Received August 29, 2020, accepted September 9, 2020, date of publication September 14, 2020, date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023685

A Time-Aware Graph Neural Network for Session-Based Recommendation

YUPU GUO, YANXIANG LING, AND HONGHUI CHEN
Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China

Corresponding author: Yanxiang Ling (guoyupu14@nudt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702526, in part by the National Advanced Research Project under Grant 6141B0801010b, and in part by the Defense Industrial Technology Development Program under Grant JCKY2017204B064.

ABSTRACT Recently, Graph Neural Networks (GNNs) have attracted increasing attention in the field of session-based recommendation, due to its strong ability on capturing complex interactive transitions within sessions. However, existing GNN-based models either lack the use of user's long-term historical behaviors or fail to address the impact of collaborative filtering information from neighbor users on the current session, which are both important to boost recommendation. In addition, previous work only focuses on the sequential relations of interactions while neglects the time interval information which can imply the correlations between different interactions. To tackle these problems, we propose a **T**ime-**A**ware **G**raph **N**eural **N**etwork (**TA-GNN**) for session-based recommendation. Specifically, we first construct a user behavior graph by linking the interacted items of the same user according to their corresponding time order. A time-aware generator is designed to model the correlations between different nodes of the user behavior graph by considering the time interval information. Moreover, items from the neighbor sessions of the current session are selected to build a neighborhood graph. Then the two graphs are respectively processed by two different modules to learn the representation of the current session, which is applied to produce the final recommendation list. Comprehensive experiments show that our model outperforms state-of-the-art baselines on three real world datastes. We also investigate the performance of TA-GNN on different numbers of historical interactions and on different session length, finding that our model presents consistently advantages under different conditions.

INDEX TERMS Session-based recommendation, sequence recommendation, graph neural network.

I. INTRODUCTION

In the information explosion era, recommender systems (RS) have become increasingly important to help user pick out the information he (or she) actually needs in many domains, e.g., web search, e-commerce, and music. Conventional RS makes recommendation mainly based on a static user rating matrix, which neglects the chronological order of user's interactions. As a result, conventional RS is deficient in catching user's short-term intent from his (or her) recent interactions. To tackle this problem, session-based recommender system (SBRS) is proposed to learn user's recent preference of the ongoing session. In SBRS, a session denotes an ordered sequence of items which are interacted by the same user within a period.

The associate editor coordinating the review of this manuscript and approving it for publication was Eyhab Al-Masri.

Current studies mostly employ the Recurrent Neural Network (RNN) to handle the session sequence in SBRS [1]–[4], while Wu *et al.* [5] argue that RNN-based methods can only model single-way transitions between consecutive items and neglect contextual transitions among the entire session sequence. To handle the complex transitions among distant items, Wu *et al.* [5] first propose a SR-GNN model to introduce Graph Neural Networks for SBRS, which achieves strong performance. Furthermore, Xu *et al.* [6] combine GNN and Self-Attention Network (SAN) to capture long-range dependencies in sessions. Yu *et al.* [7] propose a novel target attentive graph neural network to take candidate items into consideration when generating the session representation. Considering that the above work only relies on anonymous sessions lacking users' long-term profiles, Wu *et al.* [8] propose to build user behavior graph based on both user's long-term and short-term interactions.

Despite that existing GNN-based methods have made remarkable progress on SBRS, they mostly neglect collaborative information from neighbor sessions that has been proved critical to help provide satisfying recommendation for current session. Moreover, previous works only consider the chronological order of interactions while the time interval between two interactions has not been well-studied as it may help to understand the correlations of different interacted items. Take Fig. 1 as an example, if a user interact with a item after another with a short time interval, it may indicate that the two items are highly correlated by user's ongoing interest. While a long time interval between two items may indicate a low correlation since user's interest has drifted.

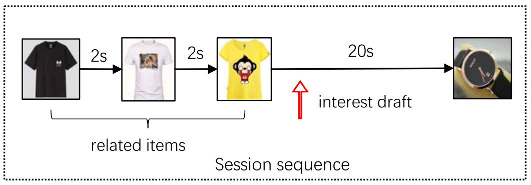


FIGURE 1. An example of how time internal reflect the correlation of items.

To address the above issues, we propose a **T**ime-**A**ware **G**raph **N**eural **N**etwork (**TA-GNN**) for session-based recommendation task. In detail, we first construct a user behavior graph by linking the interacting items of the same user based on their time order in sessions. Then we apply the idea of K-nearest neighbor [9] to find neighbor sessions for the current session and use them to build the neighborhood graph. Based on the user behavior graph, we propose a profile-based representation learning module consisting of a time-aware generator and a general generator, which build adjacency matrices according to the time interval and the number of edges between two nodes, respectively. In detail, the time-aware generator can assign a larger value to the corresponding position of the adjacency matrices when the time interval between two adjacent items is smaller, which allows the two items with shorter time internal between them to get more information from each other when generating their representations. The representation of each node in the user behavior graph is learned by a graph neural network, and an attention mechanism is applied to generate the representation of the current session by a linear combination of node representations in the current session. As to the neighborhood graph, we use a general graph neural network to obtain another representation of the current session. After that, we combine the two representations, i.e., the profile-based and neighbor-based representations to characterize the current session and produce the recommendation. We conduct extensive experiments on three real-world datasets and the results show that our TA-GNN model outperforms the state-of-art baselines in terms of Recall@20 and MRR@20.

Our main contribution can be summarized as:

- We propose a TA-GNN model for session-based recommendation, which can effectively leverage the effect of collaborative information and long-term behaviors on the short-term session to make personalized and rich recommendations.
- To the best of our knowledge, we are the first to introduce the time interval information to session-based recommendation. We construct a time-aware generator that can infer the correlation of two items from the detail of the time interval between interactions.
- We conduct extensive experiments on three publicly available datasets. Our experimental results show that TA-GNN can improve the performance on session-based recommendation task.

The remainder of this paper is organized as follows. First, we introduce the related works in section II. Then, we formulate the session-based task and detail our TA-GNN in section III. Next, in section IV and V, we present the experimental settings and results, respectively. Finally, we conclude our work and list the suggestions for future work.

II. RELATED WORK

In this section, we briefly introduce the previous work on session-based recommendation (§II-A) at first. Then we introduce the development of Graph Neural Network (§II-B) in RS.

A. SESSION-BASED RECOMMENDATION

Session-based Recommender System (SBRS) is a sub-task of sequential recommendation [10]. Early session-based approaches are based on Markov Chain (MC) models [11]–[13] to capture sequential patterns between user-item interactions. Then the Markov Decision Processes (MDP) models [14]–[16] are proposed to overcome the weakness of MC on sparse data. However, Markov-based models neglect considering sequentiality between discontinuous items, which leads to imbalance between user's general performance and sequential behavior [17].

