Long Short-Term Temporal Meta-learning in Online Recommendation

Ruobing Xie*
WeChat, Tencent
Beijing, China
ruobingxie@tencent.com

Yalong Wang*
WeChat, Tencent
Beijing, China
vinceywang@tencent.com

Rui Wang
WeChat, Tencent
Beijing, China
rysanwang@tencent.com

Yuanfu Lu
WeChat, Tencent
Beijing, China
lucasyflu@tencent.com

Yuanhang Zou
WeChat, Tencent
Beijing, China
yuanhangzou@tencent.com

Feng Xia
WeChat, Tencent
Beijing, China
xiafengxia@tencent.com

Leyu Lin
WeChat, Tencent
Beijing, China
goshawklin@tencent.com

ABSTRACT

An effective online recommendation system should jointly capture users' long-term and short-term preferences in both users' internal behaviors (from the target recommendation task) and external behaviors (from other tasks). However, it is extremely challenging to conduct fast adaptations to real-time new trends while making full use of all historical behaviors in large-scale systems, due to the real-world limitations in real-time training efficiency and external behavior acquisition. To address these practical challenges, we propose a novel Long Short-Term Temporal Meta-learning framework (LSTTM) for online recommendation. It arranges user multi-source behaviors in a global long-term graph and an internal short-term graph, and conducts different GAT-based aggregators and training strategies to learn user short-term and long-term preferences separately. To timely capture users' real-time interests, we propose a temporal meta-learning method based on MAML under an asynchronous optimization strategy for fast adaptation, which regards recommendations at different time periods as different tasks. In experiments, LSTTM achieves significant improvements on both offline and online evaluations. It has been deployed on a widely-used online recommendation system named WeChat Top Stories, affecting millions of users.

CCS CONCEPTS

• **Information systems → Recommender systems**.

*Both authors contributed equally to this research. Ruobing Xie is the corresponding author (ruobingxie@tencent.com).

KEYWORDS

recommendation, temporal meta-learning, online recommendation

ACM Reference Format:
Ruobing Xie, Yalong Wang, Rui Wang, Yuanfu Lu, Yuanhang Zou, Feng Xia, and Leyu Lin. 2022. Long Short-Term Temporal Meta-learning in Online Recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3488560.3498371

1 INTRODUCTION

Real-world industry-level recommendation systems usually need to interact with complicated practical scenarios. Large-scale online recommendation systems in super platforms such as Google and Amazon usually have the following two complexities: (1) ***user behaviors are multi-source***. Super platforms usually have multiple applications with shared user accounts (e.g., search, news, and video recommendation in Google), which can meet users' diverse demands. Users' various behaviors in multiple applications are related via shared accounts after user approvals, which can provide additional information from different aspects for the target recommendation task. In real-world scenarios, recommendation systems benefit from not only effective algorithms, but also informative data. A good online recommendation should make full use of both *user internal behaviors* (i.e., user behaviors in the target recommendation task) and *user external behaviors* (i.e., user behaviors in other applications). (2) ***User preferences are variable***. In large-scale recommendation systems, millions of new items are daily generated and added to the candidate pool. Online systems (especially news and video recommendations) should capture users' short-term preferences timely and accurately, since users' concentrations are easily attracted by new trends and hot topics. In contrast, users' long-term preferences could provide users' aggregated stable interests as effective supplements to short-term interests. Hence, a good real-world recommendation should well capture both *user variable short-term preferences* and *user stable long-term preferences*.

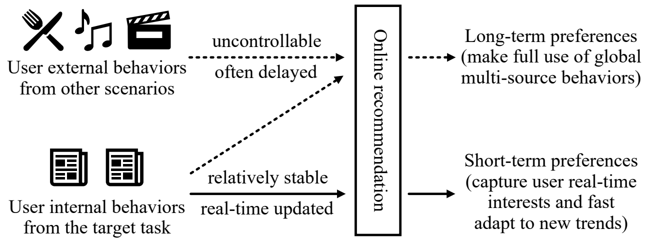
User external behaviors from other scenarios
uncontrollable
often delayed
User internal behaviors from the target task
relatively stable
real-time updated
Online recommendation
Long-term preferences (make full use of global multi-source behaviors)
Short-term preferences (capture user real-time interests and fast adapt to new trends)

Figure 1: A practical online recommendation with asynchronous multi-source internal and external behaviors, aiming to model users' short-term and long-term preferences.

In this work, we attempt to design an effective and efficient online recommendation framework, which jointly considers both **users' internal/external behaviors** and **users' short-term/long-term preferences**. This online recommendation mainly faces the following three challenges: (1) *How to jointly consider internal and external behaviors?* As in Fig. 1, users' multi-source behaviors in various applications usually have different features. It is difficult to combine this heterogeneous information. Moreover, in real-world scenarios, users' internal and external behaviors are often asynchronous (i.e., real-time internal feedbacks V.S. delayed and uncontrollable external behaviors), which makes it inconvenient to conduct a stable joint learning. (2) *How to effectively model both short-term and long-term preferences?* Both users' short-term and long-term interests are essential in real-world recommendations, while there are biases between them. Models should capture both short-term and long-term preferences, and should be able to determine which interest is more essential in different scenarios. (3) *How to conduct a timely online update to capture short-term preferences?* It is crucial to conduct an online update to capture user variable interests over time. However, it is extremely time-consuming (or even impractical) to conduct a complete model retraining or a complicated fine-tuning with million-level new behaviors. It is challenging to balance effectiveness and efficiency in online recommendation. These three challenges are important in real-world scenarios, while there are barely any works that jointly address them systematically.

To address these issues, we propose a novel **Long Short-Term Temporal Meta-learning (LSTTM)** framework for practical online recommendations. Specifically, we build two heterogeneous graphs, namely the global long-term graph and the internal short-term graph, to capture different user preferences from multi-source behaviors. The *global long-term graph* is a huge graph containing all users' internal and external behaviors. It aims to build a global view of all multi-source interactions between users and items to better capture users' long-term preferences. In contrast, the *internal short-term graph* focuses on the short-term behaviors of the target recommendation task, which is specially optimized for real-time interest evolutions. LSTTM adopts graph attention networks (GATs) with different neighbor sampling strategies to learn user long-term and short-term representations from these heterogeneous graphs, and then combines them via a gating fusion. To better capture long-/short-term preferences and balance effectiveness and efficiency in online serving, we further design an asynchronous optimization method. We propose a **temporal MAML** training strategy, which *regards recommendations at different time periods as different tasks*, based on a classical meta-learning method named MAML [6]. This new temporal meta-learning enables fast adaptations to real-time variable user preferences. The advantages of LSTTM mainly locate in three aspects: (1) LSTTM makes full use of all internal and external behaviors via two huge graphs. (2) We conduct different GAT aggregations and training strategies for two graphs to learn user short-term and long-term preferences separately, and combine them via gating. The differential designs enable more refined preference learning. (3) The asynchronous optimization with temporal MAML facilitates fast adaptations to new trends under the practical (data and computation) limitations of real-world systems.

