Graph Contextualized Self-Attention Network for Session-based Recommendation

Chengfeng Xu[1,2], **Pengpeng Zhao**[1,2,3*], **Yanchi Liu**[4], **Victor S. Sheng**[5],
Jiajie Xu[1], **Fuzhen Zhuang**[3], **Junhua Fang**[1] and **Xiaofang Zhou**[6,2]
[1]Institute of AI, School of Computer Science and Technology, Soochow University, China
[2]Zhejiang Lab, China
[3]Key Lab of IIP of CAS, Institute of Computing Technology, Beijing, China
[4]Rutgers University, New Jersey, USA
[5]The University of Central Arkansas, Conway, USA
[6]The University of Queensland, Brisbane, Australia

Abstract

Session-based recommendation, which aims to predict the user's immediate next action based on anonymous sessions, is a key task in many online services ($e.g.$, e-commerce, media streaming). Recently, Self-Attention Network (SAN) has achieved significant success in various sequence modeling tasks without using either recurrent or convolutional network. However, SAN lacks local dependencies that exist over adjacent items and limits its capacity for learning contextualized representations of items in sequences. In this paper, we propose a graph contextualized self-attention model (GC-SAN), which utilizes both graph neural network and self-attention mechanism, for session-based recommendation. In GC-SAN, we dynamically construct a graph structure for session sequences and capture rich local dependencies via a graph neural network (GNN). Then each session learns long-range dependencies by applying the self-attention mechanism. Finally, each session is represented as a linear combination of the global preference and the current interest of that session. Extensive experiments on two real-world datasets show that GC-SAN outperforms state-of-the-art methods consistently.

1 Introduction

Recommender systems play an important role in helping users alleviate the problem of information overload and select interesting contents in many applications domains, $e.g.$, e-commerce, music, and social media. Most of existing recommender systems are based on user historical interactions. However, in many application scenarios, user identification may be unknown and there are only user historical actions during an ongoing session. To solve this problem, session-based recommendation is proposed to predict the next action ($e.g.$, click on an item) that a user may take based on the sequence of the user's previous behaviors in the current session.

Due to its highly practical value, many kinds of approaches for session-based recommendation have been proposed. Markov Chain (MC) is a classic example, which assumes that the next action is based on the previous ones [Rendle *et al.*, 2010]. With such a strong assumption, an independent combination of the past interactions may limit the accuracy of recommendation. Recent studies have highlighted the importance of using recurrent neural network (RNN) in session-based recommender systems and obtained promising results [Zhao *et al.*, 2019]. For instance, Hidasi et al. [Hidasi *et al.*, 2016] proposed to model short-term preferences with GRU (a variant of RNN), and then an improved version [Tan *et al.*, 2016] is proposed to further boost its recommendation performance. Recently, NARM [Li *et al.*, 2017] is designed to capture the user's sequential pattern and main purpose simultaneously by employing a global and local RNN. However, the existing methods usually model single-way transitions between consecutive items and neglect complex transitions among the entire session sequence.

More recently, a new sequential model, $Transformer$ [Vaswani *et al.*, 2017], has achieved state-of-the-art performance and efficiency in various translation tasks. Instead of using recurrence or convolution, Transformer utilizes an encoder-decoder structure composed of stacked self-attention network to draw global dependencies between input and output. Self-attention, as a special attention mechanism, has been widely used to model the sequential data and achieved remarkable results in many applications, $e.g.$, machine translation [Vaswani *et al.*, 2017], sentiment analysis [Lin *et al.*, 2017], and sequential recommendation [Kang and McAuley, 2018; Zhou *et al.*, 2018]. The success of the Transformer model can be attributed to its self-attention network, which takes full account of all signals with a weighted averaging operation. Despite its success, such an operation disperses the distribution of attention, which results in lacking local dependencies over adjacent items and limiting its capacity for learning contextualized representations of items [Liu *et al.*, 2019]. While the local contextual information of adjacent items has been shown that it can enhance the ability of modeling dependencies among neural representations, especially for the attention models [Yang *et al.*, 2018; Liu *et al.*, 2019].

[*] Pengpeng Zhao is the corresponding author. And his email is ppzhao@suda.edu.cn.

