Sequential Recommender via Time-aware Attention Memory Network

Wendi Ji, Keqiang Wang, Xiaoling Wang, Tingwei Chen, Alexandra I. Cristea
Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China
Pingan Health Technology
Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai 200092, China
Department of Computer Science, Liaoning University, Shenyang 110036, China
Department of Computer Science, Durham University, Durham DH1 2GY, United Kingdom
wendyg8886@gmail.com,wangkeqiang265@pingan.com.cn,xlwang@cs.ecnu.edu.cn
tingwei.chen,alexandra.i.cristea@durham.ac.uk

ABSTRACT

Recommendation systems aim to assist users to discover desirable contents from an ever-growing corpus of items. Although recommenders have been greatly improved by deep learning, they still face several challenges: (1) behaviours are much more complex than words in sentences, so traditional attention and recurrent models have limitations capturing the temporal dynamics of user preferences. (2) The preferences of users are multiple and evolving, so it is difficult to integrate long-term memory and short-term intent.

In this paper, we propose a temporal gating methodology to improve the attention mechanism and recurrent units, so that temporal information can be considered for both information filtering and state transition. Additionally, we propose a hybrid sequential recommender, named **M**ulti-hop **T**ime-aware **A**ttentive **M**emory network (**MTAM**), to integrate long-term and short-term preferences. We use the proposed time-aware GRU network to learn the short-term intent and maintain prior records in user memory. We treat the short-term intent as a query and design a multi-hop memory reading operation via the proposed time-aware attention to generate user representation based on the current intent and long-term memory. Our approach is scalable for candidate retrieval tasks and can be viewed as a non-linear generalisation of latent factorisation for dot-product based Top-K recommendation. Finally, we conduct extensive experiments on six benchmark datasets and the experimental results demonstrate the effectiveness of our MTAM and temporal gating methodology.

CCS CONCEPTS

• **Information systems** → **Information retrieval**.

KEYWORDS

Sequential Recommendation, User Modeling, Memory Networks, Time-aware Attention Mechanism, Time-aware Recurrent Unit

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3411869

ACM Reference Format:
Wendi Ji, Keqiang Wang, Xiaoling Wang, Tingwei Chen, Alexandra I. Cristea. 2020. Sequential Recommender via Time-aware Attention Memory Network. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3411869

1 INTRODUCTION

In large-scale recommendation systems, it is challenging to retrieve a set of most relevant items for a user given her/his interaction history from tens or hundreds of millions of items. A common recipe to handle the huge amount and sparsity of item corpus is matrix factorisation, which facilitates efficient approximate k-nearest neighbor searches via resorting to the inner product of user representation and item representation [2, 26–28].

Typically, there are two stages in an industrial recommendation system: candidate generation and ranking [3, 6, 35]. At the candidate generation stage, time-efficient neural nominators retrieve hundreds of candidates from a large corpus of items. The candidates are then re-ranked by a fully-blown neural ranking model. The main difference between the two stages is that a ranking model can serve as a discriminator which predicts $score(u, i)$ (the preference of user u on item i) on a small candidate set, while candidate generators are required to generate the representation of a target user which can be used in k-nearest neighbour searches. As shown in Figure 1, we focus on the candidate generation stage which determines the ceiling performance of recommendation and treat it as a user modeling task.

Analogous with words of sentences, a user's interactions with items naturally form a behaviour sequence. With the quick development of deep learning, many recent researches have built recurrent and attention models to capture the sequential property of user behaviours [3, 15, 20].

A vital challenge of user modeling is that user behaviours are much more complex than words. Some context features of a behaviour, like category, action and text information, can be incorporated by injecting feature embeddings into the item embedding. However, the temporal feature is related to a pair of behaviours. Two behaviours within a short interval intuitively tend to be more relevant than two behaviours within a long interval. Therefore, classical structures of recurrent and attention networks need to be upgraded to model the temporal dynamics of sequential data better.

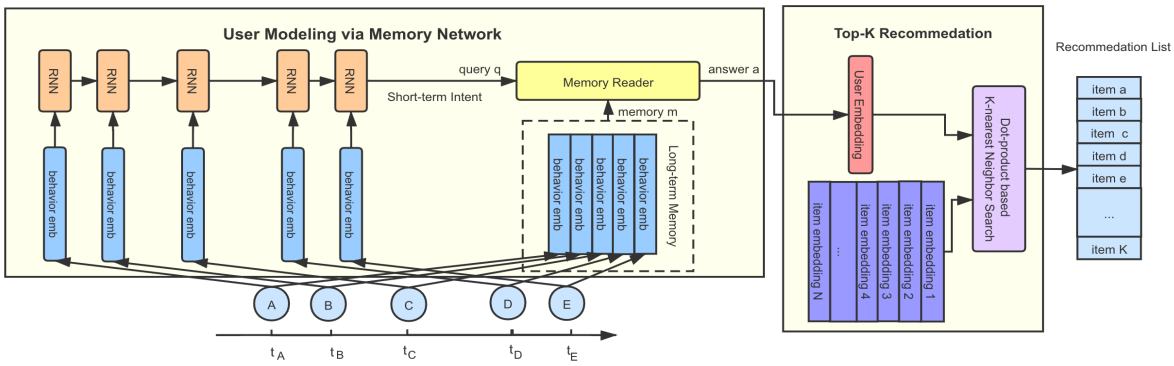


Figure 1: The general framework of MTAM for Top-K recommendation. The left part shows the user modeling module, which encodes user's short-term intent and long-term memory into the final user embedding via a memory network. And the right part shows the Top-K recommendation module, which produces a ranking list over all items via the dot-product based K-nearest neighbor search.

Some recent researches improve GRU or LSTM units by adding time gates to capture the temporal information of user behaviour sequences [5, 37, 40]. However, the temporal information is often ignored in attention mechanism. Originated from the NLP field, an attention mechanism takes a weighted sum of all components and focuses only on the information related to a query. Many recent neural recommenders utilise attention mechanism to filter diverse user interests, by concentrating on relevant behaviours and eliminating irrelevant ones to predict a user's future action [10, 20, 23, 30, 36, 39]. Among these attention neural recommenders, researches [10, 20] assign weights to compressed hidden states to build attention RNN models; assign weights to historical records to build attention memory networks [23, 30]; and assign weights to hidden variables to build deep attention feed-forward networks [36, 39]. Nevertheless, when calculating the correlation between two behaviours, the time interval between them has not been taken into consideration by most of the previous approaches. One recent research [21] addresses this challenge by directly adding the clipped relative time intervals to the dot-product of item embeddings when calculating the attention weights. However, how to upgrade attention mechanism to model the temporal patterns of sequences in a more fine-grained way is still a problem.

To overcome the aforementioned limitations of neural networks, we propose a *novel temporal gating methodology to upgrade attention mechanism and recurrent units by taking advantage of the time-aware distance between interactions in the task of user modeling.* Inspired by the gated mechanisms in LSTM and GRU, we introduce a temporal controller to encode the temporal distance between two interactions into a gate. We then propose a novel time-aware attention by equipping scaled dot-product attention with the proposed temporal gate. When calculating the relationship between two interactions, the time-aware attention kernel is capable to take the time interval into account. Meanwhile, for better short-term preference modeling, we utilise the proposed temporal gate in GRU to control how much past information can be transferred to future states, named T-GRU. Different from previous time-aware recurrent recommenders which aim to capture both the short-term and long-term interests [5, 37, 40], we only aim at short-term intent by filtering out irrelevant past information with the temporal gate.

