the period for the j-th memory slot maintained by the j-th layer of HPMN.

present the overall architecture of lifelong sequential modeling including the data flow with Hierarchical Periodic Memory Network (HPMN). The notations have been summarized in Table 1.

3.1 Preliminaries

The data in the online system are formulated as a set of triples $\{(u, v, y)\}$ each of which includes the user $u \in \mathcal{U}$, item $v \in \mathcal{V}$ and the corresponding label of user behavior indicator

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

Without loss of generality, we take click as the user behavior and the goal is to estimate click-through rate[1] (CTR) of user u on item v at the given time. It approaches CTR prediction through a learned function $f_{\Theta}(\cdot)$ with parameter Θ. There are three parts of raw features $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{c})$. Here $\boldsymbol{v}$ is the feature vector of the target item v including the item ID and some side information and $\boldsymbol{c}$ is the context feature of the prediction request such as web page URL. User side feature $\boldsymbol{u} = (\bar{\boldsymbol{u}}, \{\boldsymbol{v}_i\}_{i=1}^{T})$ contains some side information $\bar{\boldsymbol{u}}$ and a sequence of user interacted (i.e., *clicked*) items of user u. Note that, the historical sequence length T varies among different users.

The goal of sequential user modeling is to learn a function $g_{\Phi}(\cdot)$ with parameter Φ for conducting a comprehensive representation for user u

$$\boldsymbol{r} = g(\{\boldsymbol{v}_i\}_{i=1}^{s}; \Phi) \tag{2}$$

taking the recent s user behaviors. Note that, this user modeling can be drifting since the user continues interacting with online systems and generating new behaviors. Many sequential user modeling works set a fixed value $s < T$ as the maximal length of user behavior sequence, e.g., $s = 5$ in [21] for session-based recommendation and $s = 50$ in [61], to capture recent user interests.

Thus the final task of user response prediction is to estimate the probability $\hat{y}$ of user action i.e. click, over the given item as

$$\hat{y} = \Pr(y|\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{c}) = f(\boldsymbol{r}, \boldsymbol{v}, \boldsymbol{c}; \Theta). \tag{3}$$

3.2 Lifelong Sequential Modeling

Recall that, most existing works on sequential user modeling focus on the recent s behaviors, while sometimes $s \ll T$ for the whole user behavior sequence with length T. To the best of our knowledge, few of them consider lifelong sequential modeling. We define it as below.

[1]In this paper, we focus on the CTR estimation, while the estimation of other responses can be done by following the same tokens.

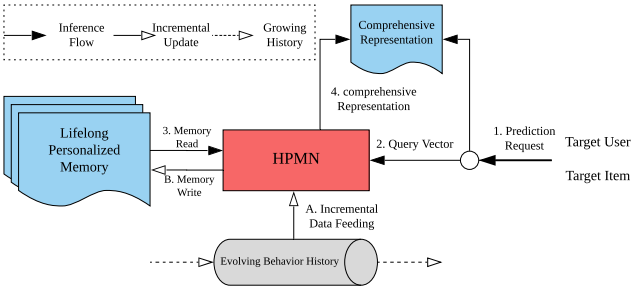

Figure 2: The LSM framework.

Definition. *Lifelong Sequential Modeling (LSM) in user response prediction is a process of continuous (online) user modeling with sequential pattern mining upon the lifelong user behavior history.*

There are three characteristics of LSM.

- LSM supports lifelong memorization of user behavior patterns. It is impossible for the model to maintain the whole behavior history of each user for real-time online inference. Thus it requires highly efficient knowledge preserving of user behavior patterns.
- LSM should conduct a comprehensive user modeling of both intrinsic user interests and temporal dynamic user tastes, for future behavior prediction.
- LSM also needs continuous adaptation to the up-to-date user behaviors.

Following the above principles, we propose a LSM framework for the whole evolving user behavior history, as is illustrated in Figure 2. Within the framework, we conduct a personalized memory with several slots for each user. This memory will be maintained through an incremental updating mechanism (as Steps A and B in the figure) along with the evolving user behavior history.

As for online inference, when a user sends a visit request, the online service will transmit the request including the information of target user and target item. Each user request will trigger a query procedure and we use the vector of the target item $\boldsymbol{v}$ as the query to obtain the associated user representation according to this specific item in the memory pool. Then HPMN model will take the query vector to *read* the lifelong maintained personalized memory of that user, to conduct the corresponding user representation, without inference over the whole historical behavior sequence. After that, the user representation $\boldsymbol{r}$, item vector $\boldsymbol{v}$ and context features $\boldsymbol{c}$ will be together considered for the subsequent user response prediction, which will be described in Section 3.4.

The details of HPMN will be presented in Section 3.3.

3.3 Hierarchical Periodic Memory Network

In this section, we first present the motivations of HPMN model and subsequently discuss the specific architectures.

Generally speaking, we propose HPMN model based on three considerations of the motivation.

- As is stated above, the main goal of LSM is to capture sequential user patterns hidden in user behavior sequences. Many works [18, 22, 61] have been proposed for sequential pattern mining to improve the subsequent prediction. Thus HPMN model firstly introduces sequential modeling through a recurrent component.