In this work, we propose to strengthen self-attention network through *graph neural network* (GNN). On the one hand, the strength of self-attention is to capture long-range dependencies by explicitly attending to all the positions. On the other hand, GNN is capable of providing rich local contextual information by encoding edge or node attribute features [Battaglia *et al.*, 2018]. Specifically, we introduce a graph contextual self-attention network, named GC-SAN, for session-based recommendation, which benefits from the complementary strengths of GNN and self-attention. We first construct a directed graph from all historical session sequences. Based on the session graph, GC-SAN is able to capture transitions of neighbor items and generate the latent vectors for all nodes involved in the graph correspondingly. Then we apply the self-attention mechanism to model long-range dependencies regardless of the distance, where session embedding vectors are composed by the latent vectors of all nodes in the graph. Finally, we use the linear weighted sum of the user's global interests and his/her local interests in that session as the embedding vector to predict the probability of clicking on the next item.

The main contributions of this work are summarized as follows.

- To improve the representation of session sequences, we present a novel graph contextual self-attention model based on graph neural network (GC-SAN). GC-SAN utilizes the complementarity between self-attention network and graph neural network to enhance the recommendation performance.
- Graph neural network is used to model local graph-structured dependencies of separated session sequences, while multi-layer self-attention network is designed to obtain contextualized non-local representations.
- We conduct extensive experiments on two benchmark datasets. Our experimental results show the effectiveness and superiority of GC-SAN, comparing with the state-of-the-art methods via comprehensive analysis.

2 Related Work

Session-based recommendation is a typical application of recommender systems based on implicit feedback, where users' identifications are unknown, and no explicit preferences (*e.g.*, ratings) but only positive observations (*e.g.*, purchases or clicks) are provided [He *et al.*, 2016]. These positive observations are usually in the form of sequential data obtained by passively tracking users' records over a sequence of time. In this setting, classical CF methods (*e.g.*, matrix factorization) break down because no user profile can be constructed from anonymous history interactions. A natural solution to this problem is the item-to-item recommendation approaches [He *et al.*, 2018]. In the session-based setting, an item-to-item similarity matrix is pre-computed for the available session data using simple item co-occurrence (frequent) patterns [Bonnin and Jannach, 2015]. Sarwar et al. [Sarwar *et al.*, 2001] analyzed different item-based recommendation generation techniques and compared their results with basic k-nearest neighbor approaches. Moreover, to model the sequence relationship between two adjacent actions, Rendle et al. [Rendle *et al.*, 2010] proposed to combine the power of MF and Markov Chain (MC) for next-basket recommendation. Though these methods are proved to be effective and widely employed, they only take into account the most recent click of the session, ignoring the global information of the whole click sequence.

Recently, researchers turn to neural networks and attention-based models for session-based recommender system. For instance, Hidasi et al. [Hidasi *et al.*, 2016] were among the first to explore Gated Recurrent Unit (GRU) as a special form of RNN for the prediction of the next action in a session, and later an improved version [Tan *et al.*, 2016] is proposed to boost its recommendation performance further. Nowadays, Graph Neural Network (GNN) has been proposed to learn the representation for graph structured data [Scarselli *et al.*, 2009; Wang *et al.*, 2019] in the form of RNN, which is broadly applied for the different tasks, e.g., image classification [Marino *et al.*, 2017], script event prediction [Li *et al.*, 2018] and recommender systems [Wu *et al.*, 2018].

On the other hand, several attention-based mechanisms have been introduced across various applications, *e.g.* natural language processing and computer vision. Standard vanilla attention mechanism has been incorporated into recommender systems [Li *et al.*, 2017; Liu *et al.*, 2018]. More recently, Vaswani et al. [Vaswani *et al.*, 2017] proposed to model the dependencies between words based entirely on self-attention without any recurrence or convolution, which has achieved state-of-the-art performance on machine translation task. Based on the simple and parallelized self-attention mechanism, Kang et al. [Kang and McAuley, 2018] proposed a self-attention based sequential model, which outperforms MC/RNN/RNN-based sequential recommendation methods. Huang et al. [Huang *et al.*, 2018] proposed a unified contextual self-attention network at feature level to capture the polysemy of heterogeneous user behaviors for sequential recommendation.

Most existing sequential recommendation models utilize the self-attention mechanism to capture distant item-item transitions in a sequence and have achieved state-of-the-art performance. However, it is still challenging for establishing complex contextual information between adjacent items. In this paper, we strengthen the self-attention network through graph neural network and meanwhile maintain the model's simplicity and flexibility. To the best of our knowledge, this is the first attempt to complement SAN and GNN for session-based recommendation, in which the first one can model the global item-item information of a session and the latter is capable of learning local contextual information by encoding attribute features of constructing graphs.

3 Graph Contextualized Self-Attention Network

In this section, we introduce the proposed contextualized self-attention recommendation model based on graph neural network (GC-SAN). We first formulate the problem of session-based recommendation, and then describe the architecture of our model in detail (As shown in Figure 1).