Additionally, both long-term preferences and short-term intents determine the behaviours of users. In this paper, we view user behaviours as a decision making program in Memory Network and propose a **M***ulti-hop* **T***ime-aware* **A***ttentive* **M***emory* (*MTAM* for short) network based on the proposed T-GRU and time-aware attention, which is illustrated in Figure 1. MTAM first utilises T-GRU to capture the short-term intent of a user and maintains a fixed length history of behaviours in a memory matrix (memory m) to store her/his long-term preference. Inspired by the memory retrieval procedure in the human mind, MTAM treats the short-term intent as a query q to search throughout the long-term memory m. The searching procedure of MTAM is to find a continuous representation for m and q via time-aware attention. When making recommendations, MTAM triggers the searching procedure for multi-hops to output an answer a, which is a comprehensive representation of the target user. The time-aware attention mechanism of MTAM provides an effective manner to learn the temporal dynamics of behaviour sequences by crediting the different contributions of prior interactions to the current decision. The proposed MTAM can be viewed as a non-linear generalisation of factorisation techniques and applied in large-scale retrieval systems. We will show experimentally that the temporal gating methodology and the multi-hop structure are crucial to good retrieval performance of MTAM in the Top-K recommendation task on 6 real-world datasets.

In summary, the main contributions of the paper can be illustrated as follows:

- We improve the attention mechanism and recurrent unit via a novel temporal gating methodology to capture the temporal dynamics of users' sequential behaviours.
- We propose a novel multi-hop time-aware memory neural network for sequential recommendation. MTAM treats the output of T-GRU as short-term intent and reads out the long-term memory effectively via time-aware attention. To the best of our knowledge, MTAM is the first memory network which takes the time-aware distances between items into account.
- We compare our model with state-of-the-art methods on six real-world datasets. The results demonstrate that the performance of Top-K recommendation is obviously improved

via adding the temporal gate into the recurrent unit and attention mechanism. Additionally, compared to encoding the user representation with the weighted sum of recurrent hidden states, MTAM is able to leverage user historical records in a more effective manner.

2 RELATED WORK

Our work focuses on the candidate generation stage and is essentially a memory-augmented sequential recommender. We will review the related works in three directions.

2.1 Candidate Generation and Ranking

In general, an industrial recommendation system consists of two stages: candidate generation and candidate ranking [3, 6, 35]. The candidate generation (a.k.a. retrieval or nomination) stage aims to provide a small set of related items from a large corpus under stringent latency requirements [6, 35]. Then, the candidate ranking model reranks the retrieved items based on click-through rate (CTR for short), rating or score [10, 30, 34]. In the retrieval stage, recommenders have to face the computational barriers of full corpus retrieval. A common recipe for candidate retrieval is modeling the user-item preference as the dot-product of the low dimensional user representation and item representation, such as matrix factorisation [18, 26] and neural recommenders [3, 6, 20]. However, the dot-product correlation limits the capability of neural recommenders. To learn deeper non-linear relationships between a target user and candidate items, some more expressive models have been proposed for the ranking stage, such as neural collaborative filtering [14], deep interest network [39], SLi-Rec [37] and user memory network [4]. In this paper, we focus on the candidate generation stage and the proposed MTAM can be viewed as a non-linear generalisation of factorisation techniques.

2.2 Sequential Recommenders

Analogous with words of sentences in natural language processing (NLP), a user's interactions with items naturally form a behaviour sequence. In recent years, deep neural networks have achieved continuous improvements in NLP [7, 11, 25, 31], which prompt a series of explorations in applying neural networks in sequential recommendation. The first stab at employing RNN-based models in session-based recommendation [15] uses GRU to model the click sequences and improves the CTR prediction by taking the sequential characteristics into consideration. Additionally, a user's purchase decision is both determined by her/his long-term stable interests and short-term intents [1, 8, 36]. Other researches [16, 20, 23, 39] combine RNNs with the attention mechanism to learn the preference evolution of users.

However, although traditional attention and recurrent models have shown excellent performance to model the sequential patterns of user behaviours, they only consider the orders of objects, without the notion of the temporal information. The time intervals between interactions are important to capture the correlations between these interactions. [40] improves LSTM by proposing some temporal gates to capture both long-term and short term preferences of users. Recent researches [5, 37] further propose two time-aware recurrent units. The main difference between our proposed T-GRU and previous models is we only use the time interval between adjacent interaction to control how much past information can be transferred to future states, while previous researches apply temporal gates to control both previous information and current content. Furthermore, how to capture the temporal context in attention mechanism is still not well explored. The lastest research [21] explores the influence of different time intervals on t next-item prediction and proposes a time-aware self-attention model. It treats time intervals as special positions and solves it by adding clipped intervals to the dot-product of item embeddings. In our work, we further update the attention mechanism by a gating technique which helps to capture the non-linear temporal differences between interactions.

2.3 Memory-augmented Recommenders

An essential challenge of user modeling is learning the dependencies of behaviours. RNN-based models encode user's previous behaviours into hidden states. Although an attention mechanism helps recurrent networks to learn long-term dependencies by concentrating on the relevant states [20, 32], it fails to distinguish between the different roles that each item plays in prediction. To tackle this challenge, external memory networks have been proposed in recent years to store and manipulate sequences effectively [12, 13], which have been successfully adapted to NLP tasks, such as question answering [19], knowledge tracing [38], translation [24] and dialogue systems [33]. Several recent researches propose memory-augmented neural networks for recommendation to leverage users' historical behaviours in a more effective manner [4, 9, 23, 30]. They introduce an external user memory to maintain users' historical information and use attention mechanism to design memory reading operations. Among these researches, [4, 9, 30] treat the target item as the query, while [23] treats the last item in the interaction sequence as the query.

There are two main differences between these existing researches and the proposed MTAM. (1) Taking advantage of the proposed time-aware attention, MTAM is the first memory network which takes the temporal context of interactions into consideration. (2) Apart from [23, 32], most previous memory-augmented recommenders focus on the candidate ranking stage, which aims to predict the $score(u, i)$. Meanwhile, MTAM focuses on the candidate retrieval stage, which aims to generate the representation of a target user merely based on her/his historical records.

3 OVERALL FRAMEWORK

We first provide the formal notations that will be used in this paper and define the task of sequential recommendation. Then we describe the multi-hop time-aware attention memory recommender overall.

3.1 Preliminaries

Suppose there are M users and N items in the system. The behaviour sequence of user u is $\mathbf{S}_u = (b_{u,1}, b_{u,2}, ..., b_{u,|\mathbf{S}_u|})$. We denote a behaviour $b_{u,i} = (x_{u,i}, t_{u,i}, e_{u,i})$ as the i-th interaction in sequence $\mathbf{S}_u$, where $x_{u,i} \in \mathbf{V}$ is the item that user u interacts with at time $t_{u,i}$ and $e_{u,i}$ presents the contextual information. The contextual information $e_{u,i}$ can include various kinds of important features, e.g. item category, behaviour position, location, duration and action. Then given the historical behaviour sequence $\mathbf{S}_u = (b_{u,1}, b_{u,2}, ..., b_{u,i})$ of a specified user u, the time-aware sequential recommendation is to predict the next item $x_{u,i+1}$ that user u will interact with at time

t_{target}. Since the corpus of items is large in an industrial recommendation system, a nominator in the candidate generation stage needs to make more than one recommendation to the user, which represent the so-called Top-K recommendations.

We aim to build a time-aware recommender **Rec** so that for any prefix behaviour sequence $\mathbf{S}_u = (b_{u,1}, b_{u,2}, ..., b_{u,i})$ and a target time t_{target}, we obtain the output $y = \mathbf{Rec}(\mathbf{S}_u, t_{\text{target}})$. As illustrated in Figure 1, $y = (y_1, y_2, ..., y_K)$ is the ranking list for the Top-K recommendation ($0 < K \ll N$).

3.2 Recommendation with Attention Memory Network

User interests are both stable and evolving. In this paper, we propose a novel attention memory network for the task of Top-K recommendation, named Multi-hop Time-aware Attention Memory Network (MTAM for short). As illustrated in Figure 1, MTAM first encodes the user's short-term intent into a query q via a recurrent network based on our proposed T-GRU and maintains a fixed length of behaviour history as long-term preferences in memory m. Then, as shown in Figure 2, the prediction procedure is to read the long-term memory m for the current short-term intent q via our proposed time-aware attention mechanism. The output a of MTAM is a hybrid user representation which takes the advantage of both short-term and long-term components.