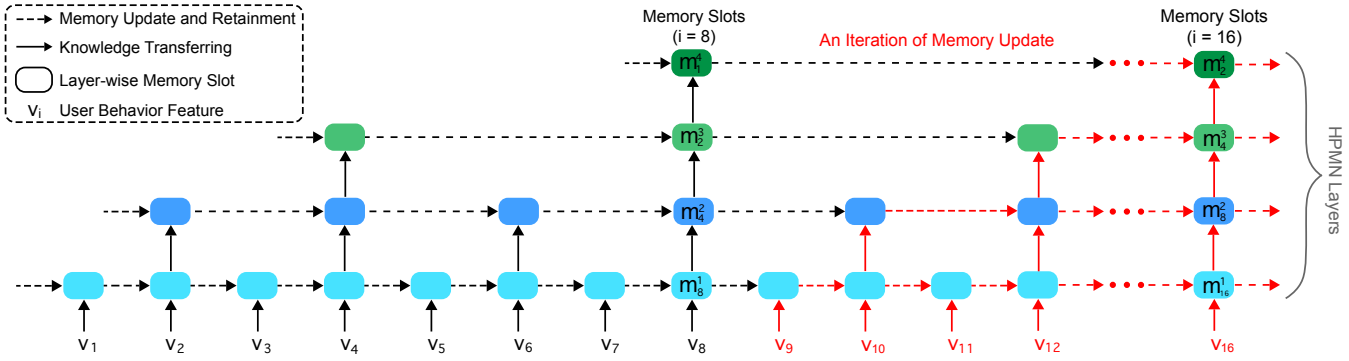

Figure 3: The framework of HPMN model with four layers maintaining user memory in four ($D = 4$) memory slots. The update period t^j of j-th layer follows an exponential sequence $\{2^{j-1}\}_{j=1}^{D}$ as an example. The red part means the incremental updating mechanism; the dotted line means the periodic memorization and forgetting.

- There also exists long-term dependencies among lifelong user behaviors, i.e., the later user decision making may have some relationship to her previous actions. We will show some examples in the experiment of the paper. However, traditional sequential modeling methods either rely on the recent user behaviors, or updates user states too frequently which may result in memorization saturation and knowledge forgetting [48]. Hence we incorporate periodic memory updating mechanism to avoid unexpected knowledge drifting.
- The behavior dependencies may span various time distances, e.g., user may show preferences on the specific item at different time along the whole history. So that it requires multi-scale sequential pattern mining. HPMN model deals with this by maintaining hierarchical memory slots with different update periods.

Moreover, since the personalized memory stores a comprehensive understanding of each user with multi-facet user preferences, so HPMN model incorporates a regularization of memory covariance to preserve diverse knowledge of user interests. Besides, for each query, the model reads the user memory through an attentional way which tries to match the target item over the multi-facet user modeling knowledge.

Next we will describe the model details from four aspects. The memory architecture will be introduced in Section 3.3.1 followed by the description of periodic yet incremental updating mechanism in Section 3.3.2. We introduce the usage of the user memory in Section 3.3.3 and the covariance regularization in Section 3.3.4.

3.3.1 Hierarchical Memory for Sequential Modeling. As is illustrated in Figure 2, for each user u, there is a user-specific memory pool containing D memory slots $\{\boldsymbol{m}^j\}_{j=1}^{D}$ and $\boldsymbol{m}^j \in \mathbb{R}^p$ is a piece of real-value representation of user modeling. The idea of the external memory has been used in the NLP field [32, 39] for better memorization of the context information embedded in the previously consumed paragraph. We utilize this external memory pool for capturing the intrinsic user interests with temporal sequential patterns, yet it is also evolving and supports incremental memory update along with the growing behavior sequences.

Generally speaking, HPMN model is a layer-wise memory network which contains D layers, as is shown in Figure 3. Each layer maintains the specific memory slot $\boldsymbol{m}^j$. The output $\boldsymbol{m}_i^j$ of the j-th layer at the i-th time step (i.e., i-th sequential user behavior) will be transmitted not only to the next time step, but also to the next layer at the specific time step.

3.3.2 Continuous Memory Update. Considering the rapidly growing user-item interactions, it is impossible for the model to scan through the complete historical behavior sequence at each prediction time. That's the reason why almost all the existing methods only consider recent short-term user behaviors. Thus it is necessary to maintain only the latest memories and implement an incremental update mechanism in real time. After each user behavior on a item at the i-th time step, the memory slot at each layer would be updated as

$$\boldsymbol{m}_i^j = \begin{cases} g^j\left(\boldsymbol{m}_i^{j-1}, \boldsymbol{m}_{i-1}^j\right) & \text{if } i \bmod t^j = 0\,, \\ \boldsymbol{m}_{i-1}^j & \text{otherwise}\,, \end{cases} \tag{4}$$

where $j \in [1, D]$ and t^j is the update period of j-th layer. In Eq. (4), the memory writing in each layer is based on the Gated Recurrent Unit (GRU) [8] cell g^j as

$$\begin{aligned} \boldsymbol{z}_i^j &= \sigma(\overline{\boldsymbol{W}}_z^j \boldsymbol{m}_i^{j-1} + \overline{\boldsymbol{U}}_z^j \boldsymbol{m}_{i-1}^j + \overline{\boldsymbol{b}}_z^j) \\ \boldsymbol{r}_i^j &= \sigma(\overline{\boldsymbol{W}}_r^j \boldsymbol{m}_i^{j-1} + \overline{\boldsymbol{U}}_r \boldsymbol{m}_{i-1}^j + \overline{\boldsymbol{b}}_r^j) \\ \boldsymbol{m}_i^j &= (1 - \boldsymbol{z}_i^j) \odot \boldsymbol{m}_{i-1}^j \\ &\quad + \boldsymbol{z}_i^j \odot \tanh(\overline{\boldsymbol{W}}_m^j \boldsymbol{m}_i^{j-1} + \overline{\boldsymbol{U}}_m(\boldsymbol{r}_i^j \odot \boldsymbol{m}_{i-1}^j) + \overline{\boldsymbol{b}}_m^j)\,. \end{aligned} \tag{5}$$

For each cell in different layers, the parameters $(\overline{\boldsymbol{W}}^j, \overline{\boldsymbol{U}}^j, \overline{\boldsymbol{b}}^j)$ of g^j differs. Note that, it is a soft-writing operation on the memory slot $\boldsymbol{m}$ since the last operation of g^j function has the "erase" vector $\boldsymbol{z}^j$ as the same as that in the other memory network literature such as Neural Turing Machine (NTM) [16]. Note that the first layer of memory $\boldsymbol{m}_i^1$ will be updated with the raw feature vector $\boldsymbol{v}_i$ of the user interacted item and the memory contents from the last time step $\boldsymbol{m}_{i-1}^1$.