Recently, RNN becomes a primary structure in SBRS due to its strong sequence processing ability [18]. Hidasi *et al.* [1] first employ RNN with Gated Recurrent Unit (GRU) in SBRS. Their model is widely known as GRU4rec which inspires much following work [19], [20]. For instance, Tan *et al.* [4] further improve GRU4rec by adopting data augmentation and a method to account for shifts in the input data distribution. Later, Li *et al.* [3] propose a Neural Attentive Recommendation Machine (NARM), which uses attention mechanism to balance local and global preference of users. To combine long-term and short-term behaviors, Quadrana *et al.* [20] propose a hierarchical RNN structure that involves historical information in generating the representation of the ongoing session. Furthermore, Li *et al.* [21] propose a Bi-LSTM structure on users' entire historical interaction profiles to make recommendation for the current session. To distinguish the influence of different historical sessions, Sun *et al.* [22] apply a two-layer nonlocal architecture to identify relevant historical sessions and then learn accurate short-term preference. With the same purpose, Chen *et al.* [23] apply a co-attention network to capture the dynamic interactions between the user's long-term and short-term interaction behaviors.

Besides RNN structures, memory networks also show superiority on sequential recommendation in recent years. For example, Liu *et al.* [24] propose a Short-Term Attention/Memory Priority Model (STAMP), which beats the RNN-based method NARM in SBRS. Moreover, a growing number of work incorporate the idea of collaborative filtering into sequential recommendations. For example, Bonnin and Jannach [25] propose SKNN to consider each session as a whole in sequential recommendation and its improved version KNN-RNN [26] combines KNN and RNN to achieve better performance. Later, Wang *et al.* [27] build an end-to-end model, which contains two modules to model user's own information in the current session and exploit collaborative information to better predict the intent in the current session, respectively.

Compared with the approaches outlined above, our TA-GNN model utilizes both the user's long-term behaviors and collaborative information from neighbor sessions into consideration, and employs the graph neural network to leverage the above information in a unified framework. Our approach differs from previous work in that we first introduce the time interval information to session-based recommendation task.

B. GRAPH NEURAL NETWORKS

Neural networks are widely used for generating representation for graph-structured data, r.g., knowledge base and social network. However, traditional graph representation methods, such as DeepWalk [28], LINE [29] and Deep Graph Kernels [30], mainly focus on unsupervised tasks and have difficulty in scaling to large graphs. In order to learn graph embedding in the supervised scenario, Yanardag and Vishwanathan [30] propose the concept of Graph Neural Network, which is extended by Micheli [31] and Scarselli *et al.* [32]. These methods mainly learn the representations of target nodes based on the recurrent neural unit. Following their work, Li *et al.* [33] propose a Gated Graph Neural Network (GGNN) to the sequential output, which makes a great improvement on SBRS.

Inspired by the success of attention mechanism in various tasks, numerous researchers apply attention mechanisms on graphs [34]. For example, Velickovic *et al.* [35] applies attention mechanism to learn node representation in a graph, which leads to the Graph Attention Network (GAT) model. With the help of attention mechanism, GAT can aggregate more information from the most critical part of the graph. Extending the superiority of GAT to SBRS, Zheng *et al.* [36] apply a multi-head attention to decide the importance of neighbor items of the current session. Wu *et al.* [8] use the self-attention mechanism to explore the effect of the historical interactions on the items of the current session.

Different from the above GNN-based models, we apply a time-aware generator when modeling the graph structure, which determines the adjacency matrix according to the time interval between two interactions.

III. METHOD

In this section, we first formalize the task of session-based recommendation and present an overall framework of our solution. Then, the proposed Time-Aware Graph Neural Network (TA-GNN) is introduced in detail.

A. OVERVIEW

Let $I = v_{i=0}^{|I|}$ be the items set in the dataset, each item v_i can be represented as a vector e_{v_i} after being transformed by an embedding matrix W_{emb}. The set of all sessions are defined as S, where sessions belonging to the user u can be defined as $\mathrm{S}^u = \{S_i^u\}_{i=1}^n$, where n is the number of sessions for user u. From S^u, the current session in which we make recommendation is defined as $S_c^u = S_n^u$, while the historical sessions before S_c^u are defined as $\mathrm{S}_h^u = \{S_i^u\}_{i=1}^{n-1}$. The neighbor sessions of the current session from other users are defined as $\mathrm{S}_n^u = \{S^j\}_{j=1}^K$, where K is the specified number of neighbors. Each session $S_i^u = \{v_{i,j}\}_{j=1}^{m_i}$ is an interaction sequence from the corresponding user u, where $v_{i,j} \in I$ represents the interactive item in the session S_i^u and m_i is the number of items in S_i^u. Sessions and items are ordered by timestamps. Given users' current session S_c^u, the target of session-based recommendation is to predict the next clicled item v_{i,m_i+1} for S_c^u based on the historical sessions S_h^u and the neighbor sessions S_n^u.

Fig. 2 presents an overview of our proposed TA-GNN model, which mainly consists of three modules, i.e., the profile-based representation module (see §III-B), the neighborhood-based representation module (see §III-C) and the prediction module (see §III-D). In the profile-based representation module, the user behavior graph $\mathcal{G}^u$ is created based on users' current session S_c^u and historical sessions S_h^u. Then $\mathcal{G}^u$ is used as the input to the profile-based generator, which aims to generates the profile-based representation s_p for the current session. Similarly, in the neighborhood-based representation module, the neighborhood graph $\mathcal{G}^n$ is built by looking up the neighbor relationship between the items of the current session S_c^u and of the neighbor sessions S_n^u. Then $\mathcal{G}^n$ is used to generate the neighbor-based presentation s_n for the current session. Finally, we concatenate s_p and s_n to obtain a unified presentation s of the current session. Using s, our model outputs the probability vector $\hat{y}$ which indicates the recommendation scores of the candidate items.

B. PROFILE-BASED REPRESENTATION MODULE

The profile-based representation module aims to represent the current session from the perspective of user's own behavior information. We build a user behavior graph based on uers's historical and current sessions, moreover design two GNN-based generators to learn the representation of current session.

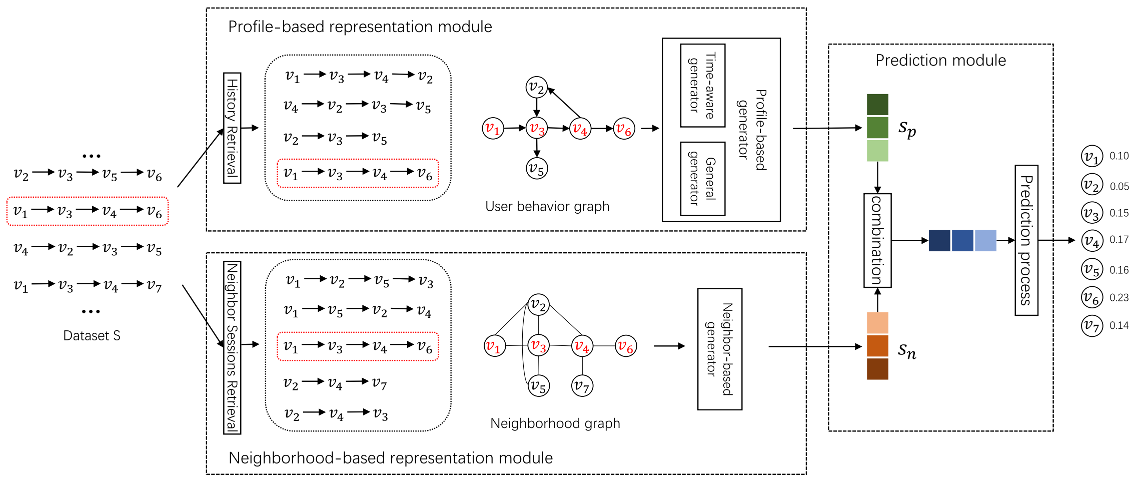


FIGURE 2. Framework of our proposed TA-GNN model. The sequence in the current session is with a red border and the items appear in the current session are highlighted in red in the user behaviour graph and neighbor diagrams.