The proposed MTAM can be viewed as a non-linear neural generalisation of collaborative filtering based on factorisation techniques. Suppose there are M users and N items in a recommendation system. Let $\mathbf{P} \in \mathbb{R}^{M \times d}$ and $\mathbf{Q} \in \mathbb{R}^{N \times d}$ be the embedding matrices for users and items. For any prefix behaviour sequence $\mathbf{S}_u = (b_{u,1}, b_{u,2}, ..., b_{u,i})$ of user u, we first project items and contextual information into embedding spaces via the look-up function and attain $\mathbf{S}'_u = ((b'_{u,1}, t_{u,1}), (b'_{u,2}, t_{u,2}), ..., (b'_{u,i}, t_{u,i}))$. Our task is building a user model MTAM, of which the output is the embedding of user u at time t_{target}:

$$p_{u,t_{\text{target}}} = \text{MTAM}(\mathbf{S}'_u, t_{\text{target}}). \tag{1}$$

Then, as a neural matrix factorisation, a nearest neighbor search can be performed to generate the Top-K recommendations based on the dot-product similarity $p_{u,t_{\text{target}}}\mathbf{Q}^T$ between the predicted user embedding $p_{u,t_{\text{target}}}$ and the embeddings of all items $\mathbf{Q}$:

$$y = \mathbf{Rec}(\mathbf{S}_u, t_{\text{target}}) = \mathbf{Top\text{-}K}(p_{u,t_{\text{target}}}\mathbf{Q}^T) \tag{2}$$

where $y = (y_1, y_2, ..., y_K)$ is the ranking list of the K most relevant items. Our model can be trained by using a standard mini-batch gradient descent on the cross-entropy loss:

$$L(y', \hat{y}) = \sum_{u}^{M} y'_u \log \hat{y}_u, \tag{3}$$

where $y'_u = softmax(p_{u,t_{\text{target}}}\mathbf{Q}^T)$ is the predicted probability distribution of the next item and $\hat{y}_u$ the one-hot coding of the ground-truth next item $x_{u,i+1}$.

In the following two sections, we first introduce a general methodology to update the traditional attention mechanism and recurrent unit for user modeling, and propose the two basic components of MTAM: time-aware attention mechanism and T-GRU unit in Section 4. Then we illustrate the proposed multi-hop time-aware attention memory network which treats the short-term intent as key and reads out the long-term memory attentively for multiple hops in Section 5.

4 TEMPORAL GATING METHODOLOGY

In this section, we first provide a general idea of temporal gating methodology to capture the time-aware context in user behaviour sequences. Then we describe how to update the traditional attention mechanism and recurrent unit with it.

4.1 Temporal Gate

Different from the semantic correlations between words in NLP problems, the relationship between two interactions in a behaviour sequence is not only related to their relative positions, but also highly influenced by the time intervals.

We model the time interval as a temporal gate to encode the non-linear time difference between two interactions. A general form of the temporal gate between two behaviours can be defined as:

$$g_{ij} = f(t_{h_i} - t_{h_j}, h_i, h_j), \tag{4}$$

where (h_*, t_{h_*}) is the hidden representation and timestamp of an interaction. In this way, the temporal relationship between two interactions is determined by the time interval and their respective representations. Next we will introduce how to use the temporal gate to update recurrent networks and the attention mechanism.

4.2 Time-aware Recurrent Unit

Recurrent networks have led to great success in user modeling due to their remarkable ability to capture sequential patterns. The recurrent updating function of recurrent networks can be formulated as:

$$h_s = f(x_s, h_{s-1}), \tag{5}$$

where x_s is the current input and h_* is a hidden state. In practice, LSTM and GRU are the two most widely used recurrent units. The computational complexity of GRU is lower than LSTM by reducing one gate. In this paper, without loss of generality, we formulate the time-aware recurrent unit with GRU and the computation rules of the GRU unit can be illustrated as:

$$z_s = \sigma([x_s, h_{s-1}]W_z + b_z) \tag{6}$$

$$r_s = \sigma([x_s, h_{s-1}]W_r + b_r) \tag{7}$$

$$h'_s = \phi([x_s, h_{s-1} \odot r_s]W_h + b_h) \tag{8}$$

$$h_s = z_s \odot h_{s-1} + (1 - z_s) \odot h'_s, \tag{9}$$

where $x_s, h_{s-1}, h_s, h'_s, z_s, r_s \in \mathbb{R}^{1 \times d}$; $W_z, W_s, W_h \in \mathbb{R}^{2d \times d}$; $[\cdot, \cdot]$ is the concatenate operation; $+$ and $\odot$ denote element-wise add and multiplication operations; ϕ and σ are tanh and sigmoid activation functions; z_s and r_s are update gate and reset gate, respectively; while h'_s is the candidate state; h_{s-1} is the history state and h_s is the output hidden state. The output hidden state h_s emitted at state s is a linear interpolation between the history state h_{s-1} and the candidate state h'_s, where update gate z_s acts as a soft switch.

In order to capture the temporal correlations in a user behaviour sequence, we design a temporal gate to upgrade GRU, which is jointly determined by the current input x_s, the history state h_{s-1} and the time interval $t_s - t_{s-1}$, that is:

$$\delta_s = \phi(\log(t_s - t_{s-1} + 1) \odot W_{\delta_s} + \mathbf{b}_{\delta_s}), \tag{10}$$

$$\tau_s = \phi([x_s, h_{s-1}]W_{\tau_s} + b_{\tau_s}), \tag{11}$$

$$g_s = \sigma(\delta_s \odot W_{g_s\delta_s} + \tau_s \odot W_{g_s\tau_s} + b_{g_s}), \tag{12}$$

where $t_s, t_{s-1} \in \mathbb{R}$, $\delta_s, \tau_s, g_s \in \mathbb{R}^{1\times d}$, $W_{\delta_s}, W_{g_s\delta_s}, W_{g_s\tau_s} \in \mathbb{R}^{1\times d}$, $b_{\delta_s}, b_{\tau_s}, b_{g_s} \in \mathbb{R}^{1\times d}$ and $W_{\tau_s} \in \mathbb{R}^{2d\times d}$. The temporal feature δ_s encodes the temporal relation between adjacent interactions at state s. The semantic feature τ_s encodes the semantic context at state s. Temporal gate g_s is a non-linear combination of temporal and semantic features.

Then, we propose a novel recurrent unit, named T-GRU, by modifying the Eq. (9) to:

$$h_s = z_s \odot g_s \odot h_{s-1} + (1 - z_s) \odot h_s^{'}, \tag{13}$$

where the temporal gate g_s controls how much past information can be transferred to the current state. Compared to [37], which uses two temporal gates to control past and current information, the experimental results show that our proposed T-GRU performs better as a recurrent recommender on 4/6 datasets and MTAM dominates on all datasets where T-GRU serves as the short-term intent encoder.

4.3 Time-aware Attention

The scaled dot-product attention [31] is a popular attention kernel. Let $Q \in \mathbb{R}^{l^Q\times d}, K \in \mathbb{R}^{l^K\times d}, V \in \mathbb{R}^{l^V\times d}$ represent query, key and value, where l^Q, l^K, l^V are the number of items in query, key and value, respectively, and d is the latent dimension. The attention correlations between query Q and key K is computed as:

$$\text{Score}(Q, K) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right), \tag{14}$$

where the scale factor $\sqrt{d}$ is used to avoid large values of the inner product, especially when the dimension d is high. $Score(Q, K) \in \mathbb{R}^{l_Q\times l_K}$ is a matrix with the shape of the lengths of query (l_Q) and key (l_K), where the i-th row evaluates the relevant percentages of item Q_i with all items in key K. Then the output of the dot-product attention can be computed as a sum of the rows in value V, weighted by the attention scores, which is formulated as:

$$\text{Attention}(Q, K, V) = \text{score}(Q, K)V. \tag{15}$$

We propose a time-aware attention mechanism based on the scaled dot-product attention, which aims to take the time context into account when it calculates the correlation between two items. It is a general update of attention mechanism and can be applied in attention RNNs, self-attention networks and memory networks. Figure 2 shows its usage in the memory reader of MTAM.

