Learning Sequential General Pattern and Dependency via Hybrid Neural Model for Session-Based Recommendation

QUAN LI, XINHUA XU, JINJUN LIU, AND GUANGMIN LI
Department of Computer and Information Engineering, Hubei Normal University, Huangshi, Hubei 435002, China
Corresponding author: Quan Li (cleverlq@hotmail.com)

This work was supported in part by the Science and Technology Research Project of Hubei Department of Education under Grant D20212503, and in part by the Research Project of Young Teachers in Hubei Normal University under Grant HS2020QN029.

ABSTRACT Recent study shows that recommendation system not only relies on user’s static preference, but also dynamic preference. Consequently, it leads to the emergence of session-based recommendation. With the development of recurrent neural network, this kind of method can capture representations of users’ sequential behaviors from a large number of sessions. However, it is prone to spurious dependency problem. Recently, convolutional neural network has also shown its potential in modelling session, especially in extracting complex local pattern of subsequence. Therefore, we propose a hybrid neural model, called SGPD, for learning sequential general pattern and dependency for session-based recommendation. In SGPD, we propose recurrent residual convolution network to extract general pattern of subsequence in a session. Furthermore, the SGPD scans sequence from forward and reverse direction by bidirectional recurrent neural network, and learns sequential dependency of a session. Finally, the objective function is constructed by cross entropy and the model parameters are learned. The experimental results show that the precision rate, recall rate and mean reciprocal ranking of SGPD are greatly improved compared with the state-of-art methods. It has good application prospect.

INDEX TERMS Session-based recommendation, recurrent residual convolution, bidirectional recurrent neural network, cross entropy.

I. INTRODUCTION

With the development of big data and cloud computing, the number of texts, images, and videos of Internet increases explosively. It is very difficult for users to find their favorite items in huge amounts of data. In order to solve data overload problems, the recommender system can help the users to find interesting items according to their preference, such as movies, news, or products. Traditional recommender system generally employs collaborative filtering method. However, it only learns user’s static preference with the assumption that all user-item interactions in the historical actions are equally important. In many real-life applications, the next item recommended to the user depends not only on his/her static preference, but also on dynamic preference, which may be influenced by the recent interactions with taking the sequential dependency into account. Therefore, session-based recommendation technique has emerged. For example, e-commerce platform can accurately recommend the next item according to the purchase sequence of user, so as to mine user’s requirement and interest for merchant [1]. The location social platform can accurately predict the next location, where the user will go according to the point-of-interest (POI) sequence, so as to improve user’s life experience [2].

Traditional session-based recommendation model generally employs matrix factorization. The method decomposes the users’ interaction data, extracts the latent feature matrix of user-item, and makes the recommendation [3], [4], [5]. However, many important sequential information cannot be well exploited for sessions. Current solutions generally employ deep learning technique to learn more representative latent feature for session-based recommendation. Among many
neural network models, the kind of recurrent neural network (RNN) [6], which is designed to learn feature from a session, seems to be a straightforward candidate. Any adjacent items in a session are highly dependent for RNN, hence it will generate spurious dependency problem. Recently, The convolutional neural network (CNN) [7] has also found its application in this important task. The main advantage of CNN is the ability to extract locally structured pattern that is important for recommendation. Furthermore, as a CNN model is trained by taking sequential input, it may happen that session 1 and session 2 with the different subsequences but their same orders result in same target, and session 1 and session 3 with the same subsequences but their different orders result in different target. We use an example illustrated by Fig. 1 to explain the motivation of our work.

In Fig. 1, sub 1, sub 2, sub 3 are a small set of toiletries, pants, and t-shirts, respectively. The target 1 is still a t-shirt, and target 2 may be a toothbrush. The session 1 and session 2 are with different subsequential patterns. Because both sub 1 and sub 4 contain the same set of items yet with different orders. The sequence \( \text{sub 1} \rightarrow \text{sub 2} \rightarrow \text{sub 3} \) and \( \text{sub 4} \rightarrow \text{sub 2} \rightarrow \text{sub 3} \) may be treated differently by a CNN model. However, the target item is the same. Meanwhile, the session 1 and session 3 are with same subsequential patterns. The sequence \( \text{sub 1} \rightarrow \text{sub 2} \rightarrow \text{sub 3} \) and \( \text{sub 2} \rightarrow \text{sub 3} \rightarrow \text{sub 1} \) may be treated the same by a CNN model, because it is designed in a local manner to focus on the structured pattern, and ignore the order of subsequence. However, the target item is different.