User Behavior Graph is constructed by connecting items in the current session S_c^u and historical sessions S_h^u directionally based on their corresponding time orders. As shown in the profile-based representation module in Fig. 2, S_c^u and S_h^u are modeled as a directed graph $\mathcal{G}^u = (\mathcal{V}^u, \xi^u)$, where a node $i \in \mathcal{V}^u$ represents an item $v_i \in V$ that the user interacted with, an edge $e_{i,j} \in \xi^u$ between nodes v_i and v_j indicates that user u interacts with item v_j after v_i in one of his session. Based on the user behavior graph, we design a profile-based generator which includes two subordinate generators, i.e., *time-aware generator* and *general generator*, to enrich the analysis of transition relationships in the user behavior graph and generates the representation for the current session.

Time-Aware Generator builds two time-aware adjacency matrices $A_T^I, A_T^O \in \mathbb{R}^{m\times m}$ according to the time interval between two nodes. In detail, A_T^I and A_T^O indicate the in-degree and out-degree time-aware adjacency relationships respectively. We argue that two interactions are more relevant when the time interval between them is shorter, so we set the weight of the edges ξ^u inversely proportional to the time interval. In order to reduce the complexity of calculation, we adopt a linear interpolation to determine the weight between two nodes as follows:

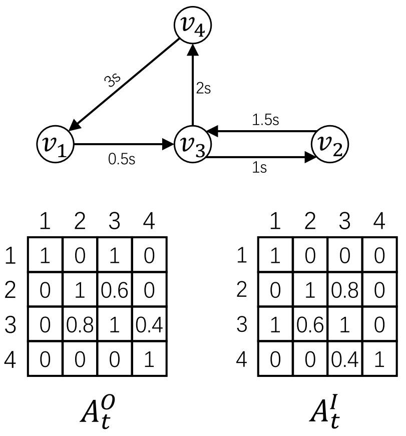
$$\omega_{i,j}^t = \begin{cases} \dfrac{t_{max} - t_{v_i,v_j}}{t_{max} - t_{min}}, & i \neq j \\ 1, & i = j \end{cases} \tag{1}$$

where t_{v_i,v_j} is the time interval between nodes v_i and v_j, if there are multiple edges from v_i to v_j, t_{v_i,v_j} is generated by averaging the time intervals of these edges. t_{max} and t_{min} are the longest and the shortest time intervals, respectively. $\omega_{i,j}^t$ denotes the time-aware weight from v_i to v_j, which is directed and tends to increase from 0 to 1 when the time interval decreases from t_{max} to t_{min}. In order to consider the self-effect of nodes, $\omega_{i,i}^t, i \in [1, m]$ are set to 1. Then the time-aware transition relationships in the user behavior graph $\mathcal{G}^u$ can be represented by the two adjacency matrices A_T^I and A_T^O, which can be written as:

$$A_T^O[i,j] = \omega_{i,j}^t \tag{2}$$

$$A_T^I[i,j] = \omega_{j,i}^t \tag{3}$$

Fig. 3 gives an example on how A_T^O and A_T^I are generated from a graph. If there is no in-degree or out-degree edge between two nodes, the corresponding position of A_T^O or A_T^I is filled with 0.



$$A_t^O: \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0.6 & 0 \\ 3 & 0 & 0.8 & 1 & 0.4 \\ 4 & 0 & 0 & 0 & 1 \end{array} \qquad A_t^I: \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0.8 & 0 \\ 3 & 1 & 0.6 & 1 & 0 \\ 4 & 0 & 0 & 0.4 & 1 \end{array}$$

FIGURE 3. A example of a graph and the time-aware adjacency matrices A_t^O, A_t^I. The time intervals between nodes are written on the edge.

General Generator builds two general adjacency matrices $A_G^I, A_G^O \in \mathbb{R}^{m\times m}$, indicating in-degree and out-degree general adjacency relationships respectively. Different from the time-aware generator, the general generator calculates the directed weights between two nodes based on the number of links between them. The weights can be denoted as follows:

$$\omega_{i,j}^{g-out} = \frac{num(v_i, v_j)}{\sum_{k\in N_{out}(i)} num(v_i, v_k)}, \tag{4}$$

$$\omega_{i,j}^{g-in} = \frac{num(v_j, v_i)}{\sum_{k\in N_{in}(i)} num(v_k, v_i)}, \tag{5}$$

where $num(x, y)$ indicates the number of occurrence that item y is interacted by the user after item x. N_{out} represents the set of predecessor nodes of v_i on a edge and N_{in} represents the set of successor nodes of v_i. Then the general adjacency can be written as:

$$A_G^O[i,j] = \omega_{i,j}^{g-out}, \tag{6}$$

$$A_G^I[i,j] = \omega_{i,j}^{g-in}, \tag{7}$$

Similarly with the time-aware generator, $\omega_{i,i}^{g-out}$ and $\omega_{i,i}^{g-in}$ are also set to 1 considering the influence of the nodes on themselves.

After obtaining the A_T^O, A_T^I and A_G^I, A_G^O, we aggregate information for the target node v_i according to the above four adjacency matrices. The aggregation process contains t step with identical operation to update the representations of nodes. The t-th step can be denoted as follow:

$$\begin{aligned} a_i^{O,t}(time) &= A_{T-i}^O([h_1^{t-1}, \cdots, h_m^{t-1}]W_T^O) + b_t^O, \\ a_i^{I,t}(time) &= A_{T-i}^I([h_1^{t-1}, \cdots, h_m^{t-1}]W_T^I) + b_t^I, \\ a_i^{O,t}(gen) &= A_{G-i}^O([h_1^{t-1}, \cdots, h_m^{t-1}]W_G^O) + b_g^O, \\ a_i^{I,t}(gen) &= A_{G-i}^I([h_1^{t-1}, \cdots, h_m^{t-1}]W_G^I) + b_g^I, \end{aligned} \tag{8}$$

where $A_{T-i}^O, A_{T-i}^O, A_{G-i}^O$ and A_{G-i}^I are the $i-th$ rows of elements in four adjacency matrices respectively corresponding to the node v_i. $W_T^O, W_T^I, W_G^O, W_G^I \in \mathbb{R}^{d\times d}$ represent parameter matrices. $b_t^O, b_t^I, b_g^O, b_g^I \in \mathbb{R}^d$ are bias vectors. $[h_1^{t-1}, \cdots, h_m^{t-1}]$ is the list of nodes representation vectors at $t-1$ step. It is worth noting that h_i^0 represents the item embedding e_{v_i} for node v_i when $t = 1$ at the beginning of the aggregation process. After drawing out the contextual information, we concatenate them to keep information integrity and obtain final contextual information representation of node v_i, which can be written as the following operation:

$$a_i^t = [a_i^{O,t}(time)||a_i^{I,t}(time)||a_i^{O,t}(gen)||a_i^{I,t}(gen)], \tag{9}$$

where $||$ is the concatenation operation. $a_i^t \in \mathbb{R}^{1\times 4d}$ is the vector representation of node v_i, which contains four types of contextual information. Then we incorporate the contextual information with the nodes representation of the previous timestep to update each node's representation based on a gated recurrent units (GRU) [37] as follows:

$$\begin{aligned} z_i^t &= \sigma(W_z a_i^t + U_z h_i^{t-1}), \\ r_i^t &= \sigma(W_r a_i^t + U_r h_i^{t-1}), \\ \tilde{h}_i^t &= tanh(W_o a_i^t + U_o(r_i^t \odot h_i^{t-1})), \\ h_i^t &= (1 - z_i^t) \odot h_i^{t-1} + z_i^t \odot \tilde{h}_i^t, \end{aligned} \tag{10}$$

where $W_z, W_r, W_o \in \mathbb{R}^{4d\times d}$, $U_z, U_r, U_o \in \mathbb{R}^{d\times d}$, are learnable parameters. $\sigma(\cdot)$ is the sigmoid function and $\odot$ represents element-wise multiplication. z_i^t and r_i^t are the reset and update gates respectively, which control preserving and discarding the information respectively. After t update steps, we get the final representation v_i^t of each node in $\mathcal{G}^u$. For simplicity, we use h_i instead of h_i^t.