We first define operation $\hat{-}$ to compute the time interval matrix between behaviour sequences. If $A \in \mathbb{R}^m$ and $B \in \mathbb{R}^n$ are two vectors, we define $\hat{-}$ as $C = A\hat{-}B$, such that $C_{ij} = A_i - B_j$ and $C \in \mathbb{R}^{m\times n}$. For two interaction sequences $(x, t_x) = \{(x_i, t_{xi})\}_i^{l_x}$ and $(y, t_y) = \{(y_i, t_{yi})\}_i^{l_y}$, the temporal gate in time-aware attention can be computed by:

$$\delta = \phi(\log(|t_x \hat{-} t_y| + 1) \odot W_\delta + b_\delta), \tag{16}$$

$$\tau = \phi(yW_\tau x + b_\tau), \tag{17}$$

$$g_{xy} = \sigma(\delta \odot W_{g\delta} + \tau \odot W_{g\tau} + b_g), \tag{18}$$

where $\delta, \tau, g_{xy} \in \mathbb{R}^{l_x\times l_y}$, $x \in \mathbb{R}^{l_x\times d}$, $y \in \mathbb{R}^{l_y\times d}$, $t_x \in \mathbb{R}^{l_x}$, $t_y \in \mathbb{R}^{l_y}$, $W_\tau \in \mathbb{R}^{d\times d}$, $W_\delta, W_{g\delta}, W_{g\tau} \in \mathbb{R}^{l_x\times l_y}$ and $b_\delta, b_\tau, b_g \in \mathbb{R}^{l_x\times l_y}$. The temporal feature δ and semantic feature τ encode the temporal correlations and semantic correlations between each pair of items in the two sequences. The temporal gate $g_{xy} \in \mathbb{R}^{l_x\times l_y}$ learns the non-linear correlations between (x, t_x) and (y, t_y) by taking both temporal and semantic information into consideration. The attention score in Eq. (14) is now changed to:

$$\text{T-Score}((x, t_x), (y, t_y)) = \text{softmax}\left(\frac{xy^T \odot g_{xy}}{\sqrt{d}}\right). \tag{19}$$

Finally, we define the time-aware attention of two temporal sequences by updating Eq. (15) as:

$$\text{T-Attention}((x, t_x), (y, t_y), (y, t_y)) = \text{T-Score}((\mathbf{x}, t_x), (y, t_y))\, y, \tag{20}$$

where the output is a representation of sequence x which is temporally related to sequence y.

4.4 Discussions

In this section, we have proposed a methodology to update the traditional attention mechanism and recurrent unit for user modeling. Despite the detailed differences between temporal gate g_{xy} in dot-product based attention kernel and temporal gate g_s in T-GRU unit, g_{xy} and g_s are both non-linear functions of the time interval between two objects and their semantic contexts. However, intuitively, the preferences tend to be similar within a short period, while large intervals may decrease the influences of the past actions. We have tried to build the temporal gate as a time-decaying function, for example:

$$g_{ij} = \exp(-\alpha(h_i, h_j)|t_i - t_j|), \text{where } \alpha(h_i, h_j) \geq 0. \tag{21}$$

But different from non-linear functions g_{xy} and g_s, it shows no obvious improvement when building a time-decaying temporal gate. Therefore, unlike our original intuition that the correlation between two actions decays with time, the temporal distances have shown to be more complex than a monotonic decrease.

5 MULTI-HOP TIME-AWARE ATTENTION MEMORY NETWORK

In this section, we describe three components of MTAM in detail. They are short-term intent encoder, long-term memory encoder and reading operation.

5.1 Short-term Intent Encoder and Long-term Memory Encoder

In MTAM, we treat the short-term intent c_u^{short} and a target time t_{target} as the query $q = (c_u^{short}, t_{\text{target}})$, and store the long-term preference in the memory matrix m. The input of the short-term intent encoder and the long-term memory encoder is a behaviour sequence $\mathrm{S}_u' = ((b_{u,1}', t_{u,1}), (b_{u,2}', t_{u,2}), ..., (b_{u,i}', t_{u,i}))$, where the i-th behaviour $(b_{u,i}', t_{u,i})$ is a tuple of behaviour embedding $b_{u,i}' \in \mathbb{R}^{1\times d}$ and timestamp $t_{u,i} \in \mathbb{R}$.

5.1.1 Short-term Intent Encoder. In the short-term intent encoder, we build a recurrent network with T-GRU unit (introduced in Section 4.3) rather than the traditional GRU [20] and LSTM [22] units to encode the current intents of users. T-GRU filters out irrelevant historical information by controlling how much past information can be transferred to future states via the temporal gate (Eq. (12)). Therefore, the proposed T-GRU unit is more capable than traditional and existing time-aware recurrent units [37, 40] to capture the current intents of users. We use the final hidden state $h_{u,i}$ as the short-term intent representation of user u: $c_u^{\text{short}} = h_{u,i}$, where $h_{u,i} \in \mathbb{R}^{1\times d}$ is the short-term intent representation.

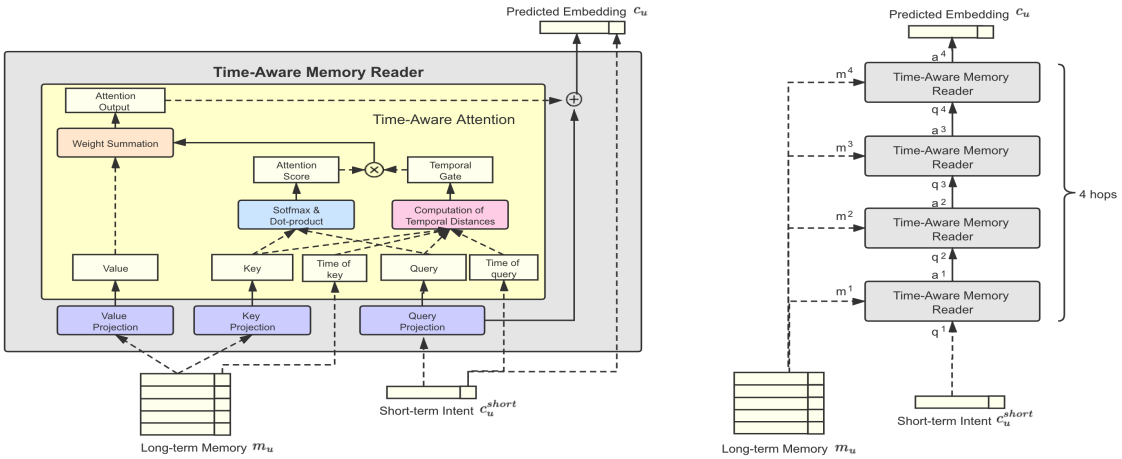


Figure 2: The illustration of multi-hop memory reader, where the left side shows a single layer version and right side shows a four-layer version. Time-aware Memory Reader takes the long-term memory m_u and the current short-term intent $(c_u^{short}, t_{target})$ of target user u as input and outputs the current predicted user embedding $(c_u, t_{\textbf{target}})$. The attention core of MTAM is time-aware attention, where the time-aware attention score combines both temporal and semantic corrections.

5.1.2 Long-term Memory Encoder. The long-term memory encoder, which can also be called a long-term memory writer, maintains user's prior records in a personalised memory m. Memory $m = (m_1, m_2, ..., m_L)$ is a fixed-length queue with L slots and each memory slot $m_i \in m$ stores a user historical record $(b'_{u,*}, t_{u,*})$. As mentioned in many researches [4, 23], users' recent behaviours usually are more important to the current predictions. We adopt a simple first-in-first-out rule and maintain the latest L behaviours of $\mathrm{S'}_u$ in the user memory m, which is:

$$m_u = ((b'_{u,i}, t_{u,i}), (b'_{u,i-1}, t_{u,i-1}), ..., (b'_{u,i-L+1}, t_{u,i-L+1})). \quad (22)$$

In the experiments, we empirically set L the same as the maximum length of $\mathrm{S'}_u$. If length of $\mathrm{S'}_u$ is less than L, we add zero-paddings to the right side of m_u to convert the m_u to a fixed-length queue.