Moreover, the memory update is periodic where each memory $\boldsymbol{m}^j$ at j-th memory slot will be updated according to the time step i and the period t^j of each layer. Here we set the period of each layer t^j as the hyperparameter which is reported in Table 3 in the experimental setup. By applying this periodic updating mechanism,

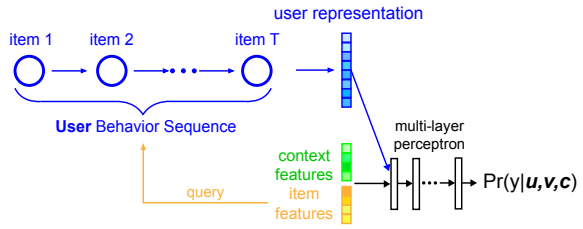

Figure 4: The overall user response prediction.

the upper layers are updated less frequently to achieve two goals. (i) First it avoids gradient vanishing or explosion, thus being able to model long sequences better; (ii) It then remembers the long-term dependency better than the memory maintained by the lower layer. The different update behaviors of each layer may capture multi-scale sequential patterns, which is illustrated in Section 4.3.

The similar idea of clockwork update has been implemented in RNN model [31]. However, they simply split the parameters in the recurrent cell and update the hidden states separately. We make two improvements that (i) we connect the network layers through state transferring so as to make layer-wise information transmitting; (ii) we incorporate the external memory component to preserve both intrinsic and multi-scale sequential patterns for lifelong sequential modeling.

3.3.3 Attentional Memory Reading. Till now, the model has conducted the long-term memorization of the intrinsic and multi-scale temporal dynamics, which may connect the intrinsic properties and the multi-scale patterns of behavior dependency, to the current user response prediction. Besides, we conduct the attentional memory usage similar to the common memory networks [16, 49, 53].

We calculate the comprehensive user representation $\boldsymbol{r}$ as

$$\boldsymbol{r} = \sum_{j=1}^{D} w^j \cdot \boldsymbol{m}^j \ . \tag{6}$$

Here $\boldsymbol{m}^j$ is the maintained memory at the last time step of the long-term sequence, i.e., $i = T$ and T is the final behavior log of the user. The weight of each memory w^j means the contribution of each memory slot to the final representation $\boldsymbol{r}$ and it is calculated as

$$w^j = \frac{\exp(e^j)}{\sum_{k=1}^{D} \exp(e^k)} \ , \tag{7}$$

where

$$e^j = E(\boldsymbol{m}^j, \boldsymbol{v}) \tag{8}$$

is an energy model which measures the relevance between the query vector $\boldsymbol{v}$ and the long-term memory $\boldsymbol{m}^j$. Note that the energy function E is a nonlinear multi-layer deep neural network with Rectifier (Relu) activation function $\text{Relu}(x) = \max(0, x)$. The way we calculate the attention through the energy function E is similar to that in the NLP field [2].

3.3.4 Memory Covariance Regularization. As is described in the previous sections, the maintained user memory captures long-term sequential patterns with multi-facet user interests. Recall that our model uses D memory slots with p dimensions to memorize user behavior patterns. We expect that different memories store knowledge of user interests from different perspectives. However, unlike the models like NTM [16], HPMN does not utilize attention mechanism to reduce redundancy when updating memory slots. In order to facilitate memorization utility, we utilize a covariance regularization on memories following [10].

Specifically, we first define $\boldsymbol{C}$ as the covariance matrix of the memory contents as

$$\boldsymbol{C} = \frac{1}{p}(\boldsymbol{M} - \overline{\boldsymbol{M}})(\boldsymbol{M} - \overline{\boldsymbol{M}})^\top \tag{9}$$

where

$$\boldsymbol{M} = [\boldsymbol{m}^1, ..., \boldsymbol{m}^j, ..., \boldsymbol{m}^D]^\top \tag{10}$$

is the matrix of memories and $\overline{\boldsymbol{M}}$ is the mean matrix with regard to each row of $\boldsymbol{M}$ and p is the dimension of each memory slot. Note that $\overline{\boldsymbol{M}}$ has the same shape with $\boldsymbol{M}$. After that, we define the loss $\mathcal{L}_c$ to regularize the covariance as

$$\mathcal{L}_c = \frac{1}{2}(\|C\|_F^2 - \|\text{diag}(C)\|_2^2) \tag{11}$$

where $\|\cdot\|_F$ is the Frobenius norm of matrix. We need to minimize covariance between different memory slots, which corresponds to penalizing the norm of $\boldsymbol{C}$.

3.4 Prediction Function and Losses

For each prediction request, we obtain the comprehensive representations $\boldsymbol{r}$ through querying the personalized memory for the target user by Eqs. (2) and (6). The final estimation for the user response probability will be calculated as that in Figure 4 as

$$\hat{y} = f(\boldsymbol{r}, \boldsymbol{v}, \boldsymbol{c}; \boldsymbol{\Theta}) \ , \tag{12}$$

where f is implemented as a multi-layer deep network with three layers, whose widths are 200, 80 and 1 respectively. The first and second layer use ReLU as activation function while the third layer uses sigmoid function as $\text{Sigmoid}(x) = \frac{1}{1+e^{-x}}$.