While both RNN and CNN show their certain advantages in modelling a session, combining two kinds of networks in a hybrid model can leverage the advantage of each approach, and produce more accurate recommendation. In this paper, we propose a hybrid neural model, called SGPD, for learning sequential general pattern and dependency for session-based recommendation. The proposed model can further improve the performance of recommendation. The main contributions of the paper are as follows:

1. Firstly, we design a sliding window, and consecutive items within the window generate item block in a session. Secondly, we propose recurrent residual convolution (RecResCon) network to learn general pattern of subsequence in the interaction session. Finally, the sliding window is moved from left to right to learn general patterns of different subsequences.

2. In order to learn sequential dependency feature, we design bidirectional gated recurrent unit (Bi-GRU). We leverage Bi-GRU to learn the user’s preference representation bidirectionally from the sequence, and finally predict the score for the user accessing to the next item.

3. We conduct experiments on four real-world datasets, the experimental results demonstrate that the proposed method is better than other state-of-the-art methods. In addition, we make a comparison of the RecResCon and Bi-GRU on the recommendation effect. The performance of SGPD is better than the model with only RecResCon or Bi-GRU, and has demonstrated the effectiveness of proposed model.

The remainder of this paper is structured as follows. In Section 2, we briefly review related work. In Section 3, we describe in detail our SGPD model. Section 4 presents the setting we used in experiment, the experimental result and result analysis. Finally, our work is summarized in the last section.

II. RELATED WORK

A. RNN BASED RECOMMENDATION METHOD

Currently, the recurrent neural network is becoming more and more popular. Due to its recursive structure, RNN can learn user’s sequence for session-based recommendation [8], [9], [10], [11]. Yu et al. [12] obtained shopping basket vector representation by pooling operation, and implemented sequence recommendation. The proposed method effectively implements the next shopping basket recommendation. Liu et al. [13] proposed the POI recommendation technique incorporating spatio-temporal context RNN (ST-RNN). The proposed method improves the RNN by
essentially integrating time information and geographic information in the time and distance transfer matrices, respectively.

When conventional RNN model analyses sequence, it is prone to problems, such as gradient explosion or gradient disappearance during backpropagation [14]. The long-short term memory (LSTM) [15] and gated recurrent unit (GRU) [16] are improved based on the RNN. The above problems are effectively addressed by adding gating unit, selecting memory input and controlling output. Zhao et al. [17] integrated time and distance gates based on the LSTM, and proposed a POI recommendation method integrating spatial temporal long-short term memory (ST-LSTM). Quadrana et al. [18] proposed a personalized session-based recommendation method based on hierarchical RNN (HRNN). The method extends the RNN session model by adding GRU layer that can model the users’ activities in different sessions. HRNN provides users with session-based personalized recommendation. Research shows that RNN can easily generate spurious dependency by over-assuming that any neighboring terms in the long sequence are highly dependent. Therefore, how to use the advantages of other neural networks, and learn sequential general patterns in different time ranges should be further investigated.

B. CNN BASED RECOMMENDATION METHOD
Recently, the convolutional neural network first embeds a user-item interaction sequence into the matrix, treats this matrix as an image, and finally uses convolutional neural network to learn local feature of sequence to achieve session-based recommendation [19]. Tang et al. [20] proposed a convolutional sequence embedding recommendation model (Caser). The model represents the user sequence as an image by the embedding method, and learns local feature of the sequence using a convolutional kernel. This approach provides a unified network structure for capturing user preference and sequence pattern. Yuan et al. [21] proposed a simple convolutional generative network for next item recommendation (NextItNet). It is a deep neural network model with dilated convolutions, which are specifically designed for modelling long-range item dependence. Xing et al. [22] proposed a context-aware POI recommendation model based on convolutional neural network. The model uses the convolutional neural network as a fundamental framework for POI recommendation, and incorporates three contextual information including location property, user interest and emotion. Because the convolutional neural network can learn local feature of the sequence in a session, it can partly compensate for the shortcoming of recurrent neural network.