5.2 Reading Operation

The reading operation is the key component of a memory network, which determines how to predict the answer based on a query and the information that is stored in the memory.

5.2.1 Single Layer. We start by describing the memory reader of MTAM in the single layer case, then show how to stack it for multiple hops.

The memory reader of MTAM reads the memory m_u attentively for a given query $(c_u^{\text{short}}, t_{\text{target}})$ and outputs a predicted user embedding c_u. We use the time-aware attention which is introduced in Section 4.3 as the attention kernel. As illustrated in Figure 2, the time-aware memory reader projects the current intent c_u^{short} and the behaviour embeddings in memory $m_{u,b}$ into value, key and query, respectively, and computes the attention output as:

$$o_u = \text{T-Attention}((c_u^{\text{short}} W_Q, t_{\text{target}}), (m_{u,b} W_K, t_{u,t}), (m_{u,b} W_V, t_{u,t})), \quad (23)$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d \times d}$ are query weight, key weight and value weight, $m_{u,b} = (b'_{u,i}, b'_{u,i-1}, ..., b'_{u,i-L+1})$ is the behaviour memory and $t_{u,t} = (t_{u,i}, t_{u,i-1}, ..., t_{u,i-L+1})$ is the time memory.

Then we compute the output of the memory reader, which is the predicted embedding of user u at time t_{target}, as: $c_u = c_u^{\text{short}} + o_u$.

5.2.2 Multiple Layers. Inspired by previous works [29, 30] where the multi-hop designs improve the performance of memory networks, we stack the single layer memory readers to construct a deeper network (MTAM). We illustrate how to build a multi-hop memory reader in Figure 2. Let the short-term intent be the query for the first hop. Then, the multi-hop memory reader can be formulated recurrently, where the output of the k-th hop is:

$$o_u^k = \text{T-Attention}((c_u^{k-1} W_Q^k, t_{\text{target}}), (m_{u,b} W_K, t_{u,t}), (m_{u,b} W_V, t_{u,t})) \quad (24)$$

$$c_u^k = c_u^{k-1} + o_u^k \quad (25)$$

Performing the reading operation for multiple hops helps us to capture the diversity of user preference, because the memory reader in different hops may concentrate on different behaviours. For a MTAM network with k' hops, the output, which is the predicted user embedding in Eq. (1), is $p_{u,t_{\text{target}}} = \text{MTAM}(S_u{}', t_{\text{target}}) = c_u^{k'}$.

6 EXPERIMENT

In this section, we first describe the setups of all experiments. Then, we demonstrate the effectiveness of our proposed models from the following aspects: (1) The performance of the proposed framework and comparable methods. (2) The effectiveness of the proposed time-aware attention kernel, T-GRU unit and the multi-hop structure of MTAM. (3) The Influence of Multiple Hops.

6.1 Datasets

We conduct experiments on six real-world datasets of two types. MovieLens-20 and Amazon datasets are rating datasets, which are not "real" user behaviour logs, but consist of comments on items. Yoochoose and Ali Mobile are transaction datasets, which directly record the behaviour trajectories of users.

Table 1: The Statistics of Datasets

Statistics	#user	#Item	#Cat.	Avg. behaviours per user	Density
ml-25m	12015	4991	712	104.27	2.0891%
Electronics	41940	87203	1063	31.67	0.0363%
CDs & Vinyl	39663	33593	419	24.78	0.0738%
Movies & TV	105321	35164	379	22.70	0.0645%
Yoochoose 1/4	108817	16296	187	15.52	0.0953%
Ali Mobile	9980	594083	6352	1226.92	0.1267%

- **MovieLens**[1] is a widely used benchmark dataset for evaluating collaborative filtering algorithms. We use the latest stable version (MovieLens-25m) which includes 25 million user ratings.
- **Amazon**[2] is a popular dataset to evaluate recommendation algorithms. It is always used as a benchmark for sequential recommendation tasks. We consider three categories: Electronics, CDs & Vinyl and Movies & TV.
- **Yoochoose**[3] from the RecSys'15 Challenge I contains clickstreams gathered from an e-commerce web site during six months. Because the Yoochoose dataset is quite large, we randomly sample 1/4 users.
- **Ali Mobile**[4] from the Alibaba Competition contains transaction data gathered from Alibaba's M-Commerce platform in one month.

We filter the users whose behaviour lengths are less than 10 and items that appear less than 30 times. The statistics of six datasets after data preprocessing are shown in Table 1. Although there are various types of context information in these datasets (e.g. actions, comments, descriptions), we only use the category of an item and the position of a behaviour in a sequence as context features in our experiments. For a behaviour sequence $S_u = (b_{u,1}, b_{u,2}, ..., b_{u,i})$, we use a sequence splitting preprocess to generate the sequences and corresponding labels $([b_{u,1}], b_{u,2})$, $([b_{u,1}, b_{u,2}], b_{u,3})$, ..., $([b_{u,1}, b_{u,2}, ..., b_{u,i-2}], b_{u,i-1})$ for the training set and the last behaviour $([b_{u,1}, b_{u,2}, ..., b_{u,i-1}], b_{u,i})$ for the testing set.

6.2 Compared Methods and Implementation Details

We compare **MTAM** with the following competitive models:

- **Top Pop/P-Pop** recommends items of the largest interactions with all users/a target user. They are commonly used baselines for all recommendation researches.
- **BPR-MF** [26] is a matrix factorisation recommender for the personalised ranking task.
- **GRU** is a classical recurrent sequential recommender.
- **GRU−−** is **GRU** without the category features.
- **T-SeqRec** equips LSTM with two temporal gates to model time intervals and time spans for recommendation problem [37]. Since it dominates other time-aware RNN-based models (e.g. [5, 40]), we treat it as a state-of-the-art time-aware recurrent unit.
- **T-GRU** is proposed in this paper, which equips GRU with a new temporal gate. Different from **T-SeqRec**, **T-GRU** only focuses on short-term preference without using time spans to capture long-term preference.
- **SASRec** [17] is a self-attention based sequential recommender.
- **TiSASRec** [21] is a time-aware self-attention sequential recommender which directly adds the clipped relative time intervals to the dot-product of item embeddings.
- **NARM** [20] is a hybrid sequential recommender which utilises attention mechanism to model the user's local purposes. It is a commonly used baseline hybrid recommender which takes both local and global preferences into consideration.
- **NARM+** is improved by equipping with our proposed time-aware attention.
- **NARM++** is improved by equipping with our proposed time-aware attention and T-GRU.
- **STAMP** [23] captures both long-term and short-term preferences using an attention MLP network.

To ensure fair comparison, we set all hidden units and low-rank embedding spaces, including RNN layers and attention layers, as 128. We set the initial learning rate as 1e-3 and use an exponential learning rate decay for every 100 iterators with 0.995 decay rate. 0.5 dropout rate and 1e-5 regularisation rate are used to reduce overfitting. The maximum length of user behaviour is set to 50. The proposed models and all compared models are implemented with TensorFlow 1.14[5], and trained and tested on a Linux server with a Tesla P100 GPU.

6.3 Evaluation Metrics

Since items that an individual can interact with are extremely sparse, recommenders can suggest a set of candidate items each time. We use HR@k and NDCG@k as the metrics for all models, where k is the number of items recommended each time.

HR@k is short for Hit Ratio, which shows whether the target item is in the recommended list or not. Since we only consider one ground truth for each sample, HR@k is equivalent to Recall@k.

NDCG@k takes the position of the hit item into account by assigning a highest score to the hit at top rank and decreasing the scores to hits at lower ranks. NDCG, short for Normalised Discounted Cumulative Gain, not only considers the HR but also the orders of ranking.