As for the loss function, we take an end-to-end training and introduce (i) the widely used cross entropy loss [43, 61, 62] $\mathcal{L}_{\text{ce}}$ over the whole dataset with (ii) the covariance regularization $\mathcal{L}_c$ and (iii) the parameter regularization $\mathcal{L}_r$. We utilize gradient descent for optimization. Thus the final loss function is

$$\begin{aligned} \arg\min_{\boldsymbol{\Theta}, \boldsymbol{\Phi}} &= \mathcal{L}_{\text{ce}} + \lambda \mathcal{L}_c + \mu \mathcal{L}_r \\ &= -\sum_{k=1}^{N} \left[y_k \log \hat{y}_k + (1 - y_k) \log(1 - \hat{y}_k) \right] \\ &+ \frac{1}{2}\lambda \left(\|C\|_F^2 - \|\text{diag}(C)\|_2^2 \right) + \frac{1}{2}\mu \left(\|\boldsymbol{\Theta}\|_2^2 + \|\boldsymbol{\Phi}\|_2^2 \right) \ , \end{aligned} \tag{13}$$

where λ and μ are the weights of the two regularization losses, $\boldsymbol{\Phi} = \{(\overline{\boldsymbol{W}}^j, \overline{\boldsymbol{U}}^j, \overline{\boldsymbol{b}}^j)\}_{j=1}^{D}$ is the set of model parameters of HPMN and N is the size of training dataset.

Discussions. We propose the lifelong sequential user modeling with the personalized memory for each user. The memory are updated periodically to capture long-term yet multi-scale sequential patterns of user behavior. For user response prediction, the maintained user memory will be queried with the target item to forecast the user preference over that item.

Note that, LSM has some essential differences from the lifelong machine learning (LML) proposed by [7]. First, the retained knowledge in LSM is user-specific while LML is model-specific; Second,

LSM is conducted for user modeling while LML aims at continuously multi-task learning [6]; Finally the user behavior patterns drift in LSM while the data samples and tasks change in LML.

The retained while compressed memory guarantees that the time complexity of our model is acceptable for industrial productions. The personalized memory will be created from the first registration of the user and maintained by HPMN model as lifelong modeling. For each prediction, the model only needs to query the maintained memory, rather than inferring over the whole behavior sequence as adopted by the other related works [22, 61]. Meanwhile, our model has an advantage of sequential behavior modeling to those aggregation-based model, such as traditional latent factor models [28, 29]. For memory updating, the time complexity is $O(DC)$ where C is the calculation time of the recurrent component. All the matrix operations can be parallelly executed on GPUs.

The model parameters of HPMN can be updated in a normal way as common methods [41, 62] where the model is retrained periodically depending on the specific situations. The number of memory slots D is the hyperparameter and the specific slot number depends on the practical situation. Along with the lifelong sequential user modeling, the memory of each user is expanded accordingly. We conduct an experiment about the relations between the number of memory slots and the task performance and discuss in Section 4.3. We may follow [48] and expand the memory when the performance drops in some margin. However, we only need to add one layer with a larger updating period on the top, without retraining all the parameters of HPMN as that in [48].

4 EXPERIMENTS

In this section, we present the details of the experiment setups and the corresponding results. We also make some discussions with an extended investigation to illustrate the effectiveness of our model. Moreover, we have also published our code[2].

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

- **RQ1** Does the incorporation of lifelong behavior sequence contributes to the final user response prediction?
- **RQ2** Under the comparable experimental settings, does HPMN achieve the best performance?
- **RQ3** What patterns does HPMN capture from user behavior sequences? Does it have the ability to capture long-term, short-term and multi-scale sequential patterns?

4.1 Experimental Setups

In this part, we present the experiment setups including dataset description, preprocessing method, evaluation metrics, experiment flow and the discussion of the compared settings.

4.1.1 Datasets. We evaluate all the compared models over three real-world datasets. The statistics of the three datasets are shown in Table 2.

Amazon [37] is a collection of user browsing logs over e-commerce products with reviews and product metadata from Amazon Inc. We use the subset of Electronic products which contains user behavior logs from May 1999 to July 2014. Moreover, we regard all the user reviews as user click behaviors. This processing method has been widely used in the related works [61, 62].

[2]Reproducible code link: https://github.com/alimamarankgroup/HPMN.

Table 2: The dataset statistics. T: length of the whole lifelong sequence (maximal length in the dataset). s: length of recent behavior sequence.

Dataset	Amazon	Taobao	XLong
User #	192,403	987,994	20,000
Item #	63,001	4,162,024	3,269,017
s	10	44	232
T	100	300	1,000

Taobao [63] is a dataset of user behaviors from the commercial platform of Taobao. The dataset contains several types of user behaviors including click, purchase, add-to-cart and item favoring. It is consisted of user behavior sequences from nearly one million users from November 25 to December 3, 2017.

XLong is sampled from the click logs of more than twenty thousand users on Alibaba e-commerce platform from April to September 2018. It contains relatively longer historical behavior sequences than the other two datasets. Note that there is no public dataset containing such long behavior history of each user for sequential user modeling. We have published this dataset for further research[3].

Dataset Properties. These datasets are selected as typical examples in real-world applications. **Amazon** dataset covers a very long time range of user behaviors during about fifteen years while some of the users were inactive and generated relatively sparse behaviors during this long time range. For **XLong** dataset, each user has a behavior sequence of one thousand clicks that happened in half a year. And modeling such long sequence is a major challenge for lifelong sequential modeling. As for **Taobao** dataset, although it only covers nine days' logs, the users in it have generated quite a few behaviors which reflects that the users are quite active.