In experiments, we conduct an offline temporal CTR prediction with competitive baselines on a real-world recommendation system, and also conduct an online A/B test. The significant offline and online improvements show the effectiveness of LSTTM. Moreover, we also conduct an ablation study to better understand the effectiveness of different components. The contributions of this work are concluded in four points as follows:

- We first systematically address the practical challenges of jointly considering users' internal/external behaviors and short-/long-term preferences in recommendation via our new proposed LSTTM framework. LSTTM is effective and easy to deploy in practical systems.
- We build two graphs focusing on different aspects to make full use of all internal/external behaviors. Moreover, we set customized GAT aggregators and training strategies to better learn user short-/long-term preferences.
- We design a novel temporal meta-learning method based on MAML, which enables fast adaptations to users' real-time preferences. To the best of our knowledge, we are the first to adopt temporal MAML in online recommendation.
- We achieve significant improvements on offline and online evaluations. LSTTM has been deployed on a real-world system for millions of users. The idea of temporal MAML can also be easily transferred to other models and tasks.

2 RELATED WORKS

Recommender System. In real-world recommendation, Factorization machine (FM) [19], NFM [8], DeepFM [7], AutoInt [20], DFN [29] are widely used to model feature interactions. User behaviors are one of the most essential features to learn user preferences. Lots of models [21, 28, 31, 34, 37] regard user behaviors as sequences to model user preferences via attention and transformer. Besides sequence-based models, graph-based models such as SR-GNN [26] and GCE-GNN [25] use graph neural networks (GNNs) on user behavior graphs built from sessions. Inspired by these works, we also adopt GAT [23] to model user internal/external behaviors, and use DeepFM to model long-/short-term feature interactions.

Both long- and short- term preferences are essential in recommendation. Xiang et al. [27] proposes an injected preference fusion on a session-based temporal graph to model users' long-term and short-term preferences simultaneously. STAMP [12] highlights users' current interests from the short-term memory of the last clicks. DIEN [36] explicitly models user's interest evolutions. Some works jointly consider short-term and long-term representations

[1, 33]. Hu et al. [9] conducts LSTM for short-term interests and GNNs for long-term preferences. Intent preference decoupling [13] and online exploration [32] are used for online recommendation. However, these models cannot capture real-time global hot topics well, and do not make full use of multiple behaviors. LSTTM builds two graphs focusing on long-term and short-term interests, and uses a temporal MAML to highlight short-term interests.

Meta-learning in Recommendation. Meta-learning aims to transfer the meta knowledge so as to rapidly adapt to new tasks with a few examples, which is regarded as "learning to learn" [24]. MAML [6] is a classical model-agnostic meta-learning method widely used in various fields, which provides a good weight initialization. Some works have also explored meta-learning on graphs [16, 41].

In recommendation, meta-learning has also been verified on various cold-start scenarios, including cold-start user [10, 40], item [22, 39], cross-domain recommendation [4, 38] and model selection [15]. MeLU [10] brings in MAML to model cold-start users. Bharadhwaj [2] improves MAML with the dynamic meta-step for cold-start users. Lu et al. [14] combines MAML with heterogeneous information networks, considering both semantic-wise and task-wise adaptations. Besides MAML-based methods, Pan et al. [17] proposes meta-embeddings for warm-up scenarios. MAMO [3] designs two task-specific and feature-specific memories. SML [35] focuses on model retraining, which learns a transfer function from old to new parameters. Different from these models, LSTTM designs a temporal MAML to accelerate model adaptation to users' short-term preferences. To the best of our knowledge, we are the first to conduct temporal MAML in recommendation.

3 METHODOLOGY

In this work, we attempt to jointly consider both user internal and external behaviors to capture users' short-term/long-term preferences in online recommendation. We first give brief definitions of the notions used in this work as follows:

Definition 1: User internal behaviors. In LSTTM, the user behaviors of the target recommendation task are viewed as the user internal behaviors. These behaviors are the main sources of user preferences inside the target recommendation.

Definition 2: User external behaviors. All user behaviors from other applications are considered as the user external behaviors. These user external behaviors are related to their internal behaviors via the shared user accounts under user approvals. These behaviors are informative complements to the internal behaviors.

Definition 3: Temporal meta-learning. Meta-learning aims to fast adapt to new tasks [6]. We define the temporal meta-learning, which *regards recommendations in different time periods as different tasks*, since user preferences can frequently change over time. The temporal meta-learning concentrates on fast adaptations between user behaviors at different times for short-term preferences.

3.1 Overall Framework

Fig. 2 displays the overall architecture of LSTTM. The internal short-term graph regards all users and items in the target recommendation task as nodes, with all user internal behaviors utilized as edges. A heterogeneous GAT with temporal neighbor sampling is used for node aggregation to highlight user short-term preferences. In contrast, the global long-term graph is a much larger heterogeneous graph having all user internal and external behaviors, which focuses on the global view of user long-term preferences in multi-source behaviors. The long short-term graph fusion module then conducts a gating strategy to combine two short-/long-term representations, followed by a feature interaction module to generate the final recommendation. We propose an asynchronous optimization to learn long-term and short-term preferences differently. For the short-term graph and gating fusion modules, we propose a temporal MAML to highlight short-term interest modeling. While for the long-term graph, we rely on a multi-hop neighbor-similarity based loss for efficient long-term preference learning. In this case, LSTTM can make full use of both internal and external information to leverage user short-term and long-term preferences, finding an industrial balance between effectiveness and efficiency in practice.