Statistical significance of observed differences between the performance of the purposed MTAM and the best baseline methods is tested by the t-test on pair-wise samples. Small p-values are associated with large t-statistics, where the threshold 0.01 means strong significance and the threshold 0.05 means weak significance.

6.4 Overall Recommendation Performance

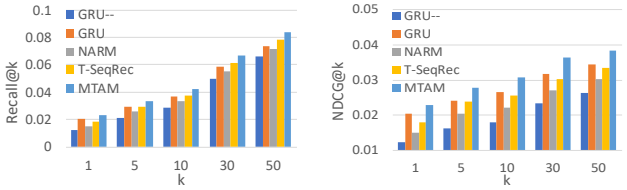
Table 2 and Figure 3, 4 illustrate the performance of **MTAM** and the baseline methods. We only report the results of $K = 10$ for all models and all datasets in Table 2 due to space limitation. And in Figure 3, 4, we take the one rating dataset (Electronics) and one transaction dataset (Ali Mobile) as representatives to show the results of different Ks for typical models.

In general, the proposed MTAM outperforms the state-of-the-art methods significantly (p-value<0.01). Our **MTAM** has achieved

[1] https://grouplens.org/datasets/movielens/20m/
[2] http://deepyeti.ucsd.edu/jianmo/amazon/index.html
[3] http://2015.recsyschallenge.com/challenge.html
[4] https://tianchi.aliyun.com/dataset/dataDetail?dataId=46
[5] https://github.com/cocoandpudding/MTAMRecommender

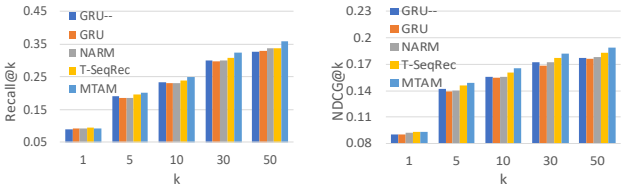
Table 2: Performance comparison of MTAM and the baseline methods. Among these baselines, T-GRU is a component of MTAM, and NARM+ and NARM+ is implemented with our proposed T-GRU and time-aware attention. We divide models into 4 groups: naive recommenders (e.g. Top Pop), non-hybrid recommenders (e.g. GRU), hybrid recommenders (e.g. NARM) and the proposed MTAM. The underlined number is the best baseline method and the boldfaced number is the best method of all. Improv. denotes the improvement of the best model over the best baseline method. Significant differences are with respect to the best baseline methods.

	Top Pop	P-Pop	BPR-MF	GRU−−	GRU	T-SeqRec	**T-GRU**	SASRec	TiSASRec	NARM	**NARM+**	**NARM++**	STAMP	**MTAM**	improv.	p-value
	HR@10															
ML-25m	0.0300	0	0.0108	0.1985	0.1946	*0.2001*	0.2006	0.1847	0.1865	0.1999	0.1978	0.1981	0.1821	**0.2053**	2.60%	3.39e-3
Electronics	0.0108	0.0017	0.0204	0.0283	0.0370	*0.0376*	0.0384	0.0328	0.0332	0.0334	0.0322	0.0371	0.0330	**0.0423**	12.5%	6.12e-11
CDs & Vinyl	0.0062	0.0028	0.0330	0.1070	0.1082	0.1130	0.1129	0.1053	0.1075	*0.1131*	0.1111	0.1160	0.1091	**0.1206**	6.63%	6.65e-3
Movies & TV	0.0114	0.0021	0.0279	0.1512	0.1505	*0.1550*	0.1523	0.1422	0.1441	0.1499	0.1548	0.1583	0.1372	**0.1605**	3.54%	6.96e-5
Yoochoose	0.0212	0.2599	0.3263	0.5273	0.5345	0.5332	0.5351	0.5240	0.5270	0.5356	0.5355	0.5360	*0.5360*	**0.5386**	0.49%	5.40e-4
Ali Mobile	0.0046	0.1366	0.1533	0.2321	0.2299	*0.2380*	0.2427	0.2274	0.2292	0.2318	0.2329	0.2427	0.2097	**0.2501**	5.08%	7.99e-4
	NDCG@10															
ML-25m	0.0140	0	0.0050	0.1146	0.1112	*0.1159*	0.1150	0.1001	0.1002	0.1129	0.1127	0.1121	0.0981	**0.1187**	2.42%	6.52e-3
Electronics	0.0051	0.0009	0.0119	0.0180	*0.0267*	0.0257	0.0263	0.0229	0.0231	0.0221	0.0205	0.0249	0.0232	**0.0307**	15%	3.81e-9
CDs & Vinyl	0.0030	0.0016	0.0138	0.0715	0.0705	*0.0756*	0.0754	0.0667	0.0679	0.0747	0.0724	0.0746	0.0723	**0.0783**	3.57%	6.51e-3
Movies & TV	0.0056	0.0013	0.0135	0.1135	0.1129	*0.1167*	0.1138	0.1045	0.1061	0.1116	0.1168	0.1169	0.1035	**0.1210**	3.68%	4.37e-5
Yoochoose	0.0102	0.1698	0.1977	0.3313	0.3365	0.3360	0.3363	0.3293	0.3295	0.3372	0.3384	**0.3386**	*0.3380*	0.3381	0.12%	6.35e-4
Ali Mobile	0.0023	0.0834	0.1011	0.1562	0.1549	*0.1602*	0.1615	0.1524	0.1541	0.1555	0.1560	0.1621	0.1424	**0.1651**	3.06%	3.83e-3



(a) Recall on Amazon Electronics (b) NDCG on Amazon Electronics

Figure 3: The overall performance comparison on Amazon Electronics.



(a) HR on Ali Mobile (b) NDCG on Ali Mobile

Figure 4: The overall performance comparison on Ali Mobile.

the best performances on five datasets except Yoochoose. On Yoochoose, **MTAM** performs the best for HR@10, but is defeated by **NARM++** for NDCG@10. **NARM++** is an attention RNN model which is updated by the proposed time-aware attention and T-GRU. Therefore, the overall experimental results demonstrate the effectiveness of our proposed methods. Comparing the results, we have four observations:

(1) Compared with the models based on traditional RNN and attention mechanism, time-aware neural networks perform obviously better on all datasets. For example, **T-SeqRec** dominates all baseline models in four of the six datasets, where it even achieves much better performance than **NARM**. And **NARM+** outperform **NARM** on three datasets, while **NARM++** outperform both **NARM+** and **NARM** on most all dataset (**NARM++** has comparable performance to **NARM+** on the ML-25m dataset). These results confirm that temporal information clearly contributes to recommendation performance and further demonstrate that the proposed time-aware attention and T-GRU evidently help to build significantly better user models.

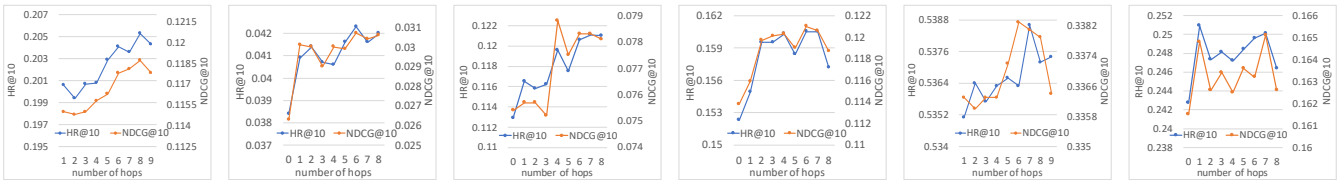
(2) When talking about whether hybrid recommenders that consider both long-term and short-term preferences provide more competitive results, it is a little complicated. First, we observe that **MTAM** dominates all pure RNN-based or self-attention based models, but **NARM** loses to **GRU** on two datasets. Then, we can see that **NARM++** performs better than all pure models. These results indicate that leveraging both long-term and short-term preferences leads to substantial performance improvement in sequential recommendation, but the hybrid models should have the ability to model the temporal dynamics, diversity and complexity of users' sequential behaviours. The outperformance of **MTAM** demonstrates that the proposed multi-hop memory network is good at dealing with sequential information with the help of time-aware attention and T-GRU.