Dataset Preprocessing. To simulate the environment of lifelong sequential modeling, for each dataset, we sort the behaviors of each user by the timestamp to form the lifelong behavior sequence for each user. Assuming there are T behaviors of user $\boldsymbol{u}$, we use this behavior sequence to predict the user response probability at the target item for the $(T+1)$-th behavior. Note that 50% target items at the prediction time in each dataset have been replaced with another item from the non-clicked item set for each user, to build the negative samples.

Training & Test Splitting. We split the training and test dataset according to the timestamp of the prediction behavior. We set a cut time within the time range covered by the full dataset. If the prediction behavior of a sequence took place before the cut time, the sequence is put into the training set. Otherwise it would be in the test set. In this way, training set is about 70% of the whole dataset and test set is about 30%.

4.1.2 Evaluation Metrics. We use two measurements for the user response prediction task. The first metric is area under ROC curve (**AUC**) which assesses the pairwise ranking performance of the classification results between the clicked and non-clicked samples. The other metric is **Log-loss** calculated as

$$\text{Log-loss} = \sum_{k=1}^{N} \left[-y_k \log \hat{y}_k - (1-y_k) \log(1-\hat{y}_k) \right] . \tag{14}$$

[3]Dataset download link: https://tianchi.aliyun.com/dataset/dataDetail?dataId=22482.

Here N is the number of samples in the test set. Log-loss is to measure the overall likelihood of the whole test data and has been widely used for the classification tasks [41, 44].

4.1.3 Experiment Flow. Recall that each sample of user behaviors contains at most T interacted items. As some of our baseline models were proposed to model recent short behavior sequence, thus we first split the recent s user behaviors as the short-term sequential data for baseline model evaluation ($s < T$), as is shown in Table 2. Moreover, for fair comparison, we also conduct the experiments over the whole lifelong sequences with length T for all the baselines.

Note that, all the compared models are fed with the same features including contextual features and side information for fair comparison.

Finally, we conduct the **significance test** to verify the statistical significance of the performance improvement of our model against the baseline models. Specifically, we deploy a MannWhitney U test [36] under AUC metric, and a t-test [4] under Log-loss metric.

4.1.4 Compared Settings. To show the effectiveness of our method, we compare it with three groups of eight baselines. The first group consists of aggregation-based models, they aggregate the user behaviors for user modeling and response prediction, without considering the sequential patterns.

DNN is a multi-layer feed-forward deep neural network which has been widely used as the base model in recent works [41, 57, 62]. We follow [62] and use sum pooling operation to integrate all the sequential behavior features concatenating the other features as the user representation.

SVD++ [28] is a MF-based model that combines the user clicked items and latent factors for response prediction.

The second group contains short-term sequential modeling methods including RNN-based models, CNN-based models and a memory network model. For these methods, they either use the behavior data within a session or just truncate the recent behavior sequence to the fixed length.

GRU4Rec [22] bases on RNN and it is the first work using recurrent cell to model sequential user behaviors. It is originally proposed for session-based recommendation.

Caser [50] is a CNN based model, using horizontal and vertical convolutional filters to capture behavior patterns at different scales.

DIEN [61] is a two-layer RNN structure with attention mechanism. It uses the calculated attention values to control the second RNN layer to model drifting user interests.

RUM [5] is a memory network model which uses an external memory following the similar architecture in NLP tasks [16, 39] to store user's behavior features. We implement feature-level RUM as it performed best in the paper [5].

The third group is formed of some long-term sequential modeling methods. However, note that, our HPMN model is the first work on the lifelong sequential modeling for user response prediction.

LSTM [23] is the first model to do long-term sequential modeling whose memory capacity is limited.

SHAN [55] is a hierarchical attention network. It uses two attention layers to handle user's long- and short-term sequences, respectively. However, this model does not capture sequential patterns.

HPMN is our proposed model described in Section 3.

We first evaluate the models in the second group over the short length data as they were proposed for short-term sequential modeling. Then we test all the models over the whole length data comparing to our proposed model.

Some state-based user models [19, 46] have been compared in [50] thus we just compare with state-of-the-art [50]. We omit comparison to the other memory-based models [13, 24] since they are not aiming at sequential user modeling.

For online inference, all of the baselines except memory models, i.e., RUM and HPMN, need to load the whole user behavior sequence to further conduct user modeling for response prediction, while the memory-based models only need to read the user's personalized memory contents for the subsequent prediction. Thus the space utility is more efficient of memory-based model considering online sequential modeling.

The difference between our model and the other memory network model, i.e., RUM, is two-fold. (i) RUM implements the memory architecture following [39] in NLP tasks, which may not be appropriate for user response prediction, since the user generated data are quite different to language sentences. And the experiment results in the below section also reflect this. (ii) Our model utilizes periodic updated memories through hierarchical network to capture multi-scale sequential patterns while RUM has no consideration of that.

4.1.5 Hyperparameters. There are two sets of hyperparameters. The first set is training hyperparameters, including learning rate and regularization weight. We consider learning rate from $\{1 \times 10^{-4}, 5 \times 10^{-3}, 1 \times 10^{-3}\}$ and regularization weight λ and μ from $\{1 \times 10^{-3}, 1 \times 10^{-4}, 1 \times 10^{-5}\}$. Batch size is fixed on 128 for all the models. The hyperparameters of each model are tuned and the best performances have been reported below. The second group is the structure hyperparameters of HPMN model, including size of each memory slot and the update periods t^j of the j-th layer which are shown in Table 3. The reported update periods are listed from the first (lowest) layer to the last (highest).

Table 3: The HPMN structures on different datasets.