Then we select item representations in the current session to characterize the profile-based presentation s_p. Take Fig. 2 as an example, the items in the current session are v_1, v_3, v_4 and v_6, after aggregation process, their corresponding representations which can be defined as h_1, h_3, h_4 and h_6, respectively, will be selected to build s_p. SR-GNN [5] and NARM [3] adopt a soft-attention mechanism, which are proved to have a strong ability to extract the user's main purpose in the current session. Inspired by the success of the soft-attention, we adopt the same way to build global representation and local representation respectively. The global representation can be calculate as:

$$s_p^{global} = \sum_{v_i \in S_c^u} \alpha_i h_i, \tag{11}$$

$$\alpha_i = W_0 \sigma(W_1 h_m + W_2 h_i + b_c), \tag{12}$$

where $W_0 \in \mathbb{R}^d$ is a parameter vector, $W_1, W_2 \in \mathbb{R}^{d\times d}$ are parameter matrices, $b_c \in \mathbb{R}^d$ is a bias vector. $\sigma(\cdot)$ indicates the sigmoid function and α_i is the weight of v_i in the current session. The local representation is always simply defined as the representation of the last item in the current session:

$$s_p^{local} = h_m, \tag{13}$$

Finally, in order to maintain the integrity of local information and global information, we adopt concatenation operation to aggregate s_p^{global} and s_p^{local} as the profile-based representation for the current session:

$$s_p = [s_p^{global}||s_p^{local}] \tag{14}$$

C. NEIGHBORHOOD-BASED REPRESENTATION MODULE

We build a Neighborhood Graph $\mathcal{G}^n$ based on the current session and its neighbor sessions to incorporate the collaborative information to enrich the representation learning of the current session. The process consists of three steps, namely *neighbor sessions retrieval*, *neighbor graph construction* and *neighbor-based generator.*

Neighbor Sessions Retrieval aims to construct neighbor set N_s for the current session. Following [36], we start by looking for the possible neighbor sessions which have co-occurrence items with the current session from the entire dataset. Then we adopt a strategy of selecting m most recent sessions from the possible neighbor sessions to build candidate neighbor session set $\mathbf{S}$, which is proved to be effective in [38]. In our paper, we set $m = 1000$ based on [38]. After obtaining $\mathbf{S}$, we begin to select neighbor sessions of the current session from it. First, we calculate similarities between the current session S_c^u and every other session $S_o \in \mathbf{S}$ based on the number of the co-occurrence item as follows:

$$sim(S_c^u, S_o) = \frac{co(S_c^u, S_o)}{\sqrt{l(S_c^u) \cdot l(S_o)}}, \tag{15}$$

where $co(S_c^u, S_o)$ is the number of items that appear in both S_c^u and S_o. $l(S_c^u)$ and $l(S_o)$ indicates the length of S_c^u and S_o respectively. After filtering out the sessions of which the similarity is lower than 0.5 in $\mathbf{S}$, we select top-k similar sessions to construct the neighbor set N_s for the current session.

Neighbor Graph Construction: Based on N_s and current session sequence, we build a neighborhood graph $\mathcal{G}^n$ to model the transition relationship between items in the current session and its neighbor sessions. As shown in Fig. 2, only the first-order neighbors of items in the current session are included in the neighborhood graph. In that graph, each edge(v_{i-1}, v_i) is single and undirected, which represents that item v_i is interacted before or after item v_{i-1}. The motivation of why we use undirected graph is similar as in [36], which argues that related items might be located in different relative positions for target items in the current session.

Neighbor-Based Generator: Similarly with in the profile-based generator, the first step in neighbor-based generator is to obtain representation for each nodes in the neighborhood graph. We do not consider time interval between two nodes in the neighborhood graph $\mathcal{G}^n$, because its nodes are interacted by different users, whose average interaction time intervals are different. This means that even facing the same two nodes, different users may interact in different time intervals. In this case, using time interval to judge the the correlation between two nodes may disturbs the performance of recommendation. Furthermore, we do not consider in-degree and out-degree transition between two nodes, as $\mathcal{G}^n$ is single undirected. Thus, we only use the graph's adjacency matrix A_N to model the transitions between nodes. In particular, if there is edge e_{ij} between two nodes v_i and v_j, we set the corresponding element A_{ij} to 1, otherwise to 0. We further set the diagonal elements A_{ii} to 1 to ensure the self-correlation of nodes in $\mathcal{G}^n$. Learning item representations in neighbor-based generator is similar with the process from the equation (8) to the equation (10), which can be written as follows:

$$\begin{aligned} a_i^t &= A_{N-i}([h_1^{t-1}, \cdots, h_m^{t-1}]W_N) + b_N, \\ z_i^t &= \sigma(W_z' a_i^t + U_z' h_i^{t-1}), \\ r_i^t &= \sigma(W_r' a_i^t + U_r' h_i^{t-1}), \\ \tilde{h}_i^t &= tanh(W_o' a_i^t + U_o'(r_i^t \odot h_i^{t-1})), \\ h_i^t &= (1 - z_i^t) \odot h_i^{t-1} + z_i^t \odot \tilde{h}_i^t, \end{aligned} \tag{16}$$

where the corresponding parameters play the same role as the parameters in equation (10) and he upper right slash is used to distinguish them. In order to distinguish item representations obtained by profile-based representation module, we use $\bar{h}_i$ to denote the updated latent vector of node v_i in the neighborhood graph in the neighborhood-based representation module.

After obtaining the representations of each nodes, the same attention mechanism as in profile-based representation module are used to model neighborhood-based representation of the current session as follows:

$$\begin{aligned} s_n^{global} &= \sum_{v_i \in S_c^u} \alpha_i \bar{h}_i, \\ \alpha_i &= W_0' \sigma(W_1' \bar{h}_m + W_2' \bar{h}_i + b_c'), \\ s_n^{local} &= \bar{h}_m, \\ s_n &= [s_n^{global} || s_n^{local}], \end{aligned} \tag{17}$$

where the parameters W_0', W_1', W_2', b_c' play the similar role as W_0, W_1, W_2, b_c in the profile-based module. s_n indicates the neighborhood-based representation for the current session S_c^u.