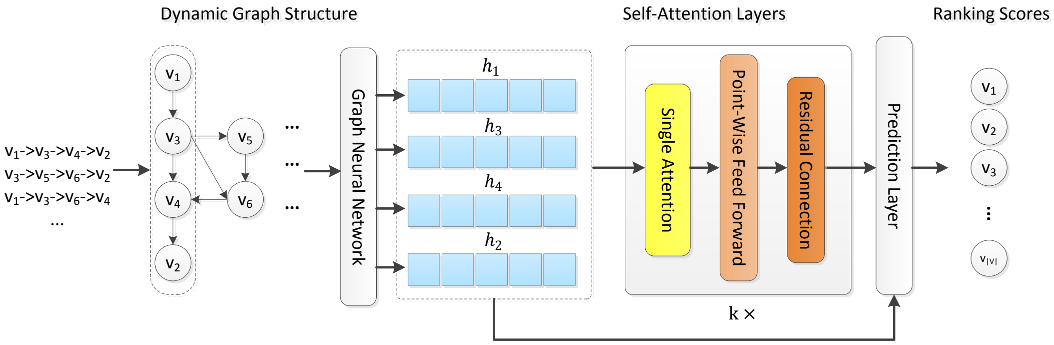
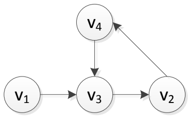


Figure 1: The general architecture of the proposed model. We first construct a directed graph of all session sequences. Based on the graph, we apply graph neural network to obtain all node vectors involved in the session graph. After that, we use a multi-layer self-attention network to capture long-range dependencies between items in the session. In prediction layer, we represent each session as a linear of the global preference and the current interest of that session. Finally, we compute the ranking scores of each candidate item for recommendation.

3.1 Problem Statement

Session-based recommendation aims to predict which item a user would like to click next, only based upon his/her current interaction sequence. Here we give a formulation of the session-based recommendation problem as below.

Let $V = \{v_1, v_2, ..., v_{|V|}\}$ denote a set of all unique items involved in all sessions. For each anonymous session, a sequence of clicked actions by the user are denoted as $S = \{s_1, s_2, ..., s_n\}$ in time order, where $s_t \in V$ represents a clicked item of the user at time step t. Formally, our model aims to predict the next possible click (*i.e.*, s_{t+1}) for a given prefix of the action sequence truncated at time t, $S_t = \{s_1, s_2, ..., s_{t-1}, s_t\}$ $(1 \leq t < n)$. To be exact, our model generates a ranking list over all candidate items that may occur in that session. $\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, ..., y_{|V|}\}$ denotes the output probability for all items, where $\hat{y}_i$ corresponds to the recommendation score of item v_i. Since a recommender typically makes more than one recommendation for the user, thus we choose the top-N items from $\hat{\mathbf{y}}$ for recommendation.



Incoming Matrix

	1	2	3	4
1	0	0	0	0
2	0	0	1	0
3	1/2	0	0	1/2
4	0	1	0	0

$\mathbf{M}^{I}$

Outgoing Matrix

	1	2	3	4
1	0	0	1	0
2	0	0	0	1
3	0	1	0	0
4	0	0	1	0

$\mathbf{M}^{O}$

Figure 2: An example of a session graph structure and the connection matrices $\mathbf{M}^I$ and $\mathbf{M}^O$.

3.2 Dynamic Graph Structure

Graph Construction. The first part of GNN is to construct meaningful graph from all sessions. Given a session $S = \{s_1, s_2, ..., s_n\}$, we treat each item s_i as a node and (s_{i-1}, s_i) as an edge which represents a user clicks item s_i after s_{i-1} in the session S. Therefore, each session sequence can be modeled as a directed graph. The graph structure is updated by promoting communication between different nodes. Specifically, let $\mathbf{M}^I, \mathbf{M}^O \in \mathbb{R}^{n \times n}$ denote weighted connections of outgoing and incoming edges in the session graph, respectively. For example, considering a session $S = \{s_1, s_3, s_2, s_4, s_3\}$, the corresponding graph and the matrix (*i.e.*, $\mathbf{M}^I$ and $\mathbf{M}^O$) are shown in Figure2. Since several items may appear in the session sequence repeatedly, we assign each edge with normalized weight, which is calculated as the occurrence of the edge divided by the outdegree of that edge's start node. Note that our model can support various strategies of constructing session graph and generate the corresponding connection matrices. Then we can apply the two weighted connection matrices with graph neural network to capture the local information of the session sequence.