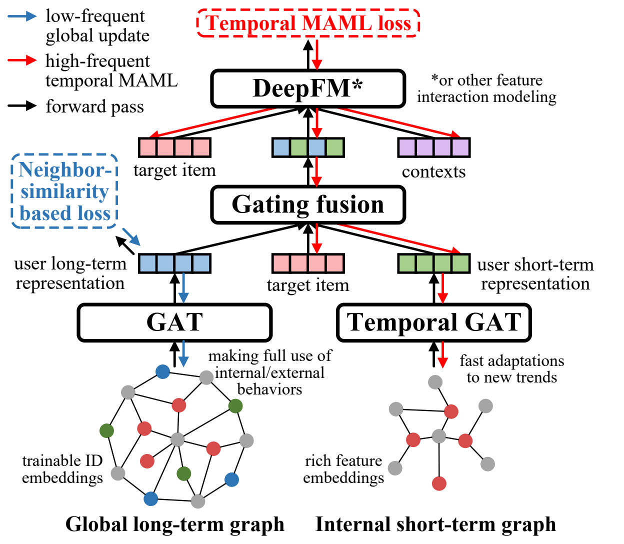
low-frequent global update
high-frequent temporal MAML
forward pass
Temporal MAML loss
DeepFM*
*or other feature interaction modeling
target item
contexts
Neighbor-similarity based loss
Gating fusion
user long-term representation
target item
user short-term representation
GAT
Temporal GAT
making full use of internal/external behaviors
fast adaptations to new trends
trainable ID embeddings
rich feature embeddings
Global long-term graph
Internal short-term graph

Figure 2: Overall architecture of LSTTM.

3.2 Internal Short-term Graph

The internal short-term graph module models user short-term internal behaviors. Specifically, it has two types of nodes indicating all users $u \in U$ and all items $d \in D_I$ in the target recommendation task, which are linked by user-item click behaviors as edges. U and D_I are the overall user set and internal item set respectively. We use $\boldsymbol{u}_i^0$ and $\boldsymbol{d}_i^0$ to represent the i-th input feature embeddings of users and items, which are trainable embeddings built from different types of user attributes and item features (e.g., tag, topic).

Inspired by Veličković et al. [23], we build an enhanced GAT layer for short-term oriented node aggregation. For a user u_i and his/her click sequence $seq_i = \{d_{i,1}, \cdots, d_{i,m}\}$, different from conventional random-based neighbor sampling, we conduct a temporal neighbor sampling, which only selects the top-K most recent clicked items. The temporal neighbor sampling generates the neighbor set N_{u_i} as:

$$N_{u_i} = \text{Temporal}(seq_i) = \{d_{i,m-K+1}, \cdots, d_{i,m}\}. \quad (1)$$

Similarly, we also generate the sampled neighbor set of items as $N_{d_i} = \{u_{i,m'-K+1}, \cdots, u_{i,m'}\}$. With the temporal neighbor set N_{u_i}, we build the user representation $\boldsymbol{u}_i^k$ at the k-th layer via item embeddings in the $k-1$ layer as follows:

$$\boldsymbol{u}_i^k = \sigma(\sum_{d_{i,j} \in N_{u_i}} \alpha_{ij}^k \boldsymbol{W}_d^k \boldsymbol{d}_{i,j}^{k-1}). \tag{2}$$

$\boldsymbol{W}_d^k$ is the weighting matrix. α_{ij}^k represents the attention between u_i and $d_{i,j}$ in this layer, which is formalized as:

$$\alpha_{ij}^k = \frac{\exp(f(\boldsymbol{w}^\top [\boldsymbol{W}_u \boldsymbol{u}_i^{k-1} || \boldsymbol{W}_d \boldsymbol{d}_{i,j}^{k-1}]))}{\sum_{d_{i,l} \in N_{u_i}} \exp(f(\boldsymbol{w}^\top [\boldsymbol{W}_u \boldsymbol{u}_i^{k-1} || \boldsymbol{W}_d \boldsymbol{d}_{i,l}^{k-1}]))}, \tag{3}$$

where $f(\cdot)$ indicates a LeakyReLU activation and $||$ indicates the concatenation. $\boldsymbol{w}^\top$, $\boldsymbol{W}_u$ and $\boldsymbol{W}_d$ are the weighting vector and matrices. Note that the temporal neighbor set N_{u_i} changes over time, since the internal short-term graph is a dynamic graph that is updated via users' new behaviors. Finally, we conduct a two-layer temporal GAT to generate the user short-term representation $\boldsymbol{u}_i^s = \boldsymbol{u}_i^2$, which is fed into the next gating fusion module. The aggregation of items is similar to that of users. We use GAT since it is effective, efficient, and easy to deploy on billion-level huge graphs. It is also convenient to conduct other enhanced GNN models in this module.

In internal short-term graph, the temporal neighbor sampling highlights the individual-level short-term preferences. We also propose a temporal meta-learning method to update this module, attempting to capture the short-term preferences at the global level, which will be introduced in Sec. 3.5.

3.3 Global Long-term Graph

The global long-term graph module aims to take advantage of all user diverse preferences in multiple applications. It considers users $u \in U$ and all internal and external items $d \in D_I \bigcup D_E$ as nodes, where D_I and D_E represent the overall internal and external item sets. All heterogeneous user-item interactions in different applications as regarded as edges. The details of the external behaviors are in Sec. 4. Since heterogeneous items usually have different feature fields that are hard to align, we represent all users and items via trainable ID embeddings $\bar{\boldsymbol{u}}_i^0$ and $\bar{\boldsymbol{d}}_i^0$ in the same space.

We also conduct a two-layer GAT for neighbor aggregation similar as Eq. (2) to Eq. (3), where the neighbor set $\bar{N}_{u_i}$ are randomly sampled or selected via certain importances. The user long-term representation $\bar{\boldsymbol{u}}_i^l = \bar{\boldsymbol{u}}_i^2$ is also utilized in the gating fusion module. Since the overall behaviors are too enormous to be fully retrained in online, and external behaviors are usually delayed and uncontrollable, we conduct an enhanced neighbor-similarity based loss to train this module asynchronously introduced in Sec. 3.5.

Comparing the internal short-term graph modeling with the global long-term graph modeling, we can find three main differences: (1) they adopt different data sources, and thus have different input feature forms (i.e., detailed user and item features V.S. trainable ID embeddings). (2) They conduct different neighbor sampling strategies (i.e., temporal-based V.S. random or importance-based) due to their different long-/short- term concentrations. (3) They are updated under different strategies (i.e., temporal meta-learning V.S. neighbor-similarity based loss), considering the effectiveness of short-term preference modeling and the efficiency of global user behavior modeling in practice. We use GNN to capture heterogeneous node interactions, and conduct GAT in node aggregation for efficiency. It is also not difficult to conduct other complicated heterogeneous information networks in LSTTM.

