Collaborative Self-Attention Network for Session-based Recommendation

Anjing Luo[1] , **Pengpeng Zhao**[1*] , **Yanchi Liu**[2] , **Fuzhen Zhuang**[3,4] ,
Deqing Wang[5] , **Jiajie Xu**[1] , **Junhua Fang**[1] and **Victor S. Sheng**[6]
[1]Institute of AI, School of Computer Science and Technology, Soochow University, China
[2]Rutgers University, New Jersey, USA
[3]Key Lab of IIP of CAS, Institute of Computing Technology, Beijing, China
[4]The University of Chinese Academy of Sciences, Beijing, China
[5]School of Computer, Beihang University, Beijing, China
[6]Texas Tech University, Texas, USA
ppzhao@suda.edu.cn

Abstract

Session-based recommendation becomes a research hotspot for its ability to make recommendations for anonymous users. However, existing session-based methods have the following limitations: (1) They either lack the capability to learn complex dependencies or focus mostly on the current session without explicitly considering collaborative information. (2) They assume that the representation of an item is static and fixed for all users at each time step. We argue that even the same item can be represented differently for different users at the same time step. To this end, we propose a novel solution, Collaborative Self-Attention Network (CoSAN) for session-based recommendation, to learn the session representation and predict the intent of the current session by investigating neighborhood sessions. Specially, we first devise a collaborative item representation by aggregating the embedding of neighborhood sessions retrieved according to each item in the current session. Then, we apply self-attention to learn long-range dependencies between collaborative items and generate collaborative session representation. Finally, each session is represented by concatenating the collaborative session representation and the embedding of the current session. Extensive experiments on two real-world datasets show that CoSAN constantly outperforms state-of-the-art methods.

1 Introduction

In the era of rapid change of information, the primary purpose of the recommender system is to provide users with the required information in a timely and effective manner. However, the user's identity is unknown in many scenarios. For example, the user is not logged in when browsing an online shop. In this scenario, only the limited user interaction records in the current session can be used to predict the user's next click. To improve the recommendation results for anonymous users, the session-based recommendation has attracted a lot of attentions.

Early works on session-based recommendation focused on discovering item-to-item relations, like transition relation and co-occurrence relation. Typical methods such as ItemKNN [Linden *et al.*, 2003; Sarwar *et al.*, 2001] and Markov Chain [Garcin *et al.*, 2013; He *et al.*, 2009] relied on the last element in the session to generate recommendations. However, relying solely on the last element of the session cannot reflect users' interests throughout the session. Then, session-based KNN [Hariri *et al.*, 2012; Bonnin and Jannach, 2015; Lerche *et al.*, 2016] was proposed to compare the entire current session with the past sessions in the training data to determine which item to recommend. However, neighbor-based approaches lack the ability to learn complex dependencies and sequential signals within the current session.

Recent studies model a series of clicks in the session as a sequence and utilize neural networks to model the whole action sequence. For example, GRU4Rec [Hidasi *et al.*, 2016] applied recurrent neural networks (RNN) for session-based recommendation and treated this problem as time series prediction. With deep learning making massive strides in various research areas, more and more deep learning models improve simple RNN-based models by considering the main purpose of the session [Li *et al.*, 2017], capturing user's general preferences and current interests [Liu *et al.*, 2018]. Other models like SR-GNN [Wu *et al.*, 2019] modeled the sessions as a graph to capture complex item interactions. However, RNNs are notoriously tricky to train [Pascanu *et al.*, 2013] because of the gradient vanishing and exploding problem. Various variants like LSTM and GRU alleviate the above problems but still struggle to capture long-term dependencies.

More recently, a new sequential model named Transformer [Vaswani *et al.*, 2017] has achieved promising performance and efficiency in recommendation tasks. Different from RNN-based methods, Transformer allows the model to access any part of the history regardless of distance, making it potentially more suitable for grasping recurring patterns with long-term dependencies. For instance, SASRec [Kang and McAuley, 2018] modeled the entire user sequence through

[*] Pengpeng Zhao is the corresponding author.

a simple and paralleled self-attention mechanism, and adaptively considered consumed items for prediction. Nevertheless, these model-based methods focus mostly on the current session without explicitly considering collaborative information which may help improve recommendation performance for the current session. Moreover, the item representation of these models is relatively static and fixed for all users. Still, we argue that the representations of items should be different for different users even at the same time step.