C. HYBRID DEEP LEARNING BASED RECOMMENDATION METHOD
Following the success of diverse deep learning methods, some researchers have proposed to design hybrid neural network for session-based recommendation task. Li et al. [23] proposed a hybrid neural attentive recommendation machine (NARM) to model the user’s sequential behavior, and capture the user’s main purpose in the current session. Zhang et al. [24] proposed a hybrid neural model (SGINM), for jointly learning sequential feature and general interest feature of each sequence for session-based recommendation. In SGINM, they design an attentive GRU, which captures representation of each sequence. Furthermore, the SGINM adopts a fully-connected neural network with residual connection to learn the general interest of each sequence. Bach et al. [25] proposed a recurrent convolutional network model (RecConRec). The hybrid model combines a CNN layer that operates on item embedding to extract local pattern with a RNN layer that is able to model long-term sequential pattern. The model can capture both local and long-term dependencies in sequence. RecConRec is closely related to our work. On the one hand, it extracts feature of the subsequence by classical CNN and stacked highway network, while we propose a kind of recurrent residual convolution to learn deep general pattern; on the other hand, it is still a unidirectional model using a GRU, while we design a Bi-GRU model to learn sequential dependency.

III. OUR APPROACH
A. PROBLEM FORMULATION
Session-based recommendation aims to predict the next action based on current session of user. Let $U = \{u_1, u_2, \ldots, u_{|U|}\}$ denotes the user set, let $I = \{i_1, i_2, \ldots, i_{|I|}\}$ denotes the item set, and let $S = \{S_1, S_2, \ldots, S_{|S|}\}$ denotes the sessions, where $|U|$, $|I|$ and $|S|$ are the total number of unique users, items and sessions, respectively. $S_i = \{v_1, v_2, \ldots, v_t\}$ denotes the sequence $i$, where $v_t \in I$ denotes the item at time $t$. Given the user $u$, the goal is to predict the next action for target user.

B. OVERVIEW
According to the characteristics of convolutional neural network and recurrent neural network, we propose a kind of SGPD model. The model consists of four layers, namely, embedded layer, RecResCon layer, Bi-GRU layer and user preference layer, as is shown in Fig. 2.

Firstly, the input sequence is converted to an embedding matrix in the embedding layer. We set the sliding window in the embedded layer. Secondly, subsequence within the window is inputted into the RecResCon layer. Local general pattern of a subsequence is deeply learned by recurrent residual convolution network. The sliding window is then moved from left to right to learn general patterns of different subsequences. Thirdly, the sequence is scanned from forward and reverse direction by Bi-GRU to analyze dependency feature of the sequence. Finally, we will extract the personalized potential characteristics of the sequence in the user preference layer.

C. EMBEDDING LAYER
We begin by encoding item IDs in a continuous low-dimensional space. Although one-hot vector representation...
has been shown to work well for session-based recommendation, it is useful to embed items in space of low dimension. This reduces the complexity of convolutional operation and the number of convolutional filter [25]. Formally, let \( E \in \mathbb{R}^{|I| \times d} \) be a matrix consisting of the item latent factor, where \(|I|\) is the size of the item dictionary, and \(d\) is the number of latent dimension. The embedding matrix \(E\) is initialized by a gaussian distribution function, where the mean \(\mu = 0\), the variance \(\sigma = \frac{1}{d} \). We first transform input sequence into fixed-length \(S_i = \{v_1, v_2, \ldots, v_m\}\), where \(m\) represents the maximum length of sequence \(S_i\). Then we apply an embedding lookup to transform \(S_i\) into the item embedding \(E_S \in \mathbb{R}^{d \times m}\).