D. PREDICTING PROCESS

After obtaining the profile-based representation s_p and the neighborhood-based representation s_n, which aggregates user's long-term behavior and collaborative information into the current session respectively, we concatenate them together to construct the final representation for the current session:

$$s = [s_p || s_n]. \tag{18}$$

Then we can give recommendation scores for each candidate item in the whole dataset by combining s with global items embedding matrix W_{emb}:

$$\hat{z} = W_{emb} \cdot B \cdot s_n, \tag{19}$$

where $B \in \mathbb{R}^{d \times 4d}$ is a learnable parameters and $\hat{z} \in \mathbb{R}^{|I|}$ is the score vector which denotes the recommendation scores over all candidate items. Finally we apply a softmax function to generate the final probability distribution vector $\hat{y}$

$$\hat{y} = softmax(z), \tag{20}$$

where the corresponding position $\hat{y}_i$ in $\hat{y}$ indicates the probabilities of item i becoming the final choose of user u in the current session S_c^u.

To learn the parameters in our models, we take each session as a sample and process them separately. In the training phases, a fixed number of samples are filled into a mini-batch and the gradient descent algorithm is used on cross-entropy loss:

$$L(y, \hat{y}) = -\sum_{i=1}^{m} [y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i)], \tag{21}$$

where y is a one-hot encoding vector and indicates the truly probability distribution. Our model is trained by a Back-Propagation Through Time (BPTT) algorithm [39] in the learning process.

IV. EXPERIMENTS

In this section, we first present the datasets in Section IV-A. Then research questions and evaluation metrics are described in Section IV-B. Finally we briefly introduce the baseline models used for comparison in this paper and the parameter settings in Section IV-C.

A. DATASETS

We conduct experiments on three real-world datasets, namely 30MUSIC, Nowplaying and Tmall. *30MUSIC* is collected based on the listening events in a whole year through Last.fm API [40]. *Nowplaying* is created from music-related tweets, where users posted which tracks they were currently listening [41]. *Tmall* is published by Tmall competition, which is composed of interaction logs of the tmall.com for one year [38].

Following our previous work [42], we preprocess each dataset as follows. First, with the purpose to ensure each session has a sequential form, we filter out the sessions containing more than one interaction. Then, we remove items with supports less than 10 to improve the item density and reduce the size of items embedding matrix to make training faster. Finally, we delete the users with less than two sessions from each dataset to ensure that each user graph contains interactions from historical sessions. The statistics of the datasets after preprocessing are shown in Table 1.

TABLE 1. Main properties of the datasets.

Datasets	30MUSIC	Nowplaying	Tmall
Users	16,462	7,967	14,815
Items	58,290	30,673	44,518
Sessions	128,773	126,249	155,187
Interactions	1,481,092	976,702	1,180,493
Average session length(num)	11.50	7.73	7.60
Sessions per user	7.82	15.85	10.47
Training-interactions	1,303,067	919,416	926,175
Test-interactions	178,025	57,286	254,318

For each user, the last session are picked up as the test set and the previous sessions as the training set. In every session, the last interaction is selected as the ground truth for training and the remaining interactions together with the interactions in the historical sessions from the same user and the interactions in neighbor sessions are used as the input to SBRS models.

B. RESEARCH QUESTIONS AND EVALUATION METRICS

We list the following research questions to guide our experiments:

RQ1 How do our proposed TA-GNN model perform in session-based recommendation task compared with the state-of-the-art baselines?

RQ2 How do time-aware generator and neighbor information affect recommendation? Do they really help boost performance of our model?

RQ3 How do TA-GNN and baselines perform under different amount of historical interactions? Do historical interactions really help the recommendation in the current session?

RQ4 How does the number of interactions in current session affect our model and baselines?

With the purpose to answer these questions, we use Recall@N and MRR@N, which are frequently-used metrics in sequential recommendation, to measure the effect of the recommendation given by different models. In the following, the test dataset is defined as $Test$, the number of test samples is $|Test|$.

- **Recall@N** indicates the proportion of the correctly recommended items amongst top-N items, which can be calculated as follows.

$$Recall@N = \frac{1}{|Test|}\sum_{i\in Test} hit_i, \tag{22}$$

where N indicates the number of recommended items and $hit_i = 1$ when the correctly recommended item i is in the recommendation list, otherwise $hit_i = 0$.

- **MRR@N** is the average of reciprocal ranks of the correctly recommended items, which is calculated as follows.

$$MRR@N = \frac{1}{|Test|}\sum_{1\in Test} \frac{1}{rank_i}, \tag{23}$$

where $rank_i$ is the ranking of the correctly recommended items in the recommendation lists. The reciprocal ranks is set to zero when the ranking is larger than N.

For simplicity, we set $N = 20$ to compare the model performance.

C. BASELINES AND PARAMETER SETTINGS

We compare our TA-GNN model to six methods to examine its effectiveness on SBRS.

- **Pop** recommends the most popular items in the dataset to each user, though simple, Pop is a strong baseline in many studies.
- **Item-KNN** [9] adopts the idea of nearest neighbor and recommends items most similar to the last clicked item in the session to the user. The similarity of two items is defined as the frequency with which they appear in the same session.
- **NARM** [3]: employs an attention mechanism to capture user's main purpose and sequential behavior based on a RNN structure. It is an essential RNN-based model in SBRS.
- **I3GN** [36] constructs intra-session graph and inter-session graph for the current session to capture the current sequence and collaborative information. GNN and multi-head attention are employed to encode information of two graphs, respectively.
- **BINN** [21] adopt a behavior-intensive neural network on the whole historical interaction sequence to incorporate both long-term preference and short-term consumption motivations to make recommendation in SBRS.
- **A-PGNN** [8] construct user behavior graph by connecting the interactions in the current session and historical sessions. Based on the graph, they use GNN to learn multi-level transactions between items and then use self-attention to explicitly model the effect of historical sessions on the current session.

Among the above baselines, Pop and Item-KNN are traditional model and do not utilize any form of neural network. The rest four baselines are based on deep learning technologies, where NARM and I3GN only focus on the sequence in current session while BINN and A-PGNN consider historical interactions. I3GN and A-PGNN adopt GNN as main structure while NARM and BINN are RNN-based models. During

TABLE 2. Model performance in terms of Recall@20 and MRR@20. The results produced by the best performer and the best baseline in each column are boldfaced and underlined. (♦ means our model is statistical significantly better than the best baseline on a *t*-test for $\alpha = .01$).

Method	30MUSIC		Nowplaying		Tmall	
	Recall@20	MRR@20	Recall@20	MRR@20	Recall@20	MRR@20
Pop	.0098	.0025	.0206	.0061	.0954	.0503
Item-KNN	.0114	.0027	.0351	.0074	.1424	.0461
NARM	.2115	.1284	.1015	.0504	.2844	.1790
I3GN	.2651	.1326	.1183	.0518	.3152	.2208
BINN	.2472	.1312	.1156	.0576	.2727	.1620
A-PGNN	.2632	.1421	.1288	.0597	.3073	.2294
TA-GNN	**.2843**♦	**.1548**♦	**.1574**♦	**.0669**♦	**.3594**♦	**.2451**♦

training for all neural modes, we use Adam Optimizer [43] with learning rate 0.01 to learn the hyper parameters. The mini-batch size is set as 256.

V. RESULTS AND DISCUSSION

A. OVERALL PERFORMANCE

To answer **(RQ1)**, we compare our TA-GNN model to baselines in terms of Recall@20 and MRR@20 on three datasets. The result is shown in Table 2.