To resolve these issues, in this paper, we propose a collaborative self-attention network. Firstly, we construct a collaborative item representation with two steps. The first step is to find the set of neighborhood sessions of the current session. For each item in the current session, we calculate the similarities between the current session and the M recent sessions which also interact with this item, then select the K most similar sessions from M recent sessions as neighborhood sessions of the current session. In the second step, we perform weighted summation on neighborhood sessions to obtain the complementary feature embedding, where the weight of neighborhood session is defined as its similarity with the current session. Then, the complementary feature embedding is merged with the original item embedding to generate collaborative item representation. Secondly, we learn long-range dependencies between collaborative items and generate collaborative session representation. Finally, we concatenate the collaborative session representation and the session embedding to predict the probability of clicking on the next item. The contributions of our work are summarized as follows.

- We propose the collaborative self-attention network (CoSAN) to learn the session representation and predict the intent of the current session by investigating neighborhood sessions and modeling the long-range dependencies between collaborative items.
- We design a collaborative item representation method through injecting complementary feature embedding represented by neighborhood sessions into the item embedding. It not only explicitly utilizes the collaborative information in neighborhood sessions, but also constructs the dynamic item representation.
- We compare our model CoSAN with state-of-the-art methods and verify the superiority of CoSAN through quantitative analysis on two real-world datasets.

2 Related Work

Since our collaborative self-attention network (CoSAN) is proposed for session-based recommendation with self-attention network, we survey related work from two areas: session-based recommendation and self-attention network.

2.1 Session-based Recommendation

Session-based recommendation makes use of implicit feedbacks in the current session instead of explicit preferences (e.g., ratings) to make recommendations for anonymous users. Therefore, the model-based methods are not suitable for the session-based recommendation when lacking user profiles. In this scenario, it is natural to employ the item-to-item recommendation approaches to solve this task. [Linden *et al.*, 2003] proposed an item-to-item collaborative filtering method to compute the similarity between items based on their co-occurrence frequency. [Sarwar *et al.*, 2001] analyzed different item-based recommendation generation techniques and compared their results with basic k-nearest neighbor approaches. In the session-based setting, [Hariri *et al.*, 2012; Bonnin and Jannach, 2015; Lerche *et al.*, 2016] compared the entire session with previous sessions in the training set and decided which item to recommend. Though these methods are proved effective in the session-based recommendation, they either ignore the global information of the whole click sequence, or cannot learn complex dependencies and sequential signals within the current session.

Recently, neural networks and attention-based models are popular in the session-based recommender system. GRU4Rec [Hidasi *et al.*, 2016], which employed session-parallel mini-batched for training, first introduced the GRU to the session-based recommendation. Later, an improved version [Tan *et al.*, 2016] was proposed to boost the recommendation performance further. NARM [Li *et al.*, 2017] proposed to model the user's sequential behavior and capture the user's main purpose of the current session by applying hybrid encoding with the attention mechanism. STAMP [Liu *et al.*, 2018] captured user's general and current interests by applying MLP networks and an attentive net. SR-GNN [Wu *et al.*, 2019] modeled the sessions as a graph structure to capture complex item interactions while the user's global preferences and current interests were combined through an attention mechanism. Nowadays, WH [Jannach and Ludewig, 2017] showed nearest-neighbor methods should be considered as competitive baselines for session-based recommendation scenarios, and combining GRU4REC with the KNN methods in a weighted hybrid approach led to a better result. CSRM [Wang *et al.*, 2019] hybridized the inner and outer memory encoder to model the preferences of the current and neighborhood sessions respectively.