D. RecResCon LAYER

We apply a sliding window, and consecutive items within the window generate item block in a session. Let the sliding window size denote \(w\). The index range of sliding window is \(idx = \{1, 2, \ldots, m - w + 1\}\). The subsequence matrix within the window is \(E_j \in \mathbb{R}^{d \times w}\), where \(j\) is the current index of the sliding window, and \(j \in idx\). Bach et al. extracted features of the subsequence by CNN and stacked highway network [25]. However, deeper networks do not benefit the model, and may cause some degree of damage in highway network [26]. The residual network achieves good performance in the field of computer vision [27]. Therefore, we propose an optimized recurrent residual convolution layer, whose each recurrent residual block consists of a set of normalization and convolution operations. We consider the block function as follows:

\[
E'_j = W(\phi(\psi(E_j)))
\]
where $\psi$ represents the layer-normalization function, $\phi$ represents the ReLU activation function, and $W$ is the 1-dimensional convolution function, where convolutional kernel $\text{kernel} \_\text{size}$ equals to 1, the input and output channel $\text{channel} \_\text{size}$ equals to $d$. We connect the input subsequence with the output of each recurrent residual block. The output of the first block can be expressed as $E^1_j = F(E^0_j) + E^j$, and other blocks can be defined as $E^j = F(E^j_{j-1}) + E^j$, where $F$ denotes the block function. The sliding window is then moved from left to right to learn general patterns of different subsequences. The outputs of all subsequences are $E^1, E^2, \ldots, E^{m-w+1}$, respectively. Finally, we adopt a fully-connected neural network to transform them into the feature vectors $e_1, e_2, \ldots, e_{m-w+1}$ as follows:

$$e_j = \sigma(W^1e_j + b_1)$$

where $W^1 \in \mathbb{R}^{d \times d}$ is weight matrix, $b_1 \in \mathbb{R}^d$ is bias vector, and $\sigma$ is the sigmoid function, $\sigma(x) = \frac{1}{1+e^{-x}}$.

### E. BI-GRU LAYER

Although we use RecResCon layer with a sliding window to capture sequential general pattern, it is not enough to represent sequential dependency, which can reflect user’s overall preference in a session. In the Bi-GRU layer, we scan a sequence from forward and reverse direction by the Bi-GRU model, and learn the personalized potential information of a user. The Bi-GRU operates on an input sequence of variable length $x_1, x_2, \ldots, x_{m-w+1}$. In our case, the input vector $x_t$ at time step $t$ is the feature vector $e_j$ extracted from the item block ending at $j$, as described above. The GRU model is formulated as follows:

$$r_t = \sigma(W^r_xh_{t-1} + W^r_xi_t + b_r)$$

$$z_t = \sigma(W^z_xh_{t-1} + W^z_xi_t + b_z)$$

$$\hat{h}_t = \tanh(W^h_x(r_t \odot h_{t-1} + W^h_xi_t + b_h))$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \hat{h}_t$$

where $x_t$ and $h_{t-1}$ stand for input vector of current time $t$ and input vector of previous time $t-1$. The $r_t$ stands for reset gate that controls the amount of information to be retained at the previous moment by the hadamard operation $r_t \odot h_{t-1}$. The $z_t$ stands for update gate that controls the proportion of input to the current time $h_t$ and previous time $h_{t-1}$ by the hadamard operations $z_t \odot \hat{h}_t$ and $(1 - z_t) \odot h_{t-1}$, respectively. $\sigma$ is the sigmoid function, and $\tanh(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$.

### F. USER PREFERENCE LEARNING

In the user preference layer, the result of forward scanning of the sequence is set to $\{h_{p1}, h_{p2}, \ldots, h_{pm-w+1}\}$, and the result of reverse scanning of the sequence is set to $\{h_{q1}, h_{q2}, \ldots, h_{qm-w+1}\}$. Feature vector representation of a user is $c = \frac{1}{2(m-w+1)} \sum_{i=1}^{m-w+1} (h_{pi} + h_{qi} - w_{pi} - w_{qi})$. Finally, we can obtain the high-level abstraction feature of the sequence by a fully-connected neural network as follows:

$$z = \phi(W^2c + b_2)$$

where $W^2 \in \mathbb{R}^{d \times d}$, $b_2 \in \mathbb{R}^d$ are parameters, and $\phi$ is an activation function, ReLU here. The predicted score for the user accessing to the next item is calculated as follows:

$$\hat{y} = q_{m+1} \times z$$

After having obtained the session representation $z$, we can compute the score $\hat{y}$ for each true target item $q_{m+1}$.