(3) The results of **Top Pop** and **P-Pop** are quite different on rating data and transaction data. We observe that the simple **Top Pop** provides adequate baseline results on the ML-25m and all Amazon datasets, but it hardly works on Yoochoose and Ali Mobile datasets. On the other hand, **P-Pop** performs well on Yoochoose and Ali Mobile datasets, but doesn't work on the rating datasets. These results directly reveal the different patterns in rating sequences and transaction sequences. Users may not rate the same item multiple times, which explains the poor performance of **P-Pop** on rating datasets. While in transaction scenarios, users tend to click on the same item many times and the more interactions may indicate more interest, which explains the good performance of **P-Pop** on transaction datasets. The experimental results demonstrate that **MTAM** performs well on both rating data and transaction data.

(4) We observe that **MTAM** provides more competitive results than **NARM++** on five of six datasets, while they are comparable on remaining one. **MTAM** and **NARM++** are both implemented with the proposed time-aware attention and T-GRU. The differences

Table 3: Ablation Analysis on six datasets. MTAM with T-SeqRec uses T-SeqRec as the recurrent unit. MTAM without Time-Aware RNN uses traditional GRU as recurrent unit and MTAM without Time-Aware attention uses traditional dot-product attention as attention kernel. MTAM via T-GRU and MTAM via GRU store the hidden states of RNN instead of behaviour embeddings. Time-Aware Self Attention is a SASRec implemented with the proposed time-aware attention. We treat GRU as the baseline model and the boldface number is the best method. We show the improvement of the best method over GRU.

	HR@10						NDCG@10					
	ML-25m	Electronics	CDs & Vinyl	Movies & TV	Yoochoose	Ali Mobile	ML-25m	Electronics	CDs & Vinyl	Movies & TV	Yoochoose	Ali Mobile
GRU	*0.1946*	*0.0370*	*0.1082*	*0.1474*	*0.5345*	*0.2299*	*0.1159*	*0.0267*	*0.0705*	*0.1104*	*0.3365*	*0.1549*
MTAM	0.2053	**0.0423**	**0.1206**	0.1605	0.5386	0.2501	0.1187	**0.0307**	**0.0783**	0.1210	0.3381	0.1651
MTAM with T-SeqRec	**0.2073(1.0% ↑)**	0.0396(6.4% ↓)	0.12(0.5% ↓)	0.1603(0.1% ↓)	0.5378(0.2% ↓)	0.2531(1.2% ↑)	**0.1203(1.3% ↑)**	0.0289(5.9% ↓)	0.0789(0.8% ↑)	0.1213(0.3% ↑)	0.337(0.3% ↓)	0.1671(1.2% ↑)
MTAM without Time-Aware RNN	0.2038(0.7% ↓)	0.025(41% ↓)	0.1135(5.9% ↓)	0.1603(0.1% ↓)	0.5389(0.1% ↑)	0.2519(0.7% ↑)	0.1171(1.3% ↓)	0.0156(49% ↓)	0.075(4.2% ↓)	0.1212(0.1% ↑)	0.3378(0.1% ↓)	0.1669(1.0% ↑)
MTAM without Time-Aware Attention	0.1952(4.9%↓)	0.0375(11%↓)	0.1127(6.6%↓)	**0.1612**(0.4%↑)	0.5361(0.4%↓)	0.2420(3.2%↑)	0.1116(6.0%↓)	0.0264(14%↓)	0.0727(7.2%↓)	**0.1218**(0.7%↑)	0.3367(0.4%↓)	0.1603(2.9%↓)
MTAM via T-GRU	0.2032(1.0% ↓)	0.0369(13% ↓)	0.1136(5.8% ↓)	0.1611(0.4% ↑)	0.5348(0.7% ↓)	0.25161(0.6% ↑)	0.1161(2.2% ↓)	0.0246(20% ↓)	0.0738(5.8% ↓)	0.1205(0.4% ↓)	0.3376(0.2% ↓)	0.1673(1.3% ↑)
MTAM via GRU	0.2002(2.5% ↓)	0.0255(40% ↓)	0.1095(9.2% ↓)	0.1593(0.8% ↓)	0.5352(0.6% ↓)	0.2501(-)	0.1141(3.8% ↓)	0.0154(50% ↓)	0.0717(8.4% ↓)	0.1207(0.3% ↓)	0.3366(0.4% ↓)	0.1652(0.1% ↑)
Time-Aware Self Attention	0.1953(4.7% ↓)	0.038(10% ↓)	0.1112(7.8% ↓)	0.1566(2.4% ↓)	**0.5393**(0.1% ↑)	**0.2549**(1.9% ↑)	0.1112(6.3% ↓)	0.0283(7.8% ↓)	0.0736(6.0% ↓)	0.1175(2.9% ↓)	**0.3394**(0.4% ↑)	**0.1682**(1.9% ↑)
improvement	6.5%	14.3%	11.5%	9.4%	0.90%	8.8%	3.8%	15.0%	11.1%	10.3%	0.86%	8.6%



(a) HR@10 and NDCG@10 on ML-25m (b) HR@10 and NDCG@10 on Amazon Electronics (c) HR@10 and NDCG@10 on Amazon CDs & Vinyl (d) HR@10 and NDCG@10 on Amazon Movies & TV (e) HR@10 and NDCG@10 on Yoochoose (f) HR@10 and NDCG@10 on Ali Mobile

Figure 5: The performance comparison among MTAMs with different number of hops on six datasets, where MTAM with 0 hop is T-GRU.

between them are that **MTAM** integrates long-term and short-term preferences via a multi-hop memory network, while **NARM++** uses an attention mechanism on the hidden states of RNN to learn local user preferences, and combines local and global preferences with a bi-linear decoder. The experimental results prove that a multi-hop memory network provides a better way for information fusion.

6.5 Ablation Analysis for MTAM

To verify the effectiveness of the proposed time-aware attention, T-GRU and the multi-hop structure of **MTAM**, we conduct an ablation analysis to demonstrate the contribution of each module. Similar to the experiments of overall performance in Section 6.4, we only report the results of $K = 10$ for all datasets in Table 3.

From the results in Table 3, we have some observations:

(1) **MTAM** defeats **MTAM with T-SeqRec** for HR@10 on four of the six datasets, while only defeats it for NDCG@10 on four of the six datasets. Similar results can also be observed in the comparison between **T-GRU** and **T-SeqRec** in Table 2. These results indicate that the proposed **T-GRU** performs better at the recall task than the ranking task. Since the mission of **MTAM** is Top-K recommendation in the candidate retrieval stage, T-GRU seems to be the better choice to capture user's short-term intent.

(2) We observe that **MTAM** performs better than MTAMs based on a traditional attention mechanism and GRU unit on five of the six datasets, except that **MTAM without Time-Aware Attention** dominates other models on the Movies & TV dataset. These results prove that in most cases the proposed temporal gating methodology improves the performance of attention and recurrent neural models for the Top-K recommendation task.

(3) **MTAM** is obviously more competitive than **MTAM via T-GRU** and **MTAM via GRU**. **MTAM** maintains behaviour embeddings in memory, while **MTAM via T-GRU** and **MTAM via GRU** store the hidden states of RNN in memory. This result confirms that a memory network is more effective to learn long-term dependencies than attention recurrent networks (e.g. **NARM**). An explanation is that RNNs would forcefully summarise the information of all prior behaviours into a hidden state, which makes it difficult to assign credit to each behaviour in prediction.

(4) To demonstrate that the proposed time-aware attention is a generally improved version of the attention mechanism and can be applied to attention RNNs, self-attention networks and memory networks, we not only equip **NARM** and **MTAM** with it, but also update **SASRec** to a time-aware version. To our surprise, **Time-Aware Self Attention** achieves the best performance amongst all models on two transaction datasets, but loses to **MTAM** and to most other reassembled MTAMs on all rating datasets. This interesting observation first illustrates that the temporal distance is of great importance to the attention mechanism when calculating the correlation between two hidden vectors. Secondly, it implies that time-aware self-attention models may be more competent to handle transaction sequences than attention recurrent models and memory models. Still, attention recurrent models and memory models are probably better choices to deal with rating sequences.