3.4 Long- Short-term Preference Fusion

This module attempts to combine both user short-term and long-term representations $\boldsymbol{u}_i^s$ and $\bar{\boldsymbol{u}}_i^l$ to generate the ranking score. We conduct a gating-based fusion to generate the final user representation $\boldsymbol{u}_i$ via $\boldsymbol{u}_i^s$ and $\bar{\boldsymbol{u}}_i^l$ as follows:

$$\boldsymbol{u}_i = g(\boldsymbol{x}_i^s)\boldsymbol{u}_i^s + g(\boldsymbol{x}_i^l)\bar{\boldsymbol{u}}_i^l. \tag{4}$$

$g(\cdot)$ indicates the gating function, which is measured via the corresponding user embeddings and the target item $\boldsymbol{d}_j^s$ as:

$$[g(\boldsymbol{x}_i^s), g(\boldsymbol{x}_i^l)] = \mathrm{Softmax}([\boldsymbol{w}_g^s[\boldsymbol{u}_i^s||\boldsymbol{d}_j^s], \boldsymbol{w}_g^l[\bar{\boldsymbol{u}}_i^l||\boldsymbol{d}_j^s]]). \tag{5}$$

$\boldsymbol{w}_g^s$ and $\boldsymbol{w}_g^l$ are weighting vectors. $\boldsymbol{d}_j^s$ is a trainable item ID embedding that is randomly initialized. With this gating-based fusion, users can get personalized weights on long-/short- term preferences for different items, which helps to improve the performances.

After gating fusion, the final user representation $\boldsymbol{u}_i$ is aggregated with the recommendation contexts $\boldsymbol{c}$ and target item embedding $\boldsymbol{d}_j^s$, and then fed into the downstream neural ranking models. We conduct a widely-used DeepFM [7] to model the feature field interactions between user, item and contexts as follows:

$$p(i,j) = \mathrm{DeepFM}(\boldsymbol{u}_i, \boldsymbol{d}_j^s, \boldsymbol{c}). \tag{6}$$

$p(i,j)$ is the click probability for u_i and d_j. It is also easy to adopt other feature interaction models for feature interactions.

3.5 Optimization with Temporal MAML

The asynchronous optimization with temporal MAML is the key contribution of LSTTM. In practice, timely model updating is significant in online recommendation, while there are two challenges in real-world systems. (1) It is extremely difficult to conduct a full model retraining or a complicated fine-tuning in real-time for GNN models with large-scale graphs, especially with the nearly billion-level interactions in the huge global graph. (2) Moreover, multi-source behaviors are usually obtained asynchronously (e.g., external behaviors are often delayed) due to some practical system limitations. Hence, we decouple the training of internal short-term graph and global long-term graph into two asynchronous optimization objectives, including a temporal MAML based cross-entropy loss and a multi-hop neighbor-similarity based loss. It enables LSTTM to be smoothly and timely updated.

3.5.1 Temporal Meta-learning. To enhance LSTTM with the capability of fast adaptation to user short-term interests, we propose a novel **temporal MAML** training strategy based on [6]. Different from conventional meta-learning based recommendations that usually consider each user or domain as a task, our temporal MAML **regards recommendation in each time period as a task**.

Specifically, we first divide all training instances into different sets according to their time periods (e.g., we view each hour as a time period for the practical demands). In temporal MAML training, we regard instances in two adjacent hours as a task. The *support set* contains instances of the former hour, while the *query set* contains instances of the latter hour. Note that an instance can belong to both a support set and a query set in two tasks. We could further divide these temporal tasks into more fine-granularity tasks, where all instances in a task derive from the same user group (which is built via similar basic profiles or user interests) at the same time. Through these fine-granularity tasks, the instances in the support set and the query set will be more relevant. In this case, the temporal MAML will focus on the new trends in certain user communities instead of whole user groups. We can choose different temporal MAML settings according to the practical needs of systems.

In training, we sample different temporal tasks containing training instances in different time periods to form a batch. To make sure our temporal MAML can learn a better initialization for fast adaptation on all time periods, we diversify the sampled temporal tasks to make them dissimilar to each other (e.g., selecting tasks in different hours and days). The inner update (line 6) with support sets and the outer update (line 8) with query sets are similar to the original MAML. We conduct one gradient update in both inner and outer updates for efficiency. In online, we also conduct one gradient update for all new user feedbacks similar to MAML, which enables high-frequent (or even real-time) online model learning. Algorithm 1 gives the pseudo-code of temporal MAML with temporal tasks.

Algorithm 1 Temporal MAML:

```
Input: The distribution over temporal tasks $p(T)$
Output: The parameter set $\theta = (\theta_s, \theta_f)$ of the internal short-term graph
    module $\theta_s$ and the long- short-term preference fusion $\theta_f$
1: Randomly initialize $\theta$
2: while not converge do
3:     Sample batch of diversified temporal tasks $T_i \sim p(T)$
4:     for all $T_i$ do
5:         Evaluate $\nabla_\theta L_{T_i}(f_\theta)$ with respect to $K$ examples in each
           support set, using the graphs in $T_i$'s time period
6:         Compute adapted parameters with gradient descent via
           the inner step size $\alpha$: $\theta'_i = \theta - \alpha\nabla_\theta L_{T_i}(f_\theta)$
7:     end for
8:     Update the parameter $\theta$ via the outer step size $\beta$: $\theta \leftarrow \theta -$
       $\beta\nabla_\theta \sum_{T_i \sim p(T)} L_{T_i}(f_{\theta'_i})$ with all query sets
9: end while
```

Under the temporal MAML training framework, we conduct a classical cross entropy loss L_T with the click probability $p(i,j)$ of user u_i and item d_j on the positive set (clicked user-item instances) S_p and negative set (unclicked user-item instances) S_n as follows:

$$L_T = -\frac{1}{N}\Big(\sum_{S_p}\log p(i,j) + \sum_{S_n}\log(1-p(i,j))\Big). \tag{7}$$

Note that the L_T is only used for updating the internal short-term graph and the long- short-term preference fusion modules via the temporal MAML as in Fig. 2. A gradient block is conducted to the global long-term graph module, since it is responsible for modeling users' stable long-term preferences from multi-source behaviors, and thus should be fully trained on all behaviors.