### G. MODEL TRAINING

The model defines the objective function by cross-entropy. The positive samples $\{p|p \in S_i\}$ and negative samples $\{n|n \notin S_i\}$ are used to generate the training set in pair-wise method. The likelihood function of predictive score in the sequence is defined as follows:

$$p(S|\Theta) = \prod_{S \in \mathcal{S}} \left( \prod_{p \in S_i} \sigma(\hat{y}_p) \prod_{n \notin S_i} (1 - \sigma(\hat{y}_n)) \right)$$

The likelihood function negative log is set to the objective function, and minimizing the objective function is defined as follows:

$$\min J(\Theta) = \sum_{S \in \mathcal{S}} \left( \sum_{p \in S_i} -\log(\hat{y}_p) + \sum_{n \notin S_i} -\log(1 - \sigma(\hat{y}_n)) \right)$$

The minimization of the objective function can update the relevant parameters by the adam descent method until the objective function converges. The concrete steps of SGPD method for fusing recurrent residual convolution and Bi-GRU are as follows:

### IV. EXPERIMENT

In this section, we first present our experimental setting. Then, we conduct experiments to answer the following research questions:

**RQ1**: How does the performance of the proposed method compare to that of the state-of-the-art methods?

**RQ2**: What are the influences of the components of the model, such as recurrent residual convolution, and Bi-GRU?

**RQ3**: How do the key hyper-parameters, for example, vector dimension, sliding window size, and recurrent residual convolution depth, affect model performance?

### A. DATASET

We experiment on four real-world datasets: Foursquare, Gowalla, Yoochoose, and MovieLens. Foursquare is a collection of check-ins made within Singapore. This dataset has previously been used in other studies [28]. The second one is the Gowalla check-ins within California and Nevada. The Gowalla dataset has previously been used in [13]. Both

1. https://sites.google.com/site/yangdingqi
2. http://snap.stanford.edu/data/loc-gowalla.html
3. http://2015.recwschallenge.com/challenge.html
4. https://grouplens.org/datasets/movielens/
**Algorithm 1** Training of SGPD

Input: set of sequence $S$, learning rate $lr$.
Output: SGPD model parameters $\Theta$.

1. Construct training data
   - For each sequence $S_i$ for user $u$ do
   2. $neg_e = generate\_negative(u)$
   3. Add $(neg_e, pos_e, S_i)$ to $D$
   4. End for

5. Training model parameters
6. While (exceed(maximum number of iteration) == False) do
   7. Get a training subsequence $S_j$ from $S_i$
   8. $E_j = item\_embedding(S_j)$
   9. $E_j = MLP\left(E_j\right), j = 1, 2, \ldots, m - w + 1$
   10. $e_j = \text{RecResCon}\left(E_j\right), j = 1, 2, \ldots, m - w + 1$
   11. $hp, hq = \text{biGRU}\left(e_1, e_2, \ldots, e_{m-w+1}\right)$
   12. $z = \phi(\text{Linear}(hp, hq))$
   13. $J(\Theta) = f(neg_e, pos_e, z)$
   14. $\Theta \leftarrow \Theta - \alpha \frac{\partial J}{\partial \Theta}$
   15. Minimizing the objective (10)
   16. End while
17. Saving the learned parameter set $\Theta$

**TABLE 1. Statistical characteristics of datasets.**

| Datasets     | Foursquare | Gowalla | Yoochoose | MovieLens |
|--------------|-----------|---------|-----------|-----------|
| Sessions     | 4489      | 4525    | 3926      | 1487      |
| Items        | 1419      | 3109    | 6393      | 2802      |
| Total actions| 17970     | 43333   | 34333     | 42260     |
| Avg. user check | 8.003     | 9.576   | 8.746     | 10.513    |
| Avg. item checked | 12.663   | 13.937  | 5.371     | 15.082    |
| Density      | 2.82E-3   | 3.08E-3 | 1.37E-3   | 1.01E-2   |