Dataset	Mem. Size	Update Periods
Amazon	32	3 layers: 1, 2, 4
Taobao	32	4 layers: 1, 2, 4, 12
XLong	32	6 layers: 1, 2, 4, 8, 16, 32

4.2 Experimental Results and Analysis

In this section, we present the experiment results in Table 4 and conduct an analysis from several perspectives. Recall that the compared models are divided into three groups as mentioned in Sec. 4.1.4.

Comparison between HPMN and baselines. From Table 4, we can tell that HPMN improves the performance significantly against all the baselines and achieves state-of-the-art performance (**RQ2**).

The aggregation-based models in Group 1, i.e., DNN and SVD++, perform not well as the sequential modeling methods, which indicates that there exist sequential patterns in user behavior data and simply aggregating user behavior features may result in poor performance.

Comparing with the other sequential modeling methods of Group 2, HPMN outperforms all of them regardless of the length of user

Table 4: Performance Comparison. (* indicates p-value < 10^{-6} in the significance test. ↑ and ↓ indicates the *performance* over lifelong sequences (with length T) is better or worse than the same model over short sequences (with length s). AUC: the higher, the better; Log-loss: the lower, the better. The second best performance of each metric is underlined.)

Model Group	Model	Len.	AUC			Log-loss		
			Amazon	Taobao	XLong	Amazon	Taobao	XLong
Group 2	GRU4Rec	s	0.7669	0.8431	0.8716	0.5650	0.4867	0.4583
	Caser	s	0.7509	0.8260	0.8467	0.5795	0.5094	0.4955
	DIEN	s	0.7725	0.8914	<u>0.8725</u>	0.5604	0.4184	<u>0.4515</u>
	RUM	s	0.7434	0.8327	0.8512	0.5819	0.5400	0.4931
Group 1	DNN	T	0.7546	0.7460	0.8152	0.6869	0.5681	0.5365
	SVD++	T	0.7155	0.8371	0.8008	0.6216	0.8371	1.7054
Group 2	GRU4Rec	T	0.7760 ↑	0.8471 ↑	0.8702 ↓	0.5569 ↑	0.4827 ↑	0.4630 ↓
	Caser	T	0.7582 ↑	0.8745 ↑	0.8390 ↓	0.5704 ↑	0.4550 ↑	0.5050 ↓
	DIEN	T	<u>0.7770</u> ↑	<u>0.8934</u> ↑	0.8716 ↓	<u>0.5564</u> ↑	<u>0.4155</u> ↑	0.4559 ↓
	RUM	T	0.7464 ↑	0.8370 ↑	0.8649 ↑	0.6301 ↓	0.4966 ↑	0.4620 ↑
Group 3	LSTM	T	0.7765	0.8681	0.8686	0.5612	0.4603	0.4570
	SHAN	T	0.7763	0.8828	0.8369	0.5595	0.4318	0.5000
	HPMN	T	**0.7809***	**0.9240***	**0.8929***	**0.5535***	**0.3487***	**0.4150***

behavior sequences. Since GRU4Rec was proposed for short-term session-based recommendation, thus it has the same issue as LSTM which may lose some knowledge of the long-term behavior dependency. Though the attention mechanism of DIEN improves the performance from GRU4Rec in a large margin, it either ignores the multi-scale user behavior patterns, which will be illustrated from an example in the next section. Moreover, DIEN model requires to conduct online inference over the whole sequence for prediction, which lacks of practical efficiency considering extremely long, especially lifelong user behavior sequences. From the results of Caser which uses CNN to extract sequential patterns, we may tell that convolution operation may not be appropriate for sequential user modeling. As for RUM model, though it utilizes an external memory for user modeling, it fails to capture sequential patterns which results in quite poor performance. Moreover, this proposed model was originally optimizing for other metrics [5], e.g., precision and recall, thus it may not perform well for user response prediction.

By comparing HPMN with the models in Group 3, i.e., LSTM and SHAN, we find that although both baselines are proposed to deal with long-term user modeling, HPMN has better performance on the very long sequences. The reason would be that LSTM has limited memory capacity to retain the knowledge, and SHAN has not considered any sequential patterns in the user behaviors.

Analysis about Lifelong Sequential Modeling. Recall that we evaluate all the short-term sequential modeling methods on both short sequence data and lifelong sequence data, as is shown in Table 4 and we have highlighted the results of the performance gain ↑ (and drop ↓) in the table of the latter case compared with the former case.

From the table, we find that almost all the models gain an improvement when modeling on the lifelong user behavior sequences on Amazon and Taobao datasets. However, on XLong dataset, the performance of GRU4Rec, Caser and DIEN drops, while the memory-based model, i.e., RUM achieve better performance than itself on short sequences. Note that, our HPMN model performs best. All the phenomenon reflect that the incorporation of lifelong sequences contributes better user modeling and response prediction (**RQ1**). Nevertheless, it also requires well designed memory model for lifelong modeling, while our HPMN model achieves satisfying performance on this problem.

Model Convergence. We plot the learning curves of HPMN model over the three datasets in Figure 5. As is shown in the figure, HPMN converges quickly, the Log-loss value on three datasets all drop to the stable convergence after about one iteration over the whole training set.

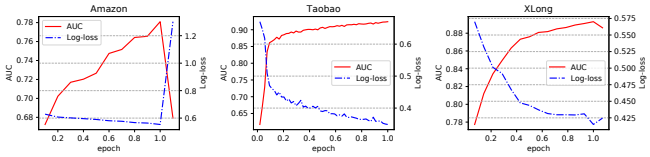

Figure 5: The learning curve on three datasets. Here one epoch means the whole iteration over the training dataset.