Node Vectors Updating. Next, we present how to obtain latent feature vectors of nodes via graph neural network. We first convert every item $v \in V$ into an unified low-dimension latent space and the node vector $\mathbf{s} \in \mathbb{R}^d$ denotes a d-dimensional real-valued latent vector of item v. For each node s_t at time t in the graph session, given by the connection matrices $\mathbf{M}^I$ and $\mathbf{M}^O$, the information propagation between different nodes can be formalized as:

$$\begin{aligned} \mathbf{a}_t = Concat(&\mathbf{M}_t^I([\mathbf{s}_1, ..., \mathbf{s}_n]\mathbf{W}_a^I + \mathbf{b}^I), \\ &\mathbf{M}_t^O([\mathbf{s}_1, ..., \mathbf{s}_n]\mathbf{W}_a^O + \mathbf{b}^O)), \end{aligned} \quad (1)$$

where $\mathbf{W}_a^I, \mathbf{W}_a^O \in \mathbb{R}^{d \times d}$ are the parameter matrices. $\mathbf{b}^I, \mathbf{b}^O \in \mathbb{R}^d$ are the bias vectors. $\mathbf{M}_t^I, \mathbf{M}_t^O \in \mathbb{R}^{1 \times n}$ are t-th row of each matrix corresponding to node s_t, respectively. $\mathbf{a}_t$ extracts the contextual information of neighborhoods for node s_t. Then we take them and the previous state $\mathbf{s}_{t-1}$ as input and feed into the graph neural network. Thus, the final output $\mathbf{h}_t$ of GNN layer is computed as follows.

$$\begin{aligned}
\mathbf{z}_t &= \sigma(\mathbf{W}_z\mathbf{a}_t + \mathbf{P}_z\mathbf{s}_{t-1}),\\
\mathbf{r}_t &= \sigma(\mathbf{W}_r\mathbf{a}_t + \mathbf{P}_r\mathbf{s}_{t-1}),\\
\tilde{\mathbf{h}}_t &= tanh(\mathbf{W}_h\mathbf{a}_t + \mathbf{P}_h(\mathbf{r}_t \odot \mathbf{s}_{t-1})),\\
\mathbf{h}_t &= (1-\mathbf{z}_t)\odot \mathbf{s}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t.
\end{aligned} \tag{2}$$

where $\mathbf{W}_z, \mathbf{W}_r, \mathbf{W}_h \in \mathbb{R}^{2d\times d}$, $\mathbf{P}_z, \mathbf{P}_r, \mathbf{P}_h \in \mathbb{R}^{d\times d}$, are learnable parameters. $\sigma(\cdot)$ represents the logistic sigmoid function and $\odot$ denotes element-wise multiplication. $\mathbf{z}_t, \mathbf{r}_t$ are update gate and reset gate, which decide what information to be preserved and discarded, respectively.

3.3 Self-Attention Layers

Self-attention is a special case of the attention mechanism and has been successfully applied in lots of research topics including NLP [Vaswani *et al.*, 2017] and QA [Li *et al.*, 2019]. The self-attention mechanism can draw global dependencies between input and output, and capture item-item transitions across the entire input and output sequence itself without regard to their distances.

Self-Attention Layer. After feeding a session sequence into the graph neural network, we can obtain the latent vectors of all nodes involved in the session graph, *i.e.*, $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n]$. Next, we feed them into the self-attention layer to better capture the global session preference.

$$\mathbf{F} = softmax(\frac{(\mathbf{HW}^Q)(\mathbf{HW}^K)^T}{\sqrt{d}})(\mathbf{HW}^V)$$

where the projection matrices $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{2d\times d}$.

Point-Wise Feed-Forward Network. After that, we apply two linear transformations with a ReLU activation function to endow the model with nonlinearity and consider interactions between different latent dimensions. However, transmission loss may occur in self-attention operations. Thus we add a residual connection after the feed-forward network, which makes the model much easier to leverage low-layer information inspired by [Vaswani *et al.*, 2017].

$$\mathbf{E} = ReLU(\mathbf{FW}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 + \mathbf{F}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are $d \times d$ matrices, $\mathbf{b}_1$ and $\mathbf{b}_2$ are d-dimensional bias vectors. Moreover, to alleviate overfitting problems in deep neural networks, we apply "Dropout" regularization techniques during training. For simplicity, we define the above whole self-attention mechanism as:

$$\mathbf{E} = SAN(\mathbf{H}) \tag{5}$$

Multi-layer Self-Attention. Recent work shows that different layers capture different types of features. In this work, we investigate which levels of layers benefit most from the features modeling to learn more complex item transitions. The 1-st layer is defined as $\mathbf{E}^{(1)} = \mathbf{E}$. The k-th ($k > 1$) self-attention layer is defined as:

$$\mathbf{E}^{(k)} = SAN(\mathbf{E}^{(k-1)}) \tag{6}$$

where $\mathbf{E}^{(k)} \in \mathbb{R}^{n\times d}$ is the final output of the multi-layer self-attention network.

