Dual-Channel Convolutional Recurrent Networks for Session-Based Recommendation

Jingjing Wang, Lap-Kei Lee, and Nga-In Wu

Abstract Recommender systems assist a Web application user in satisfying their needs or interests based on the user profile and past activities. Yet due to privacy and other concerns, some applications and services only keep anonymous information. A session-based recommender system (SRS) predicts the next item by exploring only anonymous user-item behavior orders during ongoing sessions. Recurrent neural networks (RNNs) and their two variants have dominated the research on SRS. However, there are two shortcomings in these RNN-based methods: (1) RNNs easily generate false dependencies because RNNs assume all adjacent items are highly dependent on each other; (2) the sequentially connected architecture of RNNs can only capture the point-level dependencies but ignoring neglecting the union-level dependencies. This paper proposes a Dual-channel Convolutional Recurrent Neural Network (D-CRNN) model to address these problems. This hybrid model leverages RNN to explore complex long-term dependencies and combines CNN to extract the union-level context features, which help to reduce the noise. The hybrid model was evaluated on three commonly used real-world datasets. The experimental results on Diginetica dataset D-CRNN showed an improvement of 5.8% and 4.8% respectively in terms of Recall@10 and MRR@10, demonstrating the effectiveness of D-CRNN on the session-based recommendation.

Keywords Session-based recommendation · Convolutional neural networks · Recurrent neural networks · Hybrid neural networks

J. Wang (✉) · L.-K. Lee
School of Science and Technology, Hong Kong Metropolitan University, Ho Man Tin, Hong Kong SAR, China
e-mail: s1245831@ouhk.edu.hk

L.-K. Lee
e-mail: lklee@ouhk.edu.hk

N.-I. Wu
College of Professional and Continuing Education, Hong Kong Polytechnic University, Kowloon, Hong Kong SAR, China
e-mail: ngain.wu@cpce-polyu.edu.hk

D. P. Agrawal et al. (eds.), *Cyber Security, Privacy and Networking*, Lecture Notes in Networks and Systems 370, https://doi.org/10.1007/978-981-16-8664-1_25

1 Introduction

Recommender systems (RSs) act an important role in real-world applications by assisting users in satisfying theirs needs or interests based on the sequential records of user-item interactions, such as rating, viewing, or clicking items. However, in many real-life scenarios, due to privacy and other concerns, informative profiles are not provided, and only anonymous and chronological behavior orders during ongoing sessions are available. Approaches that model these user past interactions on anonymous sessions to predict the next action are called session-based recommender systems (SRS).

The most widely used methods for SRS are Recurrent Neural Networks (RNN) and their two variants, namely the Long Short-Term Memory (LSTM) [1] and the Gated Recurrent Unit (GRU) [2]. RNNs can remember former states to hold the long-term dependencies in the dynamic and evolving sequences, benefiting from the internal memory cells and loop operation. However, gradient vanishing will be easily caused via multiple layer iterations, which motivates the emergence of the two variants. Though these variants solved the gradient issues and obtained exciting results, there are still two shortcomings in these models: (1) The sequentially connected network structure in RNN makes it easily generate false dependencies because not all the adjacent are related; irrelevant and noisy interactions such as clicked out of curiosity or by accident commonly occur in a real-world session. (2) RNNs focus on explore the transition correlation at the point-level while ignore the collective dependencies at the union-level.

For example, given a session S_1 = {a bacon, a rose, eggs, bread, a box of butter, an iPhone} which denotes items added into the cart successively by a user. The user first purchased food for breakfast, including bacon, egg, bread, and butter, and then added pieces of rose for decoration due to being attracted by some advertisements. Finally, the user ended this shopping session by picking up an iPhone. In this case, the user's purpose has been changed from breakfast to cellphone dynamically. Yet no matter which purpose is, the item "rose" is irrelevant to the whole sequence and is a noisy item. The breakfast can be regarded as the long-term preference, and the cellphone is the short-term preference as it is closer to the prediction one. Then the next item with a high probability is a bottle of milk for breakfast or a phone accessory for the cellphone. Obviously, an effective recommender method should not only encode the sequential items with the long-term and the short-term dependencies but also need to distinguish the noisy items and alleviate their influence. Apart from this, we also observed that in the real world, not all the items needed to be strictly ordered in the sequences, e.g., if the model wants to recommend an airpod, it needs the user to buy an iPhone first; while it makes little sense whether to buy eggs or bread first when a user wants to buy milk for breakfast. As to the considerable sequential patterns, RNN performs well; while others maybe not, such as in the breakfast case, the milk will have a higher probability of purchase when both eggs and bread are already purchased. The above observations occurred commonly in the real world as various advertisements appeared to catch users' attention, and thus any items may be added

to the cart. These phenomena indicated that relying on the pure RNN model might not be the perfect solution for an SRS.

In this paper, we explore a novel hybrid model of combining Convolutional Neural Network (CNN) and RNN to make up for the drawbacks in RNN mentioned above. CNNs are capable of extracting complex local patterns, and the convolution operation is beneficial to filter out the irrelevant features in the current reception field. Furthermore, these local features captured by CNN can be regarded as a union-level correlation between items. Specifically, we propose a hybrid dual-channel modeling architecture with RNN and CNN, named D-CRNN, to capture user behavior in a more reasonable way to fit the shortcoming of using pure RNN-based models. We first treat the item embedding matrix as an image and search the contextual information using CNN. Then the output of the different size convolutional filters will be separately fed into dual-channel RNN to generate