4.3 Extended Investigation

In this section, we further investigate the patterns that HPMN captures when dealing with lifelong sequence (**RQ3**) and the model capacity of memorization.

Sequential Patterns with Multi-scale Dependency. In Figure 6, we plot three real examples of user behavior sequence with length $T = 1000$ sampled from XLong dataset. These three sequences reflect the long-term, short-term and multi-scale sequential patterns captured by HPMN, respectively.

In the first example, the target item is "lotion" clicked by the user at the final prediction time. As we find in her behavior history, there are several clicks on lotions at the 31st, 33rd and 37th positions of her behavior sequence, which is far from the tail of her latest behaviors. When HPMN model takes the target item as query to conduct the user representation, from the attention heatmap calculated by HPMN as that in Eq. (7), we can tell that the fifth layer of HPMN has the maximum attentions, whose update period is relative large. It shows that HPMN captures long-term sequential pattern in the memory maintained by high layers.

In the second example, User 2 at last clicked a desk, and some similar items (table, cabinet) are also clicked in very recent history. However, these kinds of furniture are not clicked in the former part of the sequence. The first memory of HPMN has the maximum attention value which shows that the lower layer is better at modeling short-term pattern for that it updates the memory more frequently to capture user's short-term interests.

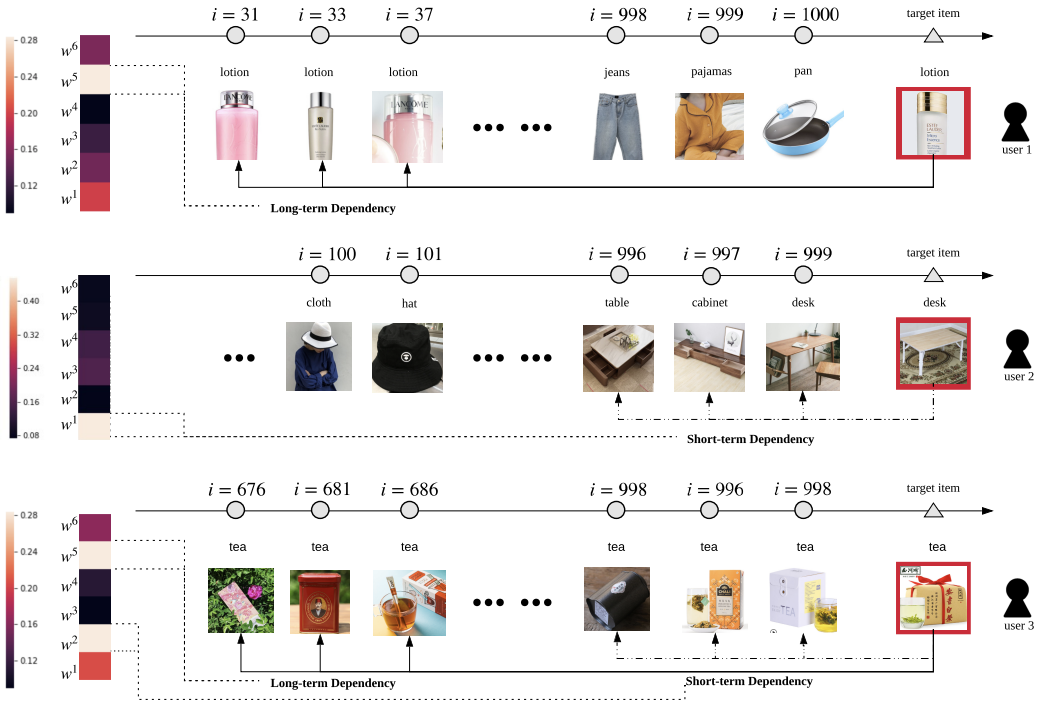

Figure 6: An illustration of long-term, short-term and multi-scale sequential patterns that are captured by HPMN.

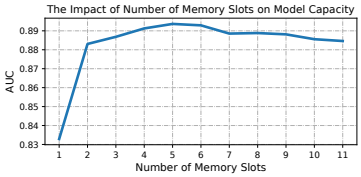

Figure 7: The performance of HPMN with various memory numbers on XLong Dataset. The update period of each j-th layer follows exponential sequence $\{2^{j-1}\}_{j=1}^{11}$.

As for User 3, the click behavior on the target item has both long-term and short-term dependencies, the similar items are clicked in the recent history and in the former part of her behavior sequence. After inference through HPMN model, the second and fifth layers have higher attention values, for they could capture short-term and long-term dependencies respectively. Thus, this demonstrates that HPMN has the ability to capture multi-scale sequential patterns.

Memory Capacity. In Figure 7, we plot the AUC performance of HPMN with different numbers of memory slots on XLong Dataset. Note that the number of memory slots is equal to the number of HPMN layers. On one hand, when the number of memory slots for each user is less than 5, the prediction performance of the model rises sharply as the memory increases. This indicates that it requires large memory of sequential patterns for long behavior sequences. And increasing the memory according to the growth of user behavior sequence helps HPMN to better capture lifelong sequential patterns. However, when the number of memory slots is larger than 5, the AUC score drops slightly as the memory number increases. This demonstrates that, on the other hand, the model capacity has some constraints for the specific length of user behavior sequence. It provides some guides about memory expanding and the principle of enlarging HPMN model for lifelong sequential modeling with evolving user behavior sequence, as is discussed in Section 3.4.

5 CONCLUSION

In this paper, we present lifelong sequential modeling for user response prediction. To achieve this goal, we conduct a framework with a memory network model maintaining the personalized hierarchical memory for each user. The model updates the corresponding user memory through periodic updating machanism to retain the knowledge of multi-scale sequential patterns. The user lifelong memory will be attentionally read for the subsequent user response prediction. The extensive experiments have demonstrated the advantage of lifelong sequential modeling and our model has achieved a significant improvement against strong baselines including state-of-the-art.