Motivations and advantages of temporal MAML. The motivations and advantages of the temporal MAML are concluded as follows: (1) we attempt to capture new trends and hot topics timely in practical recency-sensitive recommendation systems. The temporal MAML highlights the model's capability in capturing global user interest evolutions via temporal tasks, which enables fast adaptations to users' variable short-term interests on the global new trends. (2) MAML can fast adapt to new tasks [6], while classical MAML-based models mainly regard individual users as tasks, and thus cannot model the global temporal factors well. Hence, we propose the temporal MAML focusing on the temporal tasks. (3) The internal short-term graph modeling can also provide short-term interests via the temporal neighbor sampling (see Eq. (1)). However, it merely concentrates on the individual user-related short-term behaviors, ignoring the global short-term behaviors generated by other users (which is essential especially when the user does not have recent behaviors). The temporal MAML and the short-term graph modeling are strong supplements to each other in capturing user real-time preferences. (4) The temporal MAML is also naturally suitable for the asynchronous online learning with large-scale instances, since it only needs a one-step update.

3.5.2 *Multi-hop Neighbor-similarity Based Loss.*

Differing from the internal short-term graph, the global long-term graph (1) aims to model user long-term behaviors, (2) contains far more internal and external behaviors, and (3) might have uncontrollable and delayed behavior acquisitions. To make a compromise between efficiency, effectiveness, and robustness, we conduct a multi-hop neighbor-similarity based loss instead of the online temporal MAML.

We assume that both users' and items' long-term representations $\bar{\boldsymbol{u}}_i^l$ and $\bar{\boldsymbol{d}}_j^l$ learned in Sec. 3.3 should be similar to their k-hop neighbors on the global long-term graph enhanced from [11] and [30]. The multi-hop neighbor-similarity based loss on the global user-item graph can be viewed as an extended matrix factorization (MF) model, which considers multi-hop user-item paths on the global graph as multi-source user/item correlations. Precisely, we consider the 10-hop neighbors via DeepWalk based path sampling [18] to bring in more interactions via users' multi-source behaviors. We formalize our k-hop neighbor-similarity based loss L_N as follows:

$$L_N = -\sum_{p\in P}\sum_{q_i,q_j\in p}\left(\log(\sigma(\bar{\boldsymbol{q}}_i^{l\top}\bar{\boldsymbol{q}}_j^l))\right). \tag{8}$$

p is a k-length random path in the path set P generated by DeepWalk. $q_i, q_j \in p$ are different nodes in the path p. We have $\bar{\boldsymbol{q}}_i^l = \bar{\boldsymbol{u}}_i^l$ for user nodes and $\bar{\boldsymbol{q}}_j^l = \bar{\boldsymbol{d}}_j^l$ for item nodes. σ is the sigmoid function. The multi-hop neighbor-similarity based loss focuses more on the global view of user and item representations learned from all long-term internal/external behaviors. Generally, the global long-term graph trains far less frequently than the short-term graph considering its motivation and training efficiency.

The advantages of using the multi-hop neighbor-similarity based loss for the global long-term graph are as follows: (1) the L_N loss is simple, efficient, and effective, which can directly optimize the cross-source interactions via the multi-hop connections. (2) Based on the neighbor-similarity based loss, it is more convenient to introduce other heterogeneous nodes (e.g., content or tag in [30]) and their interactions in this work. Other node representation learning methods are also easy to be adopted in our framework.

3.5.3 *Overall loss.*

The overall loss L is the weighted aggregation of these two losses L_T and L_N as follows:

$$L = \lambda_T L_T + \lambda_N L_N. \tag{9}$$

We empirically set $\lambda_T = \lambda_N = 1$. In LSTTM, the neighbor-similarity based loss L_T works as an auxiliary task for the temporal MAML loss L_T, since L_T is directly related to the ranking objectives.

The advantages of our asynchronous optimization are listed as follows: (1) it decouples the short-term and long-term preference modeling, making both modules more flexible, specialized, and robust to capture different user preferences. (2) It proposes an industrial solution to jointly consider large-scale external and internal behaviors, improving the robustness against the high disturbances and uncontrollability in practical systems (e.g., the delay and noises of external behaviors have little influence on the internal short-term preference modeling). (3) The asynchronous optimization is flexible and easy to deploy. It is also convenient to use other representation learning models in this asynchronous framework.

4 ONLINE DEPLOYMENT

Online System. We have deployed LSTTM on a real-world recommendation system of WeChat. This online recommendation system is a feed stream that has nearly million-level users and daily views. It contains heterogeneous domains, including news and videos. User behaviors of the target recommendation task are viewed as the internal behaviors. After user approvals, other behaviors (e.g., clicks in other recommendation domains) linked by the same user account in the platform are regarded as the external behaviors, which also provide additional information to reflect user preferences.

Online Serving. We conduct an asynchronous optimization and online updating for different modules. In offline training, we conduct a daily complete training to update all modules via the asynchronous optimization of temporal MAML and neighbor-similarity based losses on all behaviors. It constructs an industrial balance between effectiveness and efficiency, since the access to user external behaviors is usually delayed, and the full training on billion-level global graphs cannot be conducted in real-time. In online serving, the global graph module is fixed within a day for modeling long-term preferences, while the internal graph and fusion modules are frequently updated (according to the online computing capability) to capture user short-term preferences. When recommending at the t-th time period, we consider *all previous* $t-1$ *time periods* in this day as the support set, simulating the offline temporal MAML training. Hence, we *just need to conduct the general one-step gradient updates on new user behaviors*, regarding them as the support set of the current recommendation. It enables a fast online learning since the online time complexity of temporal MAML is equivalent to the classical one-step fine-tuning.

Online Efficiency. We train our model once over all training instances in online updating considering the efficiency. The online computation does not involve the global long-term graph modeling. The online time complexity of LSTTM is $O(k(T_i + T_f))$, where k is the number of candidates (e.g., top 200 items retrieved by the previous matching module). T_i and T_f represent the computation costs of the internal short-term graph (2-layer GAT with the dynamic temporal neighbors) and the fusion. For the online memory cost, the model should store the temporal MAML model and the fixed user long-term representations. Specifically, we implement LSTTM on a self-developed distributed deep learning framework. We have 30 parameter servers and 20 workers for training. Each server has 10G memory with 3 CPUs, and each worker has 10G memory with 5 CPUs. We spend nearly 4 hours for daily complete retraining.