**B. EVALUATION METRICS**

In order to evaluate the performance of session-based recommendation method, the precision rate (Precision@K), the recall rate (Recall@K), and the mean reciprocal ranking (MRR@K) are used as the evaluation indicators, where $K$ represents the number of recommended items. The Precision represents the proportion of correctly recommended items amongst the top-K items. The Recall represents the proportion of correctly recommended items amongst the total number of items accessed by the user. The MRR is the average of reciprocal ranks of the correctly recommended items. The MRR measure considers the order of recommendation ranking, where larger MRR indicates that correct recommendation items are in the top of the ranking list.

\[
\text{Precision} = \frac{|R(u) \cap T(u)|}{|R(u)|} \quad (11)
\]

\[
\text{Recall} = \frac{|R(u) \cap T(u)|}{|T(u)|} \quad (12)
\]

\[
\text{MRR} = \frac{1}{|Q|} \times \sum_{i=1}^{\left\lfloor Q \right\rfloor} \frac{1}{\text{rank}_i} \quad (13)
\]

where the $R(u)$ is the collection of items recommended for the user $u$, the $T(u)$ is the actual collection of items for the user $u$ on the testing set, $|Q|$ represents the number of users. The $\text{rank}_i$ represents the permutation position of the first hit in the recommended list for the user $i$.

**C. PARAMETER SETTING**

When performing the session-based recommendation experiment, we use grid search to find the optimal parameter values in the validation set [25]. The range of the learning rate parameter is $[0.1, 0.01, 0.001, 0.0001]$, and the range of the regularization parameter is $[0.3, 0.03, 0.003, 0.0003]$. The number of training epoch_num is 30, the size of batch batch_size is 32, the length of the sequence $L$ is 5, the recommended length $k$ is 5, 10 or 20, respectively. The experimental platform is composed of Intel i5 CPU, 8G RAM, Windows 7 operating system. Pycharm is used as the development tool. Python 3.5 is used as the program language. Pytorch is used as the neural network learning framework.

**D. PERFORMANCE COMPARISON (RQ1)**

We conduct comparisons with the following representative baselines to evaluate the performance of our proposed method.
TABLE 2. Performance comparison results for four datasets.

| Datasets     | Evaluation | BPR-MF | FPMC-LR | GRU4Rec | Caser | NextItNet | RecConRec | SGPD |
|--------------|------------|--------|---------|---------|-------|-----------|-----------|------|
| Foursquare   | P@20       | 0.0011 | 0.0013  | 0.0015  | 0.0017| 0.0022    | 0.0057    | 0.0074|
|              | R@20       | 0.0132 | 0.0186  | 0.0239  | 0.0308| 0.0357    | 0.1131    | 0.1484|
| Gowalla      | P@20       | 0.0013 | 0.0016  | 0.0018  | 0.0021| 0.0025    | 0.0031    | 0.0036|
|              | R@20       | 0.0281 | 0.0304  | 0.0387  | 0.0398| 0.0411    | 0.0502    | 0.0624|
| Yoochoose    | P@20       | 0.0042 | 0.0058  | 0.0086  | 0.0094| 0.0103    | 0.0154    | 0.0157|
|              | R@20       | 0.0021 | 0.0028  | 0.0031  | 0.0038| 0.004    | 0.0045    | 0.0051|
| MovieLens    | P@20       | 0.0435 | 0.0462  | 0.0664  | 0.0753| 0.0782    | 0.0861    | 0.0982|
|              | R@20       | 0.0138 | 0.0198  | 0.0201  | 0.0226| 0.0232    | 0.0287    | 0.0312|

- BPR-MF [34] optimizes the matrix factorization model on implicit feedback data using a pairwise ranking loss.
- FPMC-LR\(^5\) [35] is a recommendation method based on three-order tensor decomposition, which incorporates matrix factorization and Markov chain.
- GRU4Rec\(^6\) [36] is a recommendation method based on gated recurrent unit.
- Caser\(^7\) [20] is a recommendation method based on convolutional neural network. The proposed method converts the user’s sequence into an image structure, then extracts local feature of the sequence by horizontal and vertical convolution, and finally integrates user’s the local and global features.
- NextItNet\(^8\) [21] is a recommendation method based on convolution neural network, which uses holed convolution to extract long-term feature of sequence, and uses residual network to extract deep feature of sequence.
- RecConRec [25] is a recurrent convolutional architecture that takes the advantages of both complex local feature extracted by CNN and long-term dependency learned by GRU from sequence.