Let us discuss baselines first. As shown in Table 2, we can find traditional methods, i.e., Pop and Item-KNN, both lag obviously behind the neural models. This indicates neural networks are powerful in SBRS. Comparing two GNN-based baslines, i.e., I3GN and A-PGNN, we can find that A-PGNN often leads in terms of MRR@20 and I3GN is more advantageous in terms of Recall@20. This may reveal that the addition of historical interactions contributes more to the promotion of MRR, while the collaborative filtering information from neighbors contributes more to the promotion of Recall.

Then we focus on the comparison between our model and baselines. From Table 2 we can see that TA-GNN shows obvious superiority over baselines on all datasets in terms of both metrics, which clearly demonstrates its effectiveness on the session-based recommendation task. Specifically, TA-GNN respectively produces 7.2%, 22.2% and 14.0% improvements over the best baseline on three dataset in terms of Recall@20, and the improvements in terms of MRR@20 reach up to 8.9%, 12.0% and 6.8%. As has been demonstrated in [44] that, although the performance of existing models on the offline datasets seem to be at a low level in terms of Recall and MRR, these models still have practical significance in real-world applications. Moreover, even small improvements in the performance of offline models can increase user satisfaction in actual online testing. Compared with RNN-based models, i.e., NARM and BINN, our TA-GNN model uses GNN as main structure and shows obvious superiority. This indicates that GNN is more powerful than RNN in SBRS, which may be due to the fact that GNN has strong ability to handle the complex transitions in session sequence. Further compared to I3GN, our model applies user's historical interactions to personalize the recommendation and achieves average increases of 18.1% and 18.9% on three datasets in terms of Recall@20 and MRR@20, respectively. It indicates that the participation of historical information can really help boost the recommendation. Compared to A-PGNN, which gains best performance in most times among baselines, TA-GNN achieves significant improvements. We attribute this performance improvement to the time-aware generator and the used neighbor information in our model taht greatly help enrich the recommendation. Thus we study the effect of time-aware generator and neighbor information in V-B.

B. COMPONENTS ANALYSIS

Next, we turn to **(RQ2)**. We compare TA-GNN with its different variants to confirm the effectiveness of the time-aware generator and the use of neighbor information. The variant models include:

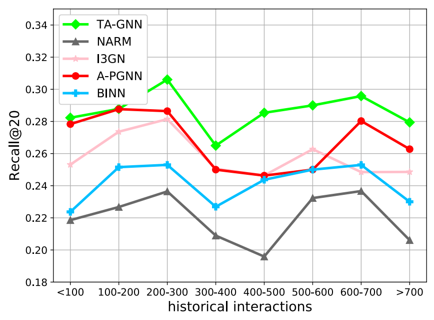
- **TA-GNN(-T-N)**: TA-GNN without the time-aware generator and the neighborhood-based representation module. This variant model makes recommendation only based on user behavior graph and the general generator.
- **TA-GNN(-N)**: TA-GNN without neighborhood-based representation module and only applies user behavior graph with both time-aware generator and general generator to model the current session.
- **TA-GNN(-T)**: TA-GNN without the time-aware generator in profile-based module. This variant uses general graph neural network to aggregate information from user behavior graph and neighborhood graph respectively.

The performance of different variants is presented in Table 3. We can draw the following conclusions:

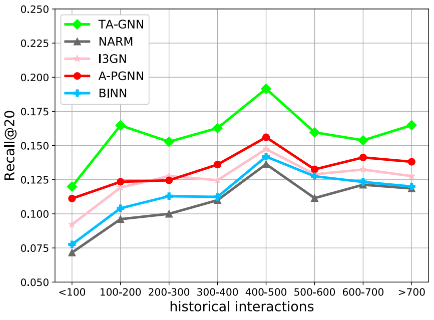
- We can see that TA-GNN(-N-T) lags behind TA-GNN(-T) and TA-GNN(-N) in terms of both metrics, which illustrates that both the utilization of time-aware generator and the collaborative information from neighbors can help improve the performance in terms of both Recall@20 and MRR@20.
- Comparing TA-GNN(-T) and TA-GNN(-N), we can find that TA-GNN(-T) performs better in terms of Recall@20 while TA-GNN(-N) shows advantages in terms of MRR@20. This phenomenon can be explained from two aspects: on one hand, time-aware generator can

TABLE 3. Performance of different variants.

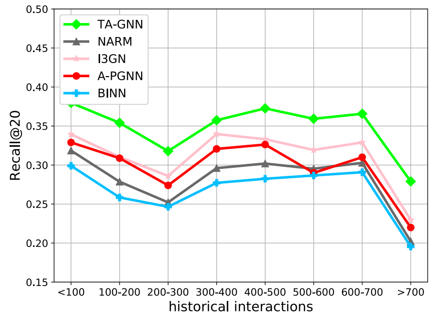
Variant	30MUSIC		Nowplaying		Tmall	
	Recall@20	MRR@20	Recall@20	MRR@20	Recall@20	MRR@20
TA-GNN(-N-T)	.2443	.1374	.1327	.0511	0.2946	0.1835
TA-GNN(-T)	.2671	.1391	.1425	.0593	0.3104	0.2145
TA-GNN(-N)	.2638	.1457	.1422	0.601	0.3274	0.2197
TA-GNN	.2843	.1548	.1574	.0639	.3594	.2451



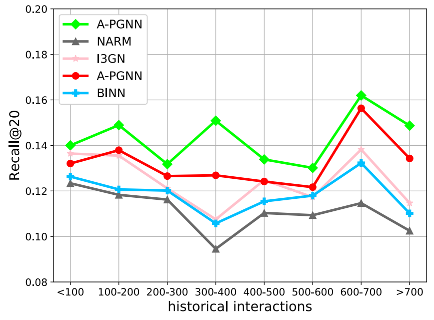
(a) Performance in terms of Recall@20 on 30MUSIC.



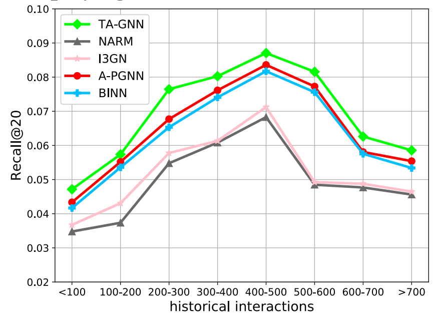
(b) Performance in terms of Recall@20 Nowplaying.



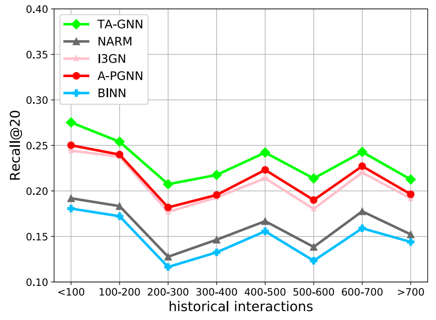
(c) Performance in terms of Recall@20 on Tmall.



(d) Performance in terms of MRR@20 on 30MUSIC.



(e) Performance in terms of MRR@20 on Nowplaying.



(f) Performance in terms of MRR@20 on Tmall.