5 EXPERIMENTS

In this section, we conduct experiments to answer the following research questions: (**RQ1**): How does LSTTM perform in offline temporal CTR prediction that simulates practical scenarios (Sec. 5.4)? (**RQ2**): How does LSTTM perform in online A/B tests (Sec. 5.5)? (**RQ3**): What are the effects of different components (Sec. 5.6)?

5.1 Dataset

Since there is no large-scale real-world dataset that contains both hourly-updated hot spots and user external behaviors, we build a new dataset NewsRec-21B extracted from a widely-used news recommendation system in WeChat. Precisely, we randomly select 58 million users and get nearly 1 billion user internal behaviors with timestamps in the target news domain. We also use these users' 20 billion external click behaviors from other recommendation domains in the same platform after user approval to build the global long-term graph. These internal and external behaviors are in the same platform, which are linked via the shared user accounts. All data are preprocessed via data masking to protect user privacy. The instances in the former eight days are regarded as the train set, and the last day's internal behaviors are considered as the test set. We follow Sec. 3.2 and Sec. 3.3 to build two huge graphs with the train set. Table 1 shows the detailed statistics of NewsRec-21B.

#user	#item	#internal	#external
58,284,406	626,736	1,022,589,888	20,087,883,624

Table 1: Statistics of the NewsRec-21B dataset.

5.2 Competitors

We implement several competitive baselines for evaluation. First, we conduct four widely-used ranking models as follows:

- **FM [19].** FM is a simple and effective model that captures second-order feature interactions via latent vectors.
- **NFM [8].** NFM combines the neural FM layer with the DNN layer sequentially to model high-order feature interactions.
- **DeepFM [7].** DeepFM follows the Wide&Deep framework and improves the Wide part with a neural FM layer. It is also used in the long-/short- term gating fusion of LSTTM.
- **AutoInt [20].** AutoInt is a strong feature interaction modeling method, which adopts self-attention layers.

These baselines use the same features of the users, internal behaviors and contexts that are also used in LSTTM, and are optimized via the same training set with the cross-entropy loss.

For fair comparisons, we also implement two enhanced DeepFM models armed with external behaviors and sequence modeling.

- **DeepFM (+external).** We add the features of user external behaviors to DeepFM, noted as DeepFM (+external). It has the same input features as the LSTTM model.
- **DIN+DeepFM (+external).** Based on DeepFM (+external), we further bring in the ability of sequence-based modeling

on user's internal and external behaviors to better model the short-term and long-term preferences. Inspired by [29], we conduct two DIN encoders [37] to model internal and external behaviors respectively. These behavior features are considered as the input feature fields of DeepFM (+external).

Finally, since we conduct the temporal MAML for online updating, we also implement two SOTA meta-learning methods based on SML [35] in online news recommendation as follows:

- **SML [35].** SML is the SOTA meta-learning based recommendation model designed for model retraining verified in online new recommendation [35]. It is the most related baseline of our task. SML attempts to learn a transfer function from old to new parameters via a sequential training over time. Following the original SML's implementation, we also build SML based on an MF model.
- **SML+DeepFM.** We further improve the original SML by replacing the MF model with the best performing DeepFM model, noted as SML+DeepFM. This model also utilizes the same features as LSTTM for fair comparisons.

Note that we do not compare with other meta-learning recommendation methods such as MeLU [10], since they focus on different tasks (e.g., cold-start users or domains) and are not suitable for our temporal setting. To further verify the effectiveness of different components and features in LSTTM, we implement four ablation versions of LSTTM, whose results are discussed in Sec. 5.6.

5.3 Experimental Settings

We randomly select up to 30 neighbors in global graph and 30 most recent behaviors in internal graph for aggregation. The dimensions of the output embeddings in both graphs are 16. We use 6 fields for users (e.g., user profiles such as age and gender) and items (e.g., item features such as tag and provider), and the dimension of each trainable feature field embedding is 16. In temporal MAML, the task number of each batch, batch size, and learning rate are essential parameters. We have tested the task number among $\{4, 8, 16\}$, the support and query set size among $\{32, 64, 128, 256\}$, and the learning rate among $\{0.001, 0.01, 0.02\}$. Finally, we let each batch size contain 8 temporal tasks of different days and hours, where each support set and query set contain 128 items. We use Adagrad [5] and empirically set the same learning rate as 0.01 for inner and outer updates. Note that all instances could belong to a certain query set used in Line 8, Algorithm 1, which directly updates model parameters. We only conduct a one-step gradient in temporal MAML for online efficiency. We conduct a grid search for parameter selection. All models share the same experimental settings.

5.4 Temporal CTR Prediction (RQ1)

We first simulate the real-world online recommendation and conduct the temporal CTR prediction task for offline evaluation.

5.4.1 Evaluation Protocol. We evaluate models on our real-world dataset NewsRec-21B. To simulate the online recommendation, we first train all models with the train set (the former few days), and divide the test set (the last day) into 24 hours. Each hour is regarded as a temporal task for evaluation, with all instances of former hours in the test set used as the support set. Considering both accuracy and online efficiency, all models including LSTTM and baselines are fine-tuned via one gradient update for fair comparisons (SML is updated via its transfer method [35]). We use AUC as our metric, which is widely utilized as the main metric in real-world systems [7, 20, 37]. For a better display, we group 24 hours into 3 periods (0:00-8:00, 8:00-16:00, 16:00-0:00+1), and report the average AUC in each period in Table 2. We conduct 3 runs for each model.

AUC	period1	period2	period3
FM (Rendle 2010)	0.8086	0.8021	0.8072
NFM (He and Chua 2017)	0.8140	0.8035	0.8154
DeepFM (Guo et al. 2017)	0.8253	0.8306	0.8296
AutoInt (Song et al. 2019)	0.8236	0.8282	0.8272
DeepFM (+external)	0.8282	0.8335	0.8327
DIN+DeepFM (+external)	0.8287	0.8346	0.8337
SML (Zhang et al. 2020)	0.7926	0.7991	0.8063
SML+DeepFM	0.8250	0.8307	0.8299
LSTTM (w/o Meta)	0.8373	0.8428	0.8421
LSTTM (final)	**0.8395**	**0.8504**	**0.8502**

Table 2: Results of the temporal CTR prediction task. The improvements of LSTTM are significant (t-test with p<0.01). Note that period1 focuses on the performances of models in a short time after complete training, while period3 focuses on the performances in a long time after complete training.