In the future, we will adopt our lifelong sequential modeling to improve multi-task user modeling such as prediction of both user clicks and conversions [35]. We also plan to investigate learning for dynamic update period of each layer, to capture more flexible user behavior patterns.

6 ACKNOWLEDGMENTS

The work is sponsored by Alibaba Innovation Research. 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, Rahul Agrawal, Rajiv Khanna, and Nagaraj Kota. 2010. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *KDD*.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR* (2015).

[3] 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*.

[4] Bhaskar Bhattacharya and Desale Habtzghi. 2002. Median of the p value under the alternative hypothesis. *The American Statistician* (2002).

[5] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *WSDM*.

[6] Zhiyuan Chen, Estevam R Hruschka Jr, and Bing Liu. 2016. Lifelong machine learning and computer reading the web. In *KDD*.

[7] Zhiyuan Chen and Bing Liu. 2016. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* (2016).

[8] 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*.

[9] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. 2016. Hierarchical multiscale recurrent neural networks. *arXiv preprint arXiv:1609.01704* (2016).

[10] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. 2016. Reducing overfitting in deep networks by decorrelating representations. *ICLR* (2016).

[11] Georges E Dupret and Benjamin Piwowarski. 2008. A user browsing model to predict search engine click data from past observations.. In *SIGIR*.

[12] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. In *SIGIR*.

[13] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. *arXiv preprint arXiv:1804.10862* (2018).

[14] Kun Gai, Xiaoqiang Zhu, Han Li, Kai Liu, and Zhe Wang. 2017. Learning Piecewise Linear Models from Large Scale Data for Ad Click Prediction. *arXiv preprint arXiv:1704.05194* (2017).

[15] Thore Graepel, Joaquin Q Candela, Thomas Borchert, and Ralf Herbrich. 2010. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *ICML*. 13–20.

[16] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* (2014).

[17] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247* (2017).

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

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

[20] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *ADKDD*.

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

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

[23] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* (1997).

[24] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR*.

[25] Meng Jiang, Peng Cui, Fei Wang, Xinran Xu, Wenwu Zhu, and Shiqiang Yang. 2014. Fema: flexible evolutionary multi-faceted analysis for dynamic behavioral pattern discovery. In *KDD*.

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

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

[28] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*.

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

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

[31] Jan Koutnik, Klaus Greff, Faustino Gomez, and Juergen Schmidhuber. 2014. A clockwork rnn. *ICML* (2014).

[32] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*.

[33] Kuang-chih Lee, Burkay Orten, Ali Dasdan, and Wentong Li. 2012. Estimating conversion rate in display advertising from past erformance data. In *KDD*.

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

[35] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. 2018. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. *arXiv preprint arXiv:1804.07931* (2018).

[36] Simon J Mason and Nicholas E Graham. 2002. Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation. *Quarterly Journal of the Royal Meteorological Society* (2002).

[37] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR*.

[38] Aditya Krishna Menon, Krishna-Prasad Chitrapura, Sachin Garg, Deepak Agarwal, and Nagaraj Kota. 2011. Response prediction using collaborative filtering with hierarchies and side-information. In *KDD*.

[39] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *EMNLP*.

[40] Richard J Oentaryo, Ee-Peng Lim, Jia-Wei Low, David Lo, and Michael Finegold. 2014. Predicting response in mobile advertising with hierarchical importance-aware factorization machine. In *WSDM*.

[41] 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*.

[42] Kan Ren, Yuchen Fang, Weinan Zhang, Shuhao Liu, Jiajun Li, Ya Zhang, Yong Yu, and Jun Wang. 2018. Learning Multi-touch Conversion Attribution with Dual-attention Mechanisms for Online Advertising. *CIKM* (2018).

[43] 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).

[44] Kan Ren, Weinan Zhang, Yifei Rong, Haifeng Zhang, Yong Yu, and Jun Wang. 2016. User response learning for directly optimizing campaign performance in display advertising. In *CIKM*.

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

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

[47] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting clicks: estimating the click-through rate for new ads. In *WWW*. ACM, 521–530.

[48] Shagun Sodhani, Sarath Chandar, and Yoshua Bengio. 2018. On Training Recurrent Neural Networks for Lifelong Learning. *arXiv preprint arXiv:1811.07017* (2018).

[49] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *NIPS*.

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

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

[52] Qinyong Wang, Hongzhi Yin, Zhiting Hu, Defu Lian, Hao Wang, and Zi Huang. 2018. Neural Memory Streaming Recommender Networks with Adversarial Training. In *KDD*.

[53] Chopra Sumit Bordes Antoine Weston, Jason. 2015. Memory Networks. *ICLR* (2015).

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

[55] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, Yanchi Liu, Guandong Xu, Xing Xie, Hui Xiong, and Jian Wu. 2018. Sequential Recommender System based on Hierarchical Attention Networks. In *IJCAI*.

[56] Shuai Zhang, Yi Tay, Lina Yao, and Aixin Sun. 2018. Next Item Recommendation with Self-Attention. *RecSys* (2018).

[57] Weinan Zhang, Tianming Du, and Jun Wang. 2016. Deep Learning over Multi-field Categorical Data: A Case Study on User Response Prediction. *ECIR* (2016).

[58] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal real-time bidding for display advertising. In *KDD*.

[59] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential click prediction for sponsored search with recurrent neural networks. *arXiv preprint arXiv:1404.5772* (2014).

[60] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *WSDM*.

[61] 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).

[62] 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*.

[63] 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*.