3.4 Prediction Layer

After several self-attention blocks that adaptively extract sequential information of sessions, we achieve the long-term self-attentive representation $\mathbf{E}^{(k)}$. To better predict the user's next clicks, we combine the long-term preference and the current interest of the session, and then use this combined embedding as the session representation. For a session $S = \{s_1, s_2, ..., s_n\}$, we take the last dimensions of $\mathbf{E}^{(k)}$ as the global embedding following [Kang and McAuley, 2018]. The local embedding can be simply defined as the last clicked-item vector, *i.e.*, $\mathbf{h}_n$. Then we weight them together as the final session embedding.

$$\mathbf{S}_f = \omega\mathbf{E}_n^{(k)} + (1-\omega)\mathbf{h}_n \tag{7}$$

where $\mathbf{E}_n^{(k)} \in \mathbf{R}^d$ represent n-th row of the matrix. Finally, we predict the next click for each candidate item $v_i \in V$ given session embedding $\mathbf{S}_f$ as follows:

$$\hat{\mathbf{y}}_i = softmax(\mathbf{S}_f^T\mathbf{v}_i). \tag{8}$$

where $\hat{\mathbf{y}}_i$ denotes the recommendation probability of item v_i to be the next click in session S. Finally, we train our model by minimizing the following objective function:

$$\mathcal{J} = -\sum_{i=1}^{n}\mathbf{y}_i log(\hat{\mathbf{y}}_i) + (1-\mathbf{y}_i)log(1-\hat{\mathbf{y}}_i) + \lambda||\theta||^2. \tag{9}$$

where $\mathbf{y}$ denotes the one-hot encoding vector of the ground truth item, θ is the set of all learnable parameters.

4 Experiments and Analysis

In this section, we first set up the experiment. And then we conduct experiments to answer the following questions:

RQ1: Does the proposed graph contextualized self-attention session-based recommendation model (GC-SAN) achieve state-of-the-art performance?

RQ2: How do the key hyper-parameters affect model performance, such as the weight factor and embedding size ?

4.1 Experimental Setup

Datasets. We study the effectiveness of our proposed approach GC-SAN on two real-world datasets, *i.e.*, *Diginetica*[1] and *Retailrocket*[2]. Diginetica dataset comes from CIKM Cup 2016, where only the transactional data is used in this study. Retailrocket dataset is published by a personalized e-commerce company, which contains six months of user browsing activities. To filter noisy data, we filter out items appearing less than 5 times and then remove all sessions with fewer than 2 items on both datasets. Furthermore, for session-based recommendation, we set the sessions data of last week as the test data, and the remaining for training. Similar to [Tan *et al.*, 2016; Yuan *et al.*, 2019], for a session sequence $S = \{s_1, s_2, ..., s_n\}$, we generate the input and corresponding labels $(\{s_1\}, s_2)$, $(\{s_1, s_2\}, s_3)...(\{s_1, ..., s_{(n-1)}\}, s_n)$ for training and testing on both datasets. After preprocessing, the statistics of the datasets are shown in Table 2.

[1] http://cikm2016.cs.iupui.edu/cikm-cup/

[2] https://www.kaggle.com/retailrocket/ecommerce-dataset

Datasets	Diginetica						Retailrocket					
Measures	HR@5	HR@10	MRR@5	MRR@10	NDCG@5	NDCG@10	HR@5	HR@10	MRR@5	MRR@10	NDCG@5	NDCG@10
Pop	0.0036	0.0077	0.0019	0.0025	0.0023	0.0037	0.0133	0.0208	0.0066	0.0076	0.0082	0.0107
BPR-MF	0.1060	0.1292	0.0789	0.0842	0.0586	0.0672	0.2106	0.2719	0.1356	0.1407	0.1138	0.1322
IKNN	0.1407	0.2083	0.0776	0.0867	0.0693	0.0902	0.1709	0.2248	0.0972	0.1043	0.0855	0.1020
FPMC	0.1855	0.2309	0.0875	0.0986	0.0811	0.1037	0.1732	0.2319	0.1013	0.1152	0.0901	0.1095
GRU4Rec	0.2577	0.3657	0.1434	0.1577	0.1276	0.1607	0.2196	0.2869	0.1286	0.1489	0.1076	0.1323
STAMP	0.3998	0.5014	0.2357	0.2469	0.2039	0.2394	0.3287	0.3972	0.2241	0.2334	0.1758	0.1970
SR-GNN	0.4082	0.5269	0.2439	0.2599	0.2078	0.2443	0.3502	0.4268	0.2422	0.2525	0.1885	0.2121
GC-SAN	**0.4280**	**0.5351**	**0.2694**	**0.2838**	**0.2223**	**0.2552**	**0.3644**	**0.4380**	**0.2506**	**0.2604**	**0.1956**	**0.2181**
Improv.	4.84%	1.56%	10.46%	9.20%	6.98%	4.44%	4.07%	2.62%	3.49%	3.15%	3.79%	2.85%