5.4.2 Experimental Results. From Table 2 we can observe that:

(1) LSTTM achieves significant improvements on all baselines in three periods, with the significance level $\alpha = 0.01$. It consistently outperforms strong baselines in all 24 hours (see Fig. 3). The deviation is less than ± 0.002. Considering the large size of our test set, the $1.1\% - 1.7\%$ AUC improvements over the best baseline are impressive and solid. It verifies the effectiveness and robustness of LSTTM in modeling both short-term and long-term preferences from users' internal and external behaviors.

(2) LSTTM (final) consistently outperforms LSTTM (w/o Meta) and SML on all tasks. It confirms the advantages of temporal MAML in Sec. 3.5.1. Thanks to the MAML-based training, LSTTM is more sensitive to global new trends in communities. Hence, it can better capture users' short-term preferences via good model initialization, and thus can fast adapt to hot topics over time in online recommendation. Nevertheless, LSTTM (w/o Meta) still performs better than baselines, which reflects the effectiveness of our global long-term and internal short-term graphs as well as the gating fusion. Sec. 5.6 gives more details of different ablation versions.

(3) We also find that models armed with external behaviors consistently outperform the same models without external behaviors (e.g., see LSTTM in Sec. 5.6 and DeepFM in Table 2). It verifies the importance of external behaviors in real-world scenarios, which works as a strong supplement to the internal behaviors. The external behaviors will be more significant in few-shot scenarios.

(4) Comparing models in different periods, we know that LSTTM achieves larger improvements in period 2 and 3 compared to LSTTM (w/o Meta). It is because that (a) humans and hot spots are often more active in period 2 and 3, where temporal MAML is superior

to baselines in capturing user real-time preferences. (b) In period3, all models have not been fully trained for at least 16 hours. LSTTM has a better online fine-tuning to catch up with new global interest evolutions. The cumulative effects of temporal MAML will gradually show up over time with growing hot topics. Fig. 3 shows the hour-level AUC trends of four representative models.

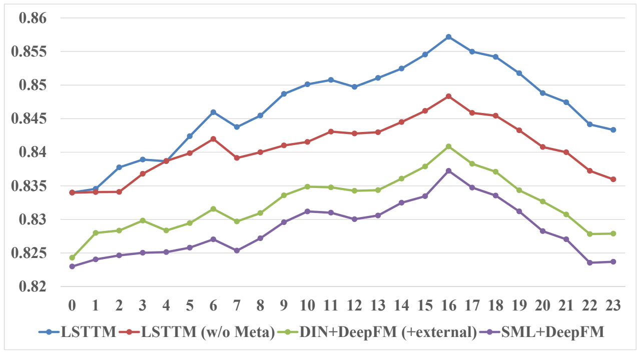

Figure 3: AUC trends in different hours.

5.5 Online A/B Tests (RQ2)

5.5.1 Evaluation Protocol. To evaluate LSTTM in real-world systems, we conduct an online A/B test on WeChat Top Stories. Following Sec. 4, we deploy our LSTTM in the ranking module of the news domain with other modules unchanged. The online base model is DeepFM with the same online update frequency of LSTTM. In online evaluation, we focus on four representative metrics, including Click-through-rate (CTR), average click number per capita (ACN), user has-click rate (HCR), and average dwell time per capita (DT) to measure recommendation accuracy and user satisfaction, which are formalized as follows:

$$\text{CTR} = \frac{\#\text{ of all clicks}}{\#\text{ of all impressions}}, \quad \text{ACN} = \frac{\#\text{ of all clicks}}{\#\text{ of all users}}, \quad \text{HCR} = \frac{\#\text{ of users having clicks}}{\#\text{ of all users}}, \quad \text{DT} = \frac{\text{all duration}}{\#\text{ of all users}}. \tag{10}$$

We conduct the A/B test for 5 days.

5.5.2 Experimental Results. Table 3 shows the improvement percentages over the base model. We can observe that:

(1) LSTTM achieves significant improvements on all metrics with the significance level $\alpha = 0.01$. It reconfirms the effectiveness of LSTTM in online. Through the asynchronous online updating with the temporal MAML, LSTTM can (a) fast adapt to new topics and hot spots, and (b) successfully combine both external and internal behaviors in online ranking without many computation costs.

(2) The improvement on CTR indicates that more appropriate items have been impressed to users (reflecting item-aspect accuracy), while the improvement on ACN represents that users are more willing to click items (reflecting user-aspect accuracy and activeness). HCR models the coverage of users that have clicked news, which implies the impacts of our recommendation function. LSTTM also outperforms the online baseline on dwell time of items, which reflects the real user satisfaction on the item contents. In conclusion, LSTTM achieves comprehensive improvements on all online metrics, which confirms the robustness of our model.

metrics	CTR	ACN	HCR	DT
LSTTM	+9.60%	+9.93%	+3.42%	+2.52%

Table 3: Online A/B tests on a widely-used system.

5.6 Ablation Tests (RQ3)

We further conduct an ablation test to verify the effectiveness of different components in LSTTM. Table 4 shows the results of different ablation settings. We observe that all components significantly benefit the recommendation. Precisely, we find that:

(1) the temporal MAML can precisely capture user's variable short-term interests without additional online computation costs. The improvements are larger when more new trends are involved as time passes by, such as in period 2 and 3.

(2) The second ablation version only considers internal behaviors by removing the global long-term graph. It verifies the effectiveness of the user external behaviors as well as the global long-term graph modeling in Sec. 3.3. The advantages of external behaviors will be more significant if we deploy LSTTM on cold-start scenarios.

(3) The gating-based fusion is also effective compared to concatenation, which provides personalized strategies in combining internal short-term and global long-term representations.

(4) The fourth ablation version removes the GAT-based aggregation in Eq. (2) and the multi-hop neighbor-similarity based loss in Eq. (8) (only use the raw features of internal and external behaviors as inputs). The GAT-based aggregation and the multi-hop neighbor-similarity based loss enable more sufficient multi-domain user-item interactions, which are beneficial in capturing user variable and diverse preferences in practice.

models	period1	period2	period3
LSTTM (final)	**0.8395**	**0.8504**	**0.8502**
– temporal MAML	0.8373	0.8428	0.8421
– user external behaviors	0.8385	0.8459	0.8459
– gating-based fusion	0.8378	0.8488	0.8489
– GAT aggregation & L_N	0.8304	0.8410	0.8411

Table 4: Ablation tests on NewsRec-21B.