FIGURE 4. Effect on the performance of neural models in terms of Recall@20 and MRR@20 with different numbers of historical interactions, tested on three datasets.

help accurately analyse the correlation between different interactions of the same user, so as to deeply understand the user's behaviors and preferences. This can help rank items closest to the user's preference at top positions in the recommendation list, so that MRR is improved; On the other hand, neighborhood graph contains the information about the similarities between items, which can help to find the most similar items for the current session, and further increase the probability that the ground truth is included in the list.

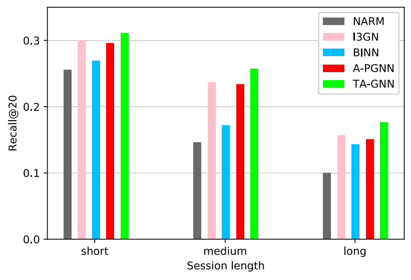
- Compared with TA-GNN(-T) and TA-GNN(-N), TA-GNN has a significant improvement in terms of both metrics, which indicates that sequential recommendations contain complex transition relationships, and it is necessary to take various factors into account to make accurate personalized recommendations.

C. IMPACT OF HISTORY LENGTH

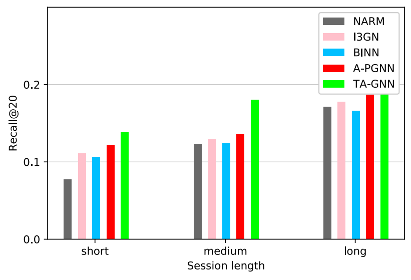
To answer **RQ3**, we group the test set according to the number of historical interactions, which we define as N. We divide the results into the following eight groups: $N < 100$, $N \in [100, 200]$, $N \in [200, 300]$, $N \in [300, 400]$, $N \in [400, 500]$, $N \in [500, 600]$, $N \in [600, 700]$, $N > 700$. Considering that neural models achive significant superiority over traditional methods (see Table. 2), we only present the results of neural models in the following experiments. Fig. 4 shows the results.

As shown in Fig. 4, our proposed TA-GNN achieves best performance with various history length on all datasets, which presents its robustness to different historical interaction number conditions.

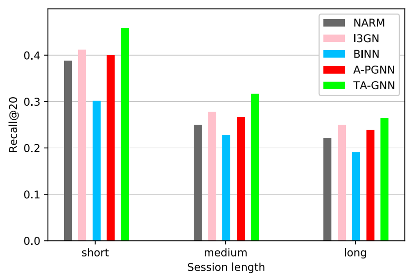
There is also an interesting phenomenon: TA-GNN shows greater advantages when longer history is available. For example, the performance gap between TA-GNN and A-PGNN, i.e., the best baseline, reaches maximum at the fifth group ($N \in [400, 500]$) on both 30MUSIC and Nowplaying in terms of Recall@20. Similarly, the performance gap between TA-GNN and I3GN are most obvious at the sixth group($N \in [500, 600]$) on Tmall in terms of Recall@20. This indicates that an appropriate number of historical interactions can help TA-GNN show greater advantages, which may be



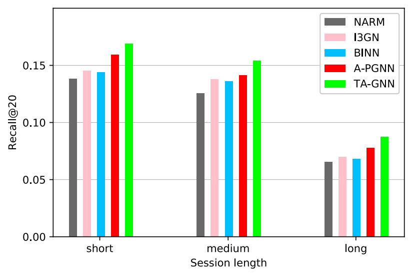
(a) Performance in terms of Recall@20 on 30MUSIC.



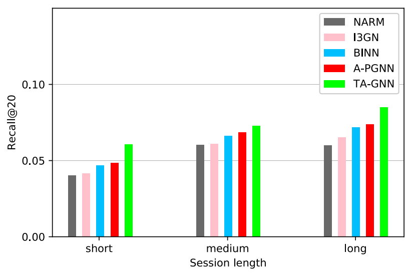
(b) Performance in terms of Recall@20 Nowplaying.



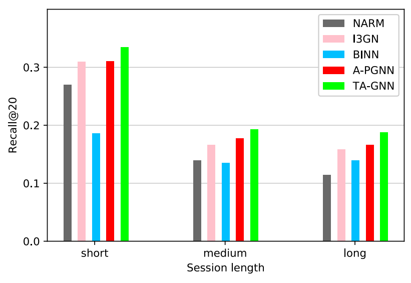
(c) Performance in terms of Recall@20 on Tmall.



(d) Performance in terms of MRR@20 on 30MUSIC.



(e) Performance in terms of MRR@20 on Nowplaying.



(f) Performance in terms of MRR@20 on Tmall.

FIGURE 5. Effect on the performance of neural models in terms of Recall@20 and MRR@20 with different session length, tested on three datasets.

due to the fact that too few historical interactions are not conducive to extract personalized information while too many historical interactions may interfere with the intent analysis of the current session.

D. IMPACT OF SESSION LENGTH

To answer **RQ4**, we examine model performance with different length of current session. We divide the test sets of three datasets into three parts according to their corresponding current session length, i.e., short (no more 5 clicks), medium (6 to 15 clicks) and long (more than 15 clicks). The result is plot in Fig. 5.

From the results we can draw some interesting conclusions. In 30MUSIC and Tmall, all the methods achieve their best performance at the short group in terms of both metrics on all datasets. It actually reflects the fact that the longer a user's current session is, the more difficult it is to get an accurate prediction of the user's true intentions, as the user's interest shifts as he clicks away. Conversely, if the user's current session is short, which indicates his intention is straightforward, it is easier to make recommendations. However, on Nowplaying dataset, model performance seems to be better on long sessions. This may be attributed to that the user's history and neighborhood information are sparse on Nowpalying dataset, and more information from current session helps make appropriate recommendations.

Compared to baselines, the proposed TA-GNN model consistently achieves the best performance on all datasets under various session lengths. The superiority of our model tends to be more obvious when the sessions are longer. For example, TA-GNN achives 17% improvement over A-PGNN on long sessions, while only 5% on short sessions in terms of Recall@20 on 30MUSIC. This shows that our model has a stronger grasp of complex user intentions than other models.

VI. CONCLUSIONS AND FUTURE WORK

We propose a time-aware graph neural network for session-based recommendation task, which can leverage user's historical interactions, neighbor information and time interval to help generate the representation for user's current session. Our extensive experiments on three real-world datasets show the superiority of our model over the state-of-art baselines in terms of Recall and MRR. From the analysis of the components of our model, we find that our proposed time-aware generator can really boost the recommendation, especially can help improve the performance in terms of MRR, which provide a evidence that the time interval between two interactions reflects the correlation between them.

As for future work, we would like to examine the scalability of TA-GNN by evaluating its effectiveness on other datasets. Most importantly, We want to put our model into practical application and see how it behaves. In addition, we would like to investigate the performance of different point-wise loss or pair-wise loss functions on session-based recommendation.

REFERENCES

[1] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," 2015, *arXiv:1511.06939*. [Online]. Available: http://arxiv.org/abs/1511.06939

[2] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk, "Parallel recurrent neural network architectures for feature-rich session-based recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, Sep. 2016, pp. 241–248.

[3] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1419–1428.

[4] Y. K. Tan, X. Xu, and Y. Liu, "Improved recurrent neural networks for session-based recommendations," in *Proc. 1st Workshop Deep Learn. Recommender Syst. (DLRS)*, 2016, pp. 17–22.

[5] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural network," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 346–353.