For the first five methods, we used the implementations provided by the authors of these methods. We reimplemented the RecConRec model with the code provided by Caser.

We compare SGPD with baselines to evaluate performance. Tables 2 summarizes the highest Precision@20, Recall@20 and MRR@20 values for each method on Foursquare, Gowalla, Yoochoose, and MovieLens, respectively. As expected, BPR-MF and FPMC-LR are the worst performers on all datasets. Because BPR-MF and FPMC-LR do not analyze the sequence feature of the user’s behavioral data. The GRU4Rec outperforms the BPR-MF and FPMC-LR in terms of Precision, Recall, and MRR. It shows that the recurrent neural network can learn the sequence feature in a session. This method is certified in both text classification and machine translation for natural language processing.

In terms of convolutional neural network, the Caser improved by an average of 16.5 %, 12.4 %, and 12.6 % in P@20, R@20, and MRR@20, as compared to GRU4Rec. The Caser uses both horizontal and vertical convolution to extract the local feature of the sequence. In the user’s sequence, the user’s behavior changes dynamically. For example, user first buy mobile phone, mobile hard drive, and then buy toothpaste and toothbrush. The method can solve the problem of generating spurious dependency of sequence to some extend. The NextItNet over the Caser improved by 15.2 %, 6.9 % and 12.1 % in P@20, R@20, MRR@20. It is useful to learn deep local feature of the sequence for convolution neural network based on residual network.

In the hybrid neural network, the RecConRec outperforms the GRU4Rec, Caser, and NextItNet on four datasets. The method can combine a CNN layer that operates on item embedding to extract local feature with a RNN layer that is able to model long-range sequential pattern. The model is allowed to capture both local and long-term dependencies in a session. It indicates that hybrid neural model can better improve the performance of session-based recommendation.

SGPD is greatly improved in P@20, R@20, and MRR@20 compared with the RecConRec, with an average improvement of 18.2%, 17.9%, and 14.3%. There are two main reasons for advantage. Firstly, we can better extract deeper general pattern of the subsequence by stacked recurrent residual convolution network. Secondly, bidirectional recurrent neural network can better learn dependency relationship of sequence than unidirectional recurrent neural network.

E. ABLATION STUDY (RQ2)
To verify the importance of recurrent residual network and bidirectional recurrent neural network, we compare the different components of model. SGPD-RecResCon only uses the bidirectional recurrent neural network to learn sequential dependency. SGPD-BiGru only uses the recurrent
residual convolution network to learn sequential pattern. SGPD-RecResCon-Bi only uses the unidirectional recurrent neural network to learn sequential dependency.

Figure 3 shows that the performance of SGPD is the best in the four datasets. SGPD-RecResCon and SGPD-RecResCon-Bi perform the worst in Recall and MRR compared to SGPD and SGPD-BiGru, indicating that the deep local pattern of the sequence is very important for session-based recommendation. Moreover, SGPD-RecResCon performs better than SGPD-RecResCon-Bi, indicating that the Bi-GRU is superior to the GRU.

F. INFLUENCE OF HYPERPARAMETERS (RQ3)

Now, we evaluate the influence of different hyperparameters based on SGPD method. There are three kinds of hyperparameters to be discussed, such as vector dimension, sliding window size, and recurrent residual convolution depth. In this paper, we compare their effects with the MRR on four datasets.

1) ANALYSIS WITH VECTOR DIMENSION

Vector dimension is very important in the model, and has important effect on the recommended result. We compare SGPD with GRU4Rec, Caser, and RecConRec, by experimenting MRR@20, and setting the vector dimension $\text{hiddensize}$ equal to [8, 16, 32, 64, 128, 256].

Figure 4 shows that when $\text{hiddensize}$ increases, the MRR increases first, and then gradually decreases. When observing results from the data viewpoint, we find that the data density of the MovieLens is greater than the others. In other words, the increase of $\text{hiddensize}$ does not necessarily lead to better results in model performance, especially for low-density data. In addition, the performance of SGPD outperforms the other three kinds of methods in different vector dimensions.