Table 1: The performance of different methods on the two datasets. We generate Top-5 and 10 items for recommendation. The best performance in each column is boldfaced (the higher, the better). Improvements over the best baseline are shown in the last row.

Dataset	# clicks	# train	# test	# items	avg.len
Diginetica	858,107	526,134	44,279	40,840	5.97
Retailrocket	710,856	433,648	15,132	36,968	5.43

Table 2: Statistics of datasets.

Evaluation Metrics. To evaluate the recommendation performance of all models, we adopt three common metrics, $i.e.$, Hit Rate (HR@N), Mean Reciprocal Rank (MRR@N) and Normalized Discounted Cumulative Gain (NDCG@N). The former one is an evaluation of unranked retrieval results, while the latter two are evaluations of ranked lists. Here, we consider Top-N ($N = \{5, 10\}$) for recommendation.

4.2 Baselines

We consider the following compared methods for performance comparisons:

- **Pop** is a simple baseline that recommends top rank items based on popularity in training data.
- **BPR-MF** [Rendle *et al.*, 2009] is the state-of-the-art method for non-sequential recommendation, which optimizes matrix factorization using a pairwise ranking loss.
- **IKNN** [Sarwar *et al.*, 2001] is a traditional item-to-item model, which recommends items similar to the candidate item within the session based on cosine similarity.
- **FPMC**[3] [Rendle *et al.*, 2010] is a classic hybrid model combing matrix factorization and first-order Markov chain for next-basket recommendation. Note that in our recommendation problem, each basket is a session.
- **GRU4Rec**[4] [Hidasi *et al.*, 2016] is a RNN-based deep learning model for session-based recommendation. It utilizes a session-parallel mini-batch training process to model user action sequences.
- **STAMP** [Liu *et al.*, 2018] is a novel short-term memory priority model to capture the user's long-term preference from previous clicks and the current interest of the last-clicks in a session.
- **SR-GNN**[5] [Wu *et al.*, 2018] is recently proposed session-based recommendation model with graph neural network, which applies GNN to generate latent vectors of items and then represent each session through traditional attention network.

4.3 Comparisons of Performance

To demonstrate the recommendation performance of our model GC-SAN, we compare it with other state-of-the-art methods (**RQ1**). The experimental results of all methods on Diginetica and Retailrocket datasets are illustrated in Table1, and we have the following observations.

The non-personalized Popularity-based methods ($i.e.$, Pop) has the most unfavorable performance on both datasets. By profiling users individually and optimizing the pairwise ranking loss function, BPR-MF performs better than Pop. This suggests the importance of personalization in recommendation tasks. IKNN and FPMC achieve better performance than BPR-MF on Diginetica dataset, while BPR-MF outperforms IKNN and FPMC on Retailrocket dataset. In fact, IKNN utilizes the similarity between items in the session and FPMC is based on first-order Markov Chain.

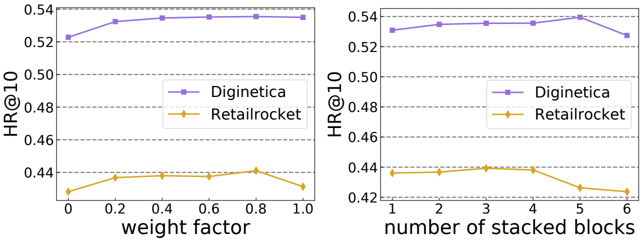
All of the neural network methods, such as GRU4Rec and STAMP, outperform the traditional baselines ($e.g.$, FPMC and IKNN) in nearly all the cases, which verifies the power of deep learning technology in this field. GRU4Rec leverages the recurrent structure with GRU as a special form of RNN to capture the user's general preference, while STAMP improves the short-term memory through the last clicked item. Unsurprisingly, STAMP performs better than GRU4Rec, which indicates the effectiveness of short-term behavior for predicting the next item problem. On the other hand, by modeling every session as a graph and applying graph neural network and the attention mechanism, SR-GNN outperforms all other baselines on both datasets. This further proves the power of neural network in recommender systems.