6 CONCLUSION AND FUTURE WORK

In this work, we propose an LSTTM for online recommendation, which captures user long-term and short-term preferences from internal/external behaviors. The temporal MAML enables fast adaptations to new topics in recommendation. The effectiveness of LSTTM is verified in offline and online real-world evaluations.

In the future, we will polish the temporal MAML to build a more robust adaptation, and transfer the idea of temporal MAML to other temporal tasks. We will also explore some enhanced combinations with other online learning and meta-learning methods.

REFERENCES

[1] Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long-and short-term user representations. In *Proceedings of ACL*.

[2] Homanga Bharadhwaj. 2019. Meta-Learning for User Cold-Start Recommendation. In *Proceedings of IJCNN*.

[3] Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. 2020. MAMO: Memory-Augmented Meta-Optimization for Cold-start Recommendation. In *Proceedings of KDD*.

[4] Zhengxiao Du, Xiaowei Wang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Sequential Scenario-Specific Meta Learner for Online Recommendation. In *Proceedings of KDD*.

[5] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR* (2011).

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of ICML*.

[7] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of IJCAI*.

[8] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of SIGIR*.

[9] Linmei Hu, Chen Li, Chuan Shi, Cheng Yang, and Chao Shao. 2020. Graph neural news recommendation with long-term and short-term interest modeling. *Information Processing & Management* (2020).

[10] Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. 2019. MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation. In *Proceedings of KDD*.

[11] Qi Liu, Ruobing Xie, Lei Chen, Shukai Liu, Ke Tu, Peng Cui, Bo Zhang, and Leyu Lin. 2020. Graph Neural Network for Tag Ranking in Tag-enhanced Video Recommendation. In *Proceedings of CIKM*.

[12] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of KDD*.

[13] Zhaoyang Liu, Haokun Chen, Fei Sun, Xu Xie, Jinyang Gao, Bolin Ding, and Yanyan Shen. 2020. Intent Preference Decoupling for User Representation on Online Recommender System. In *Proceedings of IJCAI*.

[14] Yuanfu Lu, Yuan Fang, and Chuan Shi. 2020. Meta-learning on Heterogeneous Information Networks for Cold-start Recommendation. In *Proceedings of KDD*.

[15] Mi Luo, Fei Chen, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Jiashi Feng, and Zhenguo Li. 2020. MetaSelector: Meta-Learning for Recommendation with User-Level Adaptive Model Selection. In *Proceedings of WWW*.

[16] Yadan Luo, Zi Huang, Zheng Zhang, Ziwei Wang, Mahsa Baktashmotlagh, and Yang Yang. 2020. Learning from the Past: Continual Meta-Learning with Bayesian Graph Neural Networks. In *Proceedings of AAAI*.

[17] Feiyang Pan, Shuokai Li, Xiang Ao, Pingzhong Tang, and Qing He. 2019. Warm up cold-start advertisements: Improving ctr predictions via learning to learn id embeddings. In *Proceedings of SIGIR*.

[18] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of KDD*.

[19] Steffen Rendle. 2010. Factorization machines. In *Proceedings of ICDM*.

[20] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of CIKM*.

[21] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of CIKM*.

[22] Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. 2017. A meta-learning perspective on cold-start recommendations for items. In *Proceedings of NIPS*.

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of ICLR*.

[24] Ricardo Vilalta and Youssef Drissi. 2002. A perspective view and survey of meta-learning. *Artificial intelligence review* (2002).

[25] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of SIGIR*.

[26] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based Recommendation with Graph Neural Networks. In *Proceedings of AAAI*.

[27] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. 2010. Temporal recommendation on graphs via long-and short-term preference fusion. In *Proceedings of KDD*.

[28] Chaojun Xiao, Ruobing Xie, Yuan Yao, Zhiyuan Liu, Maosong Sun, Xu Zhang, and Leyu Lin. 2021. UPRec: User-Aware Pre-training for Recommender Systems. *arXiv preprint arXiv:2102.10989* (2021).

[29] Ruobing Xie, Cheng Ling, Yalong Wang, Rui Wang, Feng Xia, and Leyu Lin. 2020. Deep Feedback Network for Recommendation. In *Proceedings of IJCAI*.

[30] Ruobing Xie, Qi Liu, Shukai Liu, Ziwei Zhang, Peng Cui, Bo Zhang, and Leyu Lin. 2021. Improving Accuracy and Diversity in Matching of Recommendation with Diversified Preference Network. *IEEE Transactions on Big Data* (2021).

[31] Ruobing Xie, Zhijie Qiu, Jun Rao, Yi Liu, Bo Zhang, and Leyu Lin. 2020. Internal and Contextual Attention Network for Cold-start Multi-channel Matching in Recommendation. In *Proceedings of IJCAI*.

[32] Ruobing Xie, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Hierarchical Reinforcement Learning for Integrated Recommendation. In *Proceedings of AAAI*.

[33] Zeping Yu, Jianxun Lian, Ahmad Mahmoody, Gongshen Liu, and Xing Xie. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation.. In *Proceedings of IJCAI*.

[34] Zheni Zeng, Chaojun Xiao, Yuan Yao, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2021. Knowledge transfer via pre-training for recommendation: A review and prospect. *Frontiers in big Data* (2021).

[35] Yang Zhang, Fuli Feng, Chenxu Wang, Xiangnan He, Meng Wang, Yan Li, and Yongdong Zhang. 2020. How to Retrain Recommender System? A Sequential Meta-Learning Method. In *Proceedings of SIGIR*.

[36] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of AAAI*.

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

[38] Yongchun Zhu, kaikai Ge, Fuzhen Zhuang, Ruobing Xie, Dongbo Xi, Xu Zhang, Leyu Lin, and Qing He. 2021. Transfer-Meta Framework for Cross-domain Recommendation to Cold-Start Users. In *Proceedings of SIGIR*.

[39] Yongchun Zhu, Yudan Liu, Ruobing Xie, Fuzhen Zhuang, Xiaobo Hao, Kaikai Ge, Xu Zhang, Leyu Lin, and Juan Cao. 2021. Learn to Expand Audience via Meta Hybrid Experts and Critics for Recommendation and Advertising. In *Proceedings of KDD*.

[40] Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. 2021. Learning to Warm Up Cold Item Embeddings for Cold-start Recommendation with Meta Scaling and Shifting Networks. In *Proceedings of SIGIR*.

[41] Daniel Zügner and Stephan Günnemann. 2019. Adversarial attacks on graph neural networks via meta learning. In *Proceedings of ICLR*.