6.6 Influence of Multiple Hops

Finally, we are curious about whether reading user memory for multiple hops helps to improve the performance of **MTAM**. We study the performance of **MTAM** for HR@10 and NDCG@10 by tuning the number of hops in the range of $0 \sim 8$.

Results are shown in Figure 5. We observe that the **MTAM**s with multiple hops preform better than the **MTAM** with one hop on five out of six datasets. The only exception is a transaction dataset, Ali Mobile, in which the average length of user behaviour sequences is much longer than the other five datasets (shown in Table 1). On Ali Mobile dataset, the single-hop **MTAM** obviously improves upon **T-GRU**, but fails to be further strengthened by performing the reading operation for multiple hops. This result indicates that a deeper memory network may be not suitable for all datasets. Nevertheless, overall, multiple hops improve the performance of **MTAM**. Whilst the best number of hops varies from one dataset to another, in most cases, **MTAM** needs to read the user memory for more than four hops in our experiments.

7 CONCLUSIONS AND FUTURE WORKS

In this paper, we proposed a novel Multi-hop Time-aware Attention Memory (MTAM) network for the task of sequential recommendation. We first updated the attention mechanism and recurrent unit with a new temporal gate to capture the temporal context of user behaviours. Then we encoded user's short-term intent with the proposed T-GRU and maintained user's long-term records in memory. Finally, the user modeling procedure can be viewed as a decision making progress by reading user memory for multiple hops based on the short-term intent. The experimental results clearly demonstrate the effectiveness of MTAM for Top-K recommendation and the general improvement of the proposed temporal gating methodology to update the traditional attention mechanism and recurrent unit for user modeling. Our experiments show a great potential in integrating the temporal gate and self-attention neural networks. Compared to NLP tasks, temporal information is much more important for dependencies between user behaviours than for words. We plan to further explore how to take full advantage of temporal information to improve self-attention neural recommenders.

8 ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China (No. 2017YFC0803700), NSFC grants (No. 61532021 and 61972155), and the Shanghai Knowledge Service Platform Project (No. ZF1213).

REFERENCES

[1] Mingxiao An, Fangzhao Wu, Chuhan Wu, and et al. 2019. Neural News Recommendation with Long-and Short-term User Representations. In *Proceedings of the 57th ACL*. 336–345.
[2] Linas Baltrunas, Bernd Ludwig, and Francesco Ricci. 2011. Matrix factorization techniques for context aware recommendation. In *Proceedings of the 15th RecSys*. 301–304.
[3] Alex Beutel, Paul Covington, Sagar Jain, and et al. 2018. Latent cross: Making use of context in recurrent recommender systems. In *Proceedings of the 11th WSDM*. 46–54.
[4] Xu Chen, Hongteng Xu, Yongfeng Zhang, and et al. 2018. Sequential recommendation with user memory networks. In *Proceedings of the 11th WSDM*. 108–116.
[5] Xu Chen, Yongfeng Zhang, and Zheng Qin. 2019. Dynamic Explainable Recommendation based on Neural Attentive Models. In *Proceedings of the 33rd AAAI*, Vol. 33. 53–60.
[6] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th RecSys*. 191–198.
[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and et al. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[8] Robin Devooght and Hugues Bersini. 2017. Long and short-term recommendations with recurrent neural networks. In *Proceedings of the 25th UMAP*. 13–21.
[9] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative memory network for recommendation systems. In *Proceedings of the 41st SIGIR*. 515–524.
[10] Yufei Feng, Fuyu Lv, Weichen Shen, and et al. 2019. Deep Session Interest Network for Click-Through Rate Prediction. In *Proceedings of the 26th IJCAI*. 2301–2307.
[11] Jonas Gehring, Michael Auli, David Grangier, and et al. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th ICML*. 1243–1252.
[12] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* (2014).
[13] Alex Graves, Greg Wayne, Malcolm Reynolds, and et al. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature* 538, 7626 (2016), 471–476.
[14] Xiangnan He, Lizi Liao, Hanwang Zhang, and et al. 2017. Neural collaborative filtering. In *Proceedings of the 26th WWW*. 173–182.
[15] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and et al. 2015. Session-based Recommendations with Recurrent Neural Networks. *Computing Research Repository* abs/1511.06939 (2015).
[16] Liang Hu, Longbing Cao, Shoujin Wang, and et al. 2017. Diversifying Personalized Recommendation with User-session Context.. In *Proceedings of the 26th AAAI*. 1858–1864.
[17] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *Proceedings of the 18th ICDM*. 197–206.
[18] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[19] Ankit Kumar, Ozan Irsoy, Peter Ondruska, and et al. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33rd ICML*. 1378–1387.
[20] Jing Li, Pengjie Ren, Zhumin Chen, and et al. 2017. Neural attentive session-based recommendation. In *Proceedings of the 26th CIKM*. 1419–1428.
[21] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *Proceedings of the 13th WSDM*. 322–330.
[22] Zhi Li, Hongke Zhao, Qi Liu, and et al. 2018. Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors. In *Proceedings of the 24th SIGKDD*. 1734–1743.
[23] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and et al. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD*. 1831–1839.
[24] Sameen Maruf and Gholamreza Haffari. 2018. Document Context Neural Machine Translation with Memory Networks. In *Proceedings of the 56th ACL*. 1275–1284.
[25] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and et al. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).
[26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and et al. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th UAI*. 452–461.
[27] Steffen Rendle and Lars Schmidt-Thieme. 2008. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *Proceedings of the 2nd RecSys*. 251–258.
[28] Xiaoyuan Su and Taghi M Khoshgoftaar. [n.d.]. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 ([n. d.]).
[29] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, and et al. 2015. End-to-end memory networks. In *Proceedings of the 28st NIPS*. 2440–2448.
[30] Thanh Tran, Xinyue Liu, Kyumin Lee, and et al. 2019. Signed Distance-based Deep Memory Recommender. In *Proceedings of the 28th WWW*. 1841–1852.
[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, and et al. 2017. Attention is all you need. In *Proceedings of the 31st NIPS*. 5998–6008.
[32] Meirui Wang, Pengjie Ren, Lei Mei, and et al. 2019. A collaborative session-based recommendation approach with parallel memory modules. In *Proceedings of the 42nd SIGIR*. 345–354.
[33] Chien-sheng Wu, Richard Socher, and Caiming Xiong. 2019. Global-to-local Memory Pointer Networks for Task-oriented Dialogue. In *Proceedings of the 7th ICLR*.
[34] Qitian Wu, Yirui Gao, Xiaofeng Gao, and et al. 2019. Dual Sequential Prediction Models Linking Sequential Recommendation and Information Dissemination. In *Proceedings of the 25th SIGKDD*. 447–457.
[35] Xinyang Yi, Ji Yang, Lichan Hong, and et al. 2019. Sampling-bias-corrected neural modeling for large corpus item recommendations. In *Proceedings of the 13th RecSys*. 269–277.
[36] Lu Yu, Chuxu Zhang, Shangsong Liang, and et al. 2019. Multi-order Attentive Ranking Model for Sequential Recommendation. In *Proceedings of the 33rd AAAI*.
[37] Zeping Yu, Jianxun Lian, Ahmad Mahmoody, and et al. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation. In *Proceedings of the 28th IJCAI)*.
[38] Jiani Zhang, Xingjian Shi, Irwin King, and et al. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th WWW*. 765–774.
[39] Guorui Zhou, Na Mou, Ying Fan, and et al. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of 33rd the AAAI*. 5941–5948.
[40] Yu Zhu, Hao Li, Yikang Liao, and et al. 2017. What to do next: modeling user behaviors by time-LSTM. In *Proceedings of the 26th IJCAI*. 3602–3608.