Compared to SR-GNN, our approach GC-SAN adopts the self-attention mechanism to adaptively assign weights to previous items regardless of their distances in the current session and captures long-range dependencies between items of a session. We combine the long-range self-attention representation and the short-term interest of the last-click in a linear way to generate the final session representation. As we can see, our method achieves the best performance among all the methods on both datasets in terms of HR, MRR, and NDCG. These results demonstrate the efficacy and validity of GC-SAN for session-based recommendation.

[3] http://github.com/khesui/FPMC

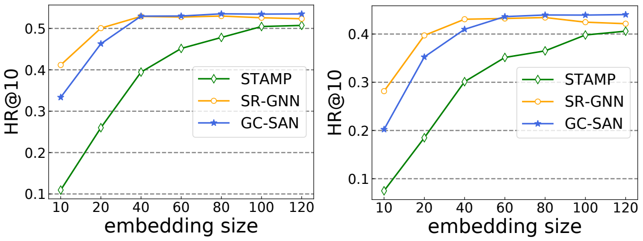
[4] http://github.com/hidasib/GRU4Rec

[5] http://github.com/CRIPAC-DIG/SR-GNN



(a) The weight factor ω (b) Number of SA blocks k

Figure 3: Effects of the weight factor ω and effects of the number of stacked self-attention blocks k on both datasets.



(a) Diginetica (b) Retailrocket

Figure 4: The performance under different embedding sizes d.

4.4 Model Analysis and Discussion

In this subsection, we take an in-depth model analysis study, aiming to further understand the framework of GC-SAN (**RQ2**). Due to the space limit, we only show the analysis results in terms of HR@10 and NDCG@10. We have obtained similar experimental results in terms of other metrics.

Dataset	Diginetica		Retailrocket	
Measures	HR@10	NDCG@10	HR@10	NDCG@10
w/GC-SAN	0.5351	0.2552	0.4380	0.2181
w/o GC-SAN	0.4199	0.2060	0.4191	0.2065

Table 3: The performance of GC-SAN with and without graph neural network in terms of HR@10 and NDCG@10.

Impact of graph neural network. Although we can infer the effectiveness of graph neural network implicitly from Table 1, we would like to verify the contribution of graph neural network in GC-SAN. We remove the graph neural network module from GC-SAN, replace it with a randomly initialized item embedding, and feed into the self-attention layer. Table 3 displays the comparisons between with and without GNN. From Table 1 and Table 3, we find that even without GNN, GC-SAN can still outperform STAMP on Retailrocket dataset, while it was beaten by GRU4Rec on Diginetica dataset. In fact, the maximum session length of Retailrocket dataset is almost four times that of Diginetica dataset. A possible reason is that short sequence lengths can construct more dense session graphs that provide richer contextual information, while the self-attention mechanism performs better with long sequence lengths. This further demonstrates that the self-attention mechanism and graph neural network play important roles in improving recommendation performance.

Impact of weight factor ω. The weight parameter ω controls the contribution of self-attention representation and the last-clicked action. Observing from Figure 3(a), taking only global self-attention dependencies ($\omega = 1.0$) as final session embedding usually achieves a better performance than considering only current interests ($\omega = 0$). Setting ω to a value from 0.4 to 0.8 is more desirable. This indicates that while the self-attention mechanism with graph neural network can adaptively assign weights to focus on long-range dependencies or more recent actions, the short-term interest is also indispensable for improving the recommendation performance.

Impact of the number of self-attention blocks k. As aforementioned, we investigate which levels of self-attention layers benefit most from GC-SAN. Figure 3(b) displays the experimental results of applying different self-attention blocks with k varying from 1 to 6. On both datasets, we can observe that increasing k can boost the performance of GC-SAN. However, it achieves the best performance when k is chosen properly and gets worse for a larger k. This may be because using more blocks ($k \geq 4$) would make GC-SAN easier to lose low-layer information.

Impact of the embedding size d. In Figure 4, we investigate the effectiveness of the embedding size d ranging from 10 to 120 on both datasets. Among all the baselines, STAMP and SR-GNN perform well and stable. Hence, we use STAMP and SR-GNN as two baselines for ease of comparisons. From figure 4, we can observe that our model GC-SAN consistently outperforms STAMP on all latent dimensions. When d is less than a certain value, SR-GNN performs better than GC-SAN. Once this value is exceeded, the performance of GC-SAN still grows and eventually stabilizes with $d \geq 100$, while the performance of SR-GNN slightly reduces. This may be because that a relatively small d limits GC-SAN to capture complex transitions between item latent factors, while SR-GNN may suffer from overfitting with a larger d.