[6] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou, "Graph contextualized self-attention network for session-based recommendation," in *Proc. IJCAI*, 2019, pp. 3940–3946.

[7] F. Yu, Y. Zhu, Q. Liu, S. Wu, L. Wang, and T. Tan, "TAGNN: Target attentive graph neural networks for session-based recommendation," 2020, *arXiv:2005.02844*. [Online]. Available: http://arxiv.org/abs/2005.02844

[8] S. Wu, M. Zhang, X. Jiang, K. Xu, and L. Wang, "Personalized graph neural networks with attention mechanism for session-aware recommendation," 2019, *arXiv:1910.08887*. [Online]. Available: https://arxiv.org/abs/1910.08887

[9] J. Davidson, B. Liebald, and J. Liu, "The YouTube video recommendation system," in *Proc. 4th ACM Conf. Recommender Syst. (RecSys)*, 2010, pp. 293–296.

[10] W. Meng, D. Yang, and Y. Xiao, "Incorporating user micro-behaviors and item knowledge into multi-task learning for session-based recommendation," 2020, *arXiv:2006.06922*. [Online]. Available: http://arxiv.org/abs/2006.06922

[11] F. Garcin, C. Dimitrakakis, and B. Faltings, "Personalized news recommendation with context trees," in *Proc. 7th ACM Conf. Recommender Syst. (RecSys)*, 2013, pp. 105–112.

[12] M. Hosseinzadeh Aghdam, N. Hariri, B. Mobasher, and R. Burke, "Adapting recommendations to contextual changes using hierarchical hidden Markov models," in *Proc. 9th ACM Conf. Recommender Syst. (RecSys)*, 2015, pp. 241–244.

[13] B. McFee and G. R. G. Lanckriet, "The natural language of playlists," in *Proc. ISMIR*, Oct. 2011, pp. 537–542.

[14] O. Moling, L. Baltrunas, and F. Ricci, "Optimal radio channel recommendations with explicit and implicit feedback," in *Proc. 6th ACM Conf. Recommender Syst. (RecSys)*, 2012, pp. 75–82.

[15] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-based recommender system," *J. Mach. Learn. Res.*, vol. 6, pp. 1265–1295, Sep. 2005.

[16] M. Tavakol and U. Brefeld, "Factored MDPs for detecting topics of user sessions," in *Proc. 8th ACM Conf. Recommender Syst. (RecSys)*, 2014, pp. 33–40.

[17] S. Rendle, C. Freudenthaler, and L. Schmidtthieme, "Factorizing personalized Markov chains for next-basket recommendation," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 811–820.

[18] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*. [Online]. Available: https://arxiv.org/abs/1506.00019

[19] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with Top-k gains for session-based recommendations," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 843–852.

[20] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, "Personalizing session-based recommendations with hierarchical recurrent neural networks," in *Proc. 11th ACM Conf. Recommender Syst.*, Aug. 2017, pp. 130–137.

[21] Z. Li, H. Zhao, Q. Liu, Z. Huang, T. Mei, and E. Chen, "Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1734–1743.

[22] K. Sun, T. Chen, T. Qian, Y. Chen, H. Yin, and L. Chen, "What can history tell us? Identifying relevant sessions for next-item recommendation," in *Proc. Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 1593–1602.

[23] W. Chen, F. Cai, H. Chen, and M. de Rijke, "A dynamic co-attention network for session-based recommendation," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 1461–1470.

[24] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "STAMP: Short-term Attention/Memory priority model for session-based recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1831–1839.

[25] G. Bonnin and D. Jannach, "Automated generation of music playlists: Survey and experiments," *ACM Comput. Surveys*, vol. 47, no. 2, p. 26, 2015.

[26] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation," in *Proc. 11th ACM Conf. Recommender Syst.*, Aug. 2017, pp. 306–310.

[27] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, and M. De Rijke, "A collaborative session-based recommendation approach with parallel memory modules," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2019, pp. 345–354.

[28] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2014, pp. 701–710.

[29] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web (WWW)*, 2015, pp. 1067–1077.

[30] P. Yanardag and S. V. N. Vishwanathan, "Deep graph kernels," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2015, pp. 1365–1374.

[31] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, pp. 498–511, Mar. 2009.

[32] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[33] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," 2015, *arXiv:1511.05493*. [Online]. Available: http://arxiv.org/abs/1511.05493

[34] E. Choi, M. T. Bahadori, L. Song, W. F. Stewart, and J. Sun, "Gram: Graph-based attention model for healthcare representation learning," in *Proc. 23rd Int. Conf. Knowl. Discovery Data Mining (ACM SIGKDD)*, 2017, pp. 787–795.

[35] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: https://arxiv.org/abs/1710.10903

[36] Y. Zheng, S. Liu, and Z. Zhou, "Balancing multi-level interactions for session-based recommendation," *CoRR*, vol. abs/1910.13527, Oct. 2019.

[37] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: http://arxiv.org/abs/1406.1078

[38] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms," *User Model. User-Adapted Interact.*, vol. 28, nos. 4–5, pp. 331–390, Dec. 2018.

[39] S.-C. Huang, P.-H. Hung, C.-H. Hong, and H.-M. Wang, "A new image blood pressure sensor based on PPG, RRT, BPTT, and harmonic balancing," *IEEE Sensors J.*, vol. 14, no. 10, pp. 3685–3692, Oct. 2014.

[40] R. Turrin, M. Quadrana, A. Condorelli, R. Pagano, and P. Cremonesi, "30Music listening and playlists dataset," in *Proc. 9th ACM Conf. Recommender Syst. (RecSys)*, Vienna, Austria, Sep. 2015.

[41] E. Zangerle, M. Pichl, W. Gassler, and G. Specht, "#Nowplaying music dataset: Extracting listening behavior from Twitter," in *Proc. 1st Int. Workshop Internet-Scale Multimedia Manage. (WISMM)*, 2014, pp. 21–26.

[42] Y. Guo, Y. Ling, and H. Chen, "A neighbor-guided memory-based neural network for session-aware recommendation," *IEEE Access*, vol. 8, pp. 120668–120678, 2020.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations (ICLR)*, San Diego, CA, USA, Y. Bengio and Y. LeCun, Eds., May 2015, pp. 1–15.

[44] J. Wang, P. Huang, H. Zhao, Z. Zhang, B. Zhao, and D. L. Lee, "Billion-scale commodity embedding for E-commerce recommendation in alibaba," in *Proc. 24th Int. Conf. Knowl. Discovery Data Mining (ACM SIGKDD)*, London, U.K., Jul. 2018, pp. 839–848.

YUPU GUO received the B.S. degree in information system engineering from the National University of Defense Technology, Hunan, China, in 2018, where he is currently pursuing the M.S. degree in management science and engineering.

His research interests include recommendation systems and information retrieval.

YANXIANG LING received the M.S. degree in information system engineering from the National University of Defense Technology, Hunan, China, in 2013, where she is currently pursuing the Ph.D. degree in management science and engineering.

Her research interests include dialog systems and information retrieval.

HONGHUI CHEN received the Ph.D. degree in operational research from the National University of Defense Technology, Hunan, China, in 2007.

He is currently a Professor with the National University of Defense Technology, Hunan, China. He has published several articles at SIGIR, IPM, and other top journals. His research interests include information systems and information retrieval.