2.2 Self-Attention Network

Since Transformer [Vaswani *et al.*, 2017] showed its excellent performance on machine translation tasks, self-attention mechanism has been widely used to model the sequential data and achieved remarkable results in the recommender system. SASRec [Kang and McAuley, 2018] modeled the entire user sequence through the simple and parallelized self-attention mechanism, and adaptively considered consumed items for prediction. CSAN [Huang *et al.*, 2018] proposed a unified contextual self-attention network at the feature level to capture the polysemy of heterogeneous user behaviors for sequential recommendation. FDSA [Zhang *et al.*, 2019] modeled the transition patterns between items and features through an item-based and a feature-based self-attention block, respectively. GC-SAN [Xu *et al.*, 2019] utilized the complementarity between self-attention network and graph neural network to enhance the recommendation performance.

A recently proposed CSRM [Wang *et al.*, 2019], which is closely related to our work, incorporates two parallel memory modules to consider current session information and collaborative neighborhood information, respectively. The differences between our work CoSAN and CSRM are threefolds. First, the item representation of CoSAN is dynamic. In CoSAN, a collaborative item representation is proposed, which is capable to dynamically generate different representations when encountering different users and time steps, while the item representation of CSRM is relatively static and fixed for all users. Second, when retrieving neighborhood sessions, CoSAN retrieves K most similar neighborhood sessions for each item in the current session. In contrast, CSRM retrieves K most similar neighborhood sessions according to the local encoder which reflects the main intent of the current session. Therefore, CoSAN contains more diverse collaborative information than CSRM. Third, CoSAN employs the self-attention network to capture the long-range dependencies between collaborative items regardless of distance. Meanwhile, CSRM models a user's information in the current session with the help of RNNs and an attention mechanism, which is insufficient to capture long-term dependencies.

3 Collaborative Self-Attention Network

In this section, we first formulate the problem of session-based recommendation, and then elaborate on our proposed collaborative self-attention network (As shown in Figure 1).

3.1 Problem Statement

The task of session-based recommendation is to predict which item the user will click next based on the current session. Let $V = \{v_1, v_2, ..., v_{|V|}\}$ denote a set of all unique items involved in all sessions while $S = \{s_1, s_2, ..., s_{|S|}\}$ denotes a set of sessions, where $|V|$ and $|S|$ are the total number of unique items and sessions, respectively. For an anonymous session, a sequence of n clicked actions by the user is denoted as $s_i = \{x_1, x_2, ..., x_n\}$ in the time order, where $x_t \in V$ represents a clicked item of the user at time step t. The goal of session-based recommendation is to predict the next click (i.e., x_{n+1}) for session s_i. Formally, our model aims to generate a ranked list of all candidate items by predicting their click probability. The scores of all candidate items are denoted by $\hat{y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_{|V|}\}$, where $\hat{y}_i$ refers to the score of item v_i. The prediction scores are ranked in the descending order, and the items ranked in the top-k are used for recommendation.

3.2 Collaborative Item Representation

Collaborative item representation is a dynamic item representation with collaborative information by explicitly considering other sessions that also interacted with the item in the current session. The process of constructing the collaborative item representation can be divided into searching for neighborhood sessions of the current session and constructing collaborative item representation. To be specific, the embedding of session s_i and item x_t are defined as e_{s_i} and e_{x_t} according to session id and item id, respectively. Let $N_i = \{N_{i,1}, N_{i,2}, ..., N_{i,n}\}$ represent the set of the embedding of neighborhood sessions for the whole session s_i, where $N_{i,t}$ is the set of the embedding of neighborhood sessions that interact with x_t in s_i.

Search for Neighborhood Sessions. The process of searching for neighborhood sessions is as follows. Given the current session s_i and the item x_t which is interacted by the current session at time step t, we first take the recent M sessions which also interact with x_t as candidate neighborhood sessions. To further obtain the most similar K sessions, we calculate the similarity [Bonnin and Jannach, 2015] between current session s_i and candidate neighborhood session $m_j \in M$ at time step t as follows:

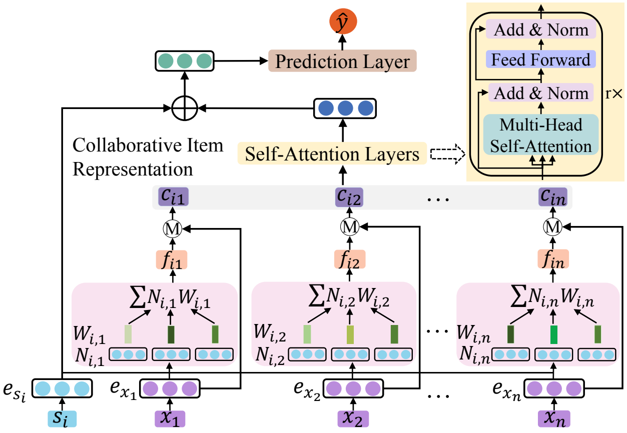


Figure 1: The architecture of our proposed CoSAN.

$$sim(s_{it}, m_j) = \frac{|s_{it} \cap m_j|}{\sqrt{|s_{it}| \cdot |m_j|}} \tag{1}$$

where s_{it} is the set of the first t items in the current session s_i whose last element is x_t, and m_j represents all items in the j-th session in M. According to the K largest similarity scores, we take the corresponding sessions as the final neighborhood sessions, and the similarity scores are regarded as the weights of these neighborhood sessions.

Construct Collaborative Item Representation. After obtaining the K neighborhood sessions and their weights, the second step is to construct the collaborative item representation. As shown in Figure 1, with the embedding of K neighborhood sessions $N_{i,t} = \{n_{it}^1, n_{it}^2, ..., n_{it}^K\}$ and their similarities $W_{i,t} = \{w_{it}^1, w_{it}^2, ..., w_{it}^K\}$, the complementary feature embedding of s_i according to x_t is formed by a weighted sum of $N_{i,t}$ as follows:

$$f_{it} = \sum_{k=1}^{K} n_{it}^k w_{it}^k \tag{2}$$

where n_{it}^k is the embedding of the k-th neighborhood session of s_i at time step t, which is generated according to session id. Finally, the item embedding and complementary feature embedding are merged together to get the collaborative item representation which is a session-related embedding for item x_t, defined by:

$$c_{it} = merge(e_{x_t}, f_{it}) \tag{3}$$

where $merge(.)$ is a function that combines two vectors into one. The particular choice of $merge(.)$ in our model is a simple weighted vector addition as follows:

$$merge(x, y) = x + \alpha y \tag{4}$$

where α is a weighting parameter to indicate the importance of variable y. In our model, α is used to measure the importance of the complementary feature embedding f_{it}.

3.3 Self-Attention Layers

Self-attention, an attention mechanism relating to different positions of a single sequence to compute a representation of the sequence, has been successfully employed in various fields [Cheng *et al.*, 2016; Parikh *et al.*, 2016; Paulus *et al.*, 2017; Lin *et al.*, 2017]. To learn global dependencies between the collaborative items and generate the collaborative session representation, we employ the self-attention layers composed of self-attention blocks, multi-head self-attention, feed-forward network, and multi-layer self-attention.

Self-Attention Blocks. An attention function can be described as a mapping query and a set of key-value pairs to an output, where the queries, keys, values are the set of collaborative item representations $C = \{c_{i1}, c_{i2}, ..., c_{in}\}$ for session s_i in our case while the output is the collaborative session representation which can reflect the intent of the current session.

$$H = softmax(\frac{(CW^Q)(CW^K)^T}{\sqrt{d}})(CW^V) \tag{5}$$

where the projection matrices W^Q, W^K, $W^V \in \mathbb{R}^{d \times d}$ and $\sqrt{d}$ is the scale factor which is used to avoid overly large values of the inner product, especially when the d is high. d is the latent dimensionality.

Multi-Head Self-Attention. To enable the model to jointly attend to the information from different representation subspaces at the different positions, we adopt multi-head attention employing h separate attention models with distinct parameters in parallel. The output of all attention models are concatenated to generate final values.

$$\begin{aligned} O &= Concat(H_1, H_2, ..., H_h) \\ H_i &= softmax(\frac{(CW_i^Q)(CW_i^K)^T}{\sqrt{d}})(CW_i^V) \end{aligned} \tag{6}$$

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

Feed-Forward Network. In order to overcome the shortcomings of self-attention being a linear model, we use a point-wise feedback network with the ReLU activation function to endow the model with nonlinearity and consider interactions between different latent dimensions. Then we use residual connection to make full use of low-layer information.

$$F = ReLU(OW_1 + b_1)W_2 + b_2 + O \tag{7}$$

where W_1, W_2 are $d \times d$ matrices and $b_1, b_2 \in \mathbb{R}^d$ are bias vectors. We also employ normalization to normalize the inputs across features while dropout is used to avoid overfitting.

Multi-Layer Self-Attention. To learn more complex item transitions, we stack the self-attention block to construct multi-layer self-attention. We define the above whole self-attention network for simplicity as follows:

$$F = SAN(C) \tag{8}$$

Then, the r-th (r>1) self-attention layer is defined as:

$$F^{(r)} = SAN(F^{(r-1)}) \tag{9}$$

where $F^{(1)} = F$ and $F^{(r)}$ is the final output of the multi-layer self-attention network.

Dataset	Retailrocket	Yoochoose 1/64	Yoochoose 1/4
clicks	1,085,217	557,248	8,326,407
train	455,327	369,859	5,917,746
test	16,240	55,898	55,898
items	48,989	16,766	29,618
avg.len	3.54	6.16	5.71

Table 1: Statistics of the datasets.

3.4 Prediction Layer

To predict the next click of the session s_i, we get the final session embedding by concatenating the last dimension of $F^{(r)}$ [Kang and McAuley, 2018] and session embedding e_{s_i}:

$$S_f = concat(F_n^{(r)}, e_{s_i}) \tag{10}$$

where $F_n^{(r)}$ represents the n-th row of the matrix. Then, the score of each candidate item $v_p \in V$ is:

$$\hat{y}_p = (S_f)^T v_p \tag{11}$$

where $\hat{y}_p$ denotes the recommendation probability of the item v_p to be the next click of the session s_i. Finally, we adopt the binary cross-entropy as the optimization objective function:

$$L = -\sum_{\substack{v_p \in s_i \\ v_q \notin s_i}} log(\sigma(\hat{y}_p)) + log(1 - \sigma(\hat{y}_q)) + \lambda\|\theta\|^2 \tag{12}$$

where σ is the sigmoid function. θ is the set of all learnable parameters and λ represents the regularization term. Moreover, for each target item v_p of the session s_i, we randomly sample a negative item v_q.

4 Experiments

In this section, we first set up the experiment. And then, we compare our model with the start-of-the-art baselines and analyze the results. Finally, we explore the role of components (e.g., neighborhood sessions, self-attention network) of the model and the influence of hyper-parameters (e.g., the latent dimensionality and the number of neighborhood sessions).

4.1 Experimental Setup

Datasets

We study the effectiveness of our proposed model CoSAN on two real-world datasets, i.e., Retailrocket[1] and Yoochoose[2].

- Retailrocket is the users' click stream data published by a personalized e-commerce company that contains six months of user browsing activities. In our experiments, we first manually partition the user's history into sessions in a 30-minute interval, then filter out sessions of length 1 and items that appear less than 5 times.
- Yoochoose which contains click-streams on an e-commerce site is a public dataset released by RecSys Challenge 2015. After filtering out sessions of length 1 and items that appear less than 5 times, there remain 7,981,581 sessions and 37,483 items.

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

[2] http://2015.recsyschallenge.com/challege.html

Datasets	Retailrocket			Yoochoose 1/64			Yoochoose 1/4		
Measures	HR@5	MRR@5	NDCG@5	HR@5	MRR@5	NDCG@5	HR@5	MRR@5	NDCG@5
BPR-MF	0.3292	0.2904	0.3003	0.2519	0.1780	0.1966	0.2083	0.1615	0.1733
FPMC	0.3055	0.2699	0.2789	0.2904	0.1917	0.2162	0.2660	0.1672	0.1917
IKNN	0.1762	0.1042	0.1221	0.3282	0.1997	0.2315	0.3206	0.1957	0.2267
SKNN	0.4606	0.3276	0.3608	0.3944	0.2268	0.2684	0.3936	0.2266	0.2680
GRU4Rec	0.3390	0.2433	0.2674	0.3621	0.2270	0.2610	0.3951	0.2451	0.2717
STAMP	0.4639	0.2776	0.3238	0.4637	0.2784	0.3244	0.4693	0.2804	0.3273
NARM	0.4516	0.3241	0.3559	0.4677	0.2803	0.3269	0.4760	0.2824	0.3305
SASRec	0.4943	0.3929	0.4166	0.4376	0.2926	0.3264	0.4462	0.2922	0.3306
CSRM	0.4520	0.3257	0.3572	0.4707	0.2815	0.3285	0.4783	0.2846	0.3328
CoSAN	**0.5399**	**0.4139**	**0.4453**	**0.4801**	**0.3093**	**0.3517**	**0.4894**	**0.3105**	**0.3525**
Improv.	9.22%	5.34%	6.90%	2.00%	5.69%	7.06%	2.32%	6.25%	5.92%

Table 2: The performance of different methods on the two datasets. We generate the Top-5 items for recommendation. Boldface indicates the best results (the higher, the better), while the second best is underlined.

We take the sessions of the subsequent day on Yoochoose and the sessions of the subsequent week on Retailrocket for testing. Since Yoochoose is quite large, we sorted the training sequences by time and reported our results on more recent fractions 1/64 and 1/4 of the training sequences [Li *et al.*, 2017]. Note that the items in the test set should be included in the training set. After preprocessing, the statistics of the datasets are shown in Table 1. For the session $s_i = \{x_1, x_2, ..., x_n\}$, the input of our model is $\{x_1, x_2, ..., x_{n-1}\}$ and its expected output is a 'shifted' version of the same session $\{x_2, x_3, ..., x_n\}$ during the training process. In the test process, we take the last item as the ground truth and the remaining as the input to generate the session representation. It is different from previous session-based methods [Li *et al.*, 2017; Liu *et al.*, 2018] because the self-attention network performs better in the task of sequence to sequence than the sequence to item. Furthermore, we add zero-padding to the left side of the clicks in the session if the session is shorter than the fixed-length l. Otherwise, we take the most recent l clicks.

Evaluation Metrics and Implementation Details. To evaluate the performance of all models, we adopt Hit Rate (HR@N), Mean Reciprocal Rank (MRR@N) and Normalized Discounted Cumulative Gain (NDCG@N) as metrics. The former one is an evaluation of unranked retrieval results while the latter two are evaluations of ranked lists. Here, we consider Top-N ($N = 5$) for recommendation. Without a special mention, we set the number of self-attention heads h and self-attention layers r to 1 and 2 respectively. Also, the weighting parameter α is set to 0.5.

4.2 Baselines

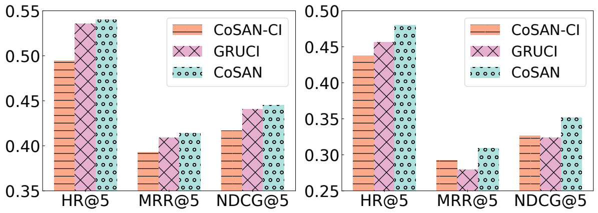
We compare our model with the following methods.

- **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.
- **FPMC** [Rendle *et al.*, 2010] combines a Markov chain model and matrix factorization for the next basket recommendation. Note that in our recommendation problem, each basket is a session.
- **IKNN** recommends the most similar K items according to the last item in the current session based on cosine similarity.
- **SKNN** computes the scores of candidate items according to their occurrences in the neighborhood sessions when predicting the next item for the current session.
- **GRU4Rec** [Hidasi *et al.*, 2016] is an 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 short-term memory priority model which captures the user's long-term preference from previous clicks and the current interest of the last clicks in a session.
- **NARM** [Li *et al.*, 2017] employs RNNs with attention mechanisms to capture a user's main purpose and sequential behaviors, which are treated as equally critical complementary features.
- **SASRec** [Kang and McAuley, 2018] is a self-attention based sequential model which can consider consumed items for next item recommendation. Here, we view a session as a sequence.
- **CSRM** [Wang *et al.*, 2019] is a deep learning model that consists of an inner and an outer memory encoder to model the preference of the current session and the neighborhood sessions.

4.3 Performance Comparisons

Table 2 illustrates the experimental results of all methods on both datasets, and we have the following observations.

Among traditional methods, SKNN performs the best because it takes advantage of all clicks in the session and considers collaborative information. BPR-MF and FPMC perform stably on different datasets, but IKNN has a large floating gap. This may be because the first two models take into account the general preferences of the user, while IKNN only considers the similar items of the last element in the session. The last item cannot represent the main intention of the current session stably in different situations. Besides, FPMC with sequential information on the Yoochoose dataset performs better than BPR-MF. This shows that the sequential pattern plays a positive role in the recommendation. But at the same time, we find that BPR-MF performs better than FPMC on the Retailrocket dataset, and SKNN performs equally well



(a) Retailrocket (b) Yoochoose 1/64

Figure 2: Effectiveness of neighborhood sessions and self-attention networks on Retailrocket and Yoochoose 1/64.

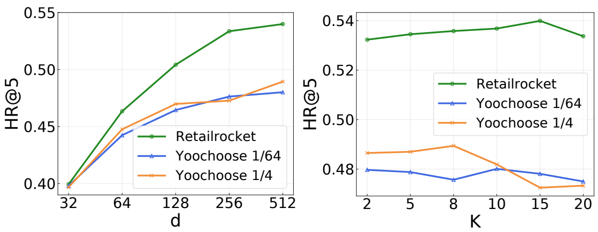
or significantly better than deep learning methods. These two phenomena suggest that sequential patterns in the session may not be as important as we think in some cases.

The deep learning approaches (GRU4Rec, STAMP, NARM, SASRec and CSRM) outperform the most traditional methods, indicating the advantages of deep learning methods in dealing with sequential information in sessions. STAMP and NARM perform better than GRU4REC, confirming the effectiveness of capturing short-term memory and the main intent in a session to improve recommendation performance. Besides, SASRec outperforms all other baselines on the Retailrocket dataset by learning the long-term dependencies between items in the session regardless of distance, and CSRM performs the best among the baselines on the Yoochoose dataset for considering the collaborative information.

Our proposed method CoSAN consistently outperforms other competitive methods in terms of three evaluation metrics on both datasets. Compared with CSRM, the better results of CoSAN mean explicitly finding neighborhood sessions for each item in the current session instead of retrieving the neighborhood sessions based on the main intent of the current session is useful, because the latter model may contain a lot of noise from irrelevant sessions and the collaborative information it utilizes is lack of diversity. Moreover, CoSAN employs the self-attention network to capture long-range dependencies between collaborative items through adaptively assigning weights to previous collaborative items regardless of their distances in the current session, while CSRM applies GRU to express user preferences. This indicates the effectiveness of self-attention network for modeling long-term dependencies and learning session representation. Furthermore, the improvement on Retailrocket is larger than that on Yoochoose. This probably because the neighborhood sessions may play a greater role when the session is shorter.

4.4 Influence of Components

To further illustrate the effect of collaborative information and self-attention networks, we compare the performance of CoSAN and two variants of CoSAN on Retailrocket and Yoochoose 1/64. CoSAN-CI refers to CoSAN without the collaborative item representation which actually amounts to SASRec, and GRUCI refers to the model which replaces the self-attention layers in CoSAN with GRU. In Figure 2, we first find that CoSAN outperforms CoSAN-CI. This proves that focusing mostly on considering the sequential characteristics



(a) Latent dimensionality (b) Number of neighbors

Figure 3: Effectiveness of the latent dimensionality d and the number of neighborhood sessions K on all datatsets.

is insufficient to predict the next-item. It is essential to involve collaborative information since similar sessions tend to click on similar items. And the collaborative item representation in CoSAN could convert the item embedding to session-related item embedding which is well-suited to the personalized recommendation. Second, CoSAN achieves a better performance than GRUCI. This shows the effectiveness of self-attention networks for session-based recommendation rather than GRU. Finally, since self-attention networks are superior to GRU in modeling session preferences, GRUCI outperforms CoSAN-CI on Retailrocket, which further demonstrates the vital role of collaborative item representation.

4.5 Influence of Hyper-Parameters

We evaluate the influence of varying the latent dimensionality d and the number of neighborhood sessions K only in terms of HR@5 on all datasets, due to the space constraint. As shown in Figure 3, increasing latent dimensionality can improve our model, and the proper number of neighborhood sessions is important. The latent dimensionality determines the complexity of the model, and the larger latent dimensionality may fit the model better. Moreover, fewer neighborhood sessions are insufficient to provide enough collaborative information for the current session, while more neighborhood sessions bring the noise.

5 Conclusion

In this paper, we proposed a novel method named Collaborative Self-Attention Network (CoSAN) for Session-based Recommendation. Specifically, we design a collaborative item representation to learn a dynamic item representation by aggregating the embedding of neighborhood sessions which are similar to the current session, and then learn the collaborative session representation and model the long-range dependencies between collaborative items with a self-attention network. Our model considers not only the preference of current and neighborhood sessions but also the dynamics of item representation. Our experimental results showed that CoSAN outperforms the state-of-the-art methods on two real-world datasets in terms of HR, MRR, and NDCG.

Acknowledgments

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

References

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

[Cheng *et al.*, 2016] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[Garcin *et al.*, 2013] Florent Garcin, Christos Dimitrakakis, and Boi Faltings. Personalized news recommendation with context trees. pages 105–112, 2013.

[Hariri *et al.*, 2012] Negar Hariri, Bamshad Mobasher, and Robin Burke. Context-aware music recommendation based on latenttopic sequential patterns. In *RecSys*, pages 131–138. ACM, 2012.

[He *et al.*, 2009] Qi He, Daxin Jiang, Zhen Liao, Steven CH Hoi, Kuiyu Chang, Ee-Peng Lim, and Hang Li. Web query recommendation via sequential query prediction. In *ICDE*, pages 1443–1454. IEEE, 2009.

[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 *MM*, pages 447–455. ACM, 2018.

[Jannach and Ludewig, 2017] Dietmar Jannach and Malte Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *RecSys*, pages 306–310. ACM, 2017.

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

[Lerche *et al.*, 2016] Lukas Lerche, Dietmar Jannach, and Malte Ludewig. On the value of reminders within e-commerce recommendations. In *UMAP*, pages 27–35. ACM, 2016.

[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.

[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. *ICLR*, 2017.

[Linden *et al.*, 2003] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, (1):76–80, 2003.

[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 *SIGKDD*, pages 1831–1839. ACM, 2018.

[Parikh *et al.*, 2016] Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

[Pascanu *et al.*, 2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *ICML*, pages 1310–1318, 2013.

[Paulus *et al.*, 2017] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 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 Munir Sarwar, George Karypis, Joseph A Konstan, John Riedl, et al. Item-based collaborative filtering recommendation algorithms. *WWW*, 1:285–295, 2001.

[Tan *et al.*, 2016] Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. In *DLRS*, 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] Meirui Wang, Pengjie Ren, Lei Mei, Zhumin Chen, Jun Ma, and Maarten de Rijke. A collaborative session-based recommendation approach with parallel memory modules. In *SIGIR*, pages 345–354, 2019.

[Wu *et al.*, 2019] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *AAAI*, volume 33, pages 346–353, 2019.

[Xu *et al.*, 2019] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. pages 3940–3946, 2019.

[Zhang *et al.*, 2019] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. Feature-level deeper self-attention network for sequential recommendation. In *IJCAI*, pages 4320–4326. AAAI Press, 2019.