5 Conclusion

In this paper, we proposed a graph contextualized self-attention network (GC-SAN) based on graph neural network for session-based recommendation. Specifically, we first constructed directed graphs from anonymous session records and then applied graph neural network to generate new latent vectors for all items, which contained local contextual information of sequences. Next, we used the self-attention network to capture global dependencies between distant position. Finally, we combined the local short-term dynamics (*i.e.*, the last-clicked item) and global self-attended dependencies to represent session sequences in a linear way. Extensive experimental analysis verified that our proposed model GC-SAN consistently outperformed the state-of-the-art methods.

Acknowledgments

This research was partially supported by NSFC (No. 61876117, 61876217, 61872258, 61728205), Major Project of Zhejiang Lab (No. 2019DH0ZX01), Open Program of Key Lab of IIP of CAS (No. IIP2019-1) and PAPD.

References

[Battaglia *et al.*, 2018] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[Bonnin and Jannach, 2015] Geoffray Bonnin and Dietmar Jannach. Automated generation of music playlists: Survey and experiments. *CSUR*, 47(2):26, 2015.

[He *et al.*, 2016] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast matrix factorization for online recommendation with implicit feedback. In *SIGIR*, pages 549–558. ACM, 2016.

[He *et al.*, 2018] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. NAIS: neural attentive item similarity model for recommendation. *TKDE*, 30(12):2354–2366, 2018.

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

[Huang *et al.*, 2018] Xiaowen Huang, Shengsheng Qian, Quan Fang, Jitao Sang, and Changsheng Xu. Csan: Contextual self-attention network for user sequential recommendation. In *ACM Multimedia*, pages 447–455. ACM, 2018.

[Kang and McAuley, 2018] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *ICDM*, pages 197–206. IEEE, 2018.

[Li *et al.*, 2017] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *CIKM*, pages 1419–1428. ACM, 2017.

[Li *et al.*, 2018] Zhongyang Li, Xiao Ding, and Ting Liu. Constructing narrative event evolutionary graph for script event prediction. *arXiv preprint arXiv:1805.05081*, 2018.

[Li *et al.*, 2019] Xiangpeng Li, Jingkuan Song, Lianli Gao, Xianglong Liu, Wenbing Huang, Xiangnan He, and Chuang Gan. Beyond rnns: Positional self-attention with co-attention for video question answering. In *AAAI*, 2019.

[Lin *et al.*, 2017] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *ICLR*, 2017.

[Liu *et al.*, 2018] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. Stamp: short-term attention/memory priority model for session-based recommendation. In *KDD*, pages 1831–1839. ACM, 2018.

[Liu *et al.*, 2019] Pengfei Liu, Shuaichen Chang, Xuanjing Huang, Jian Tang, and Jackie Chi Kit Cheung. Contextualized non-local neural networks for sequence learning. *AAAI*, 2019.

[Marino *et al.*, 2017] Kenneth Marino, Ruslan Salakhutdinov, and Abhinav Gupta. The more you know: Using knowledge graphs for image classification. *CVPR*, pages 20–28, 2017.

[Rendle *et al.*, 2009] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461. AUAI Press, 2009.

[Rendle *et al.*, 2010] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820. ACM, 2010.

[Sarwar *et al.*, 2001] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295. ACM, 2001.

[Scarselli *et al.*, 2009] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[Tan *et al.*, 2016] Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. In *RecSys*, pages 17–22. ACM, 2016.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[Wang *et al.*, 2019] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *SIGIR*, 2019.

[Wu *et al.*, 2018] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. *AAAI*, 2018.

[Yang *et al.*, 2018] Baosong Yang, Zhaopeng Tu, Derek F Wong, Fandong Meng, Lidia S Chao, and Tong Zhang. Modeling localness for self-attention networks. *EMNLP*, 2018.

[Yuan *et al.*, 2019] Fajie Yuan, Alexandros Karatzoglou andIoannis Arapakis, Joemon M. Jose, and Xiangnan He. A simple convolutional generative network for next item recommendation. In *WSDM*, pages 582–590, 2019.

[Zhao *et al.*, 2019] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzhen Zhuang, Victor S. Sheng, and Xiaofang Zhou. Where to go next: A spatio-temporal gated network for next poi recommendation. In *AAAI*, 2019.

[Zhou *et al.*, 2018] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. Atrank: An attention-based user behavior modeling framework for recommendation. In *AAAI*, 2018.