2) ANALYSIS WITH SLIDING WINDOW

When training sessions, our proposed SGPD first generates the item block of width $w$, then moves sliding window with stride width of 1 from left to right. The different window widths can extract different local patterns of sequence in a session. We set the recommended list length $L$ equal to 5, and sliding window $w$ equal to [1, 2, 3, 4, 5].

Figure 5 shows that when $w$ increases, the MRR gradually increases in the Foursquare and Gowalla. When $w = 4$, the improvement saturates. We notice that the length of check-in pattern is relatively long, and the smaller width may destroy the continuity and integrity of the user’s POI sequence. In the Yoochoose and MovieLens, when $w$ increases, the MRR first increases. When $w = 2$, the MRR reaches the maximum value. When $2 < w \leq 5$, the MRR gradually decreases. We find that the length of consumption pattern is generally relatively short, and bigger width may make user consumption pattern contain many spurious dependencies.
FIGURE 4. Comparison experiment of vector dimension.

FIGURE 5. Comparison experiment of sliding window.
3) ANALYSIS WITH NETWORK DEPTH
To study the effectiveness of the deep network architecture, we compare recurrent residual network with residual network and highway network. The window width of Foursquare and Gowalla is chosen to be 4, and the window width of Yoochoose and MovieLens is chosen to be 2. The network depth $d$ is $[50, 100, 150, 200, 250, 300]$, and the other hyper-parameters remained unchanged.

As shown from Figure 6, the MRR of recurrent residual network is better than residual network and highway network. In four datasets, when $d$ increases, the MRR increases gradually. However, the MRR does not constantly increase. When $d$ exceeds a certain threshold, the MRR changes very little and becomes stable. When $d = 200$, it is more appropriate for the check-in datasets, such as Foursquare and Gowalla. When $d = 150$, it is enough for the consumer datasets, such as Yoochoose and MovieLens. The experimental results indicate that increasing the network depth can improve the performance of the session-based recommendation. However, we should choose optimal network depth for different kinds of datasets.

V. CONCLUSION
In this paper, we propose a hybrid neural model for integrating recurrent residual convolution network and bidirectional recurrent neural network. Sequential general pattern within the window is extracted in the RecResCon layer, then the sequential dependency is learned in the Bi-GRU layer, and we finally predict the score for the user accessing to the next item in the user preference layer. We have conducted experiments in four public datasets. The experimental results demonstrate the superiority of the proposed SGPD compared with the other state-of-the-art models. For future work, we will further improve performance of session-based recommendation by incorporating social network, and more effective side information.

REFERENCES
[1] J. Wang, P. Huang, H. Zhao, Z. Zhang, B. Zhao, and D. L. Lee, “Billion-scale commodity embedding for E-commerce recommendation in Alibaba,” 2018, arXiv:1803.02349.
[2] N. Lim, B. Hooi, S.-K. Ng, X. Wang, Y. L. Goh, R. Weng, and J. Varadarajan, “STP-UDGAT: Spatial-temporal-preference user dimensional graph attention network for next POI recommendation,” in Proc. 29th Int. Conf. Inf. Knowl. Manage., 2020, pp. 845–854, doi: 10.1145/3340531.3411876.
[3] C. Cheng, H. Yang, I. King, and M. R. Lyu, “Fused matrix factorization with geographical and social influence in location-based social networks,” in Proc. Int. Conf. Artif. Intell., 2012, pp. 17–23, doi: 10.5555/2900728.2900731.
[4] R. Salakhutdinov and A. Mnih, “Probabilistic matrix factorization,” in Proc. 20th Int. Conf. Neural Inf. Process., 2007, pp. 1257–1264, doi: 10.5555/2981562.2981720.
[5] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009, doi: 10.1109/MC.2009.263.
[6] Q. Cui, S. Wu, Q. Liu, W. Zhong, and L. Wang, “MV-RNN: A multi-view recurrent neural network for sequential recommendation,” IEEE Trans. Knowl. Data Eng., vol. 32, no. 2, pp. 317–331, Feb. 2020, doi: 10.1109/TKDE.2018.2881260.
[7] T. X. Tuan and T. M. Phuong, “3D convolutional networks for session-based recommendation with content features,” in Proc. 11th ACM Conf. Recommender Syst., Aug. 2017, pp. 138–146, doi: 10.1145/3109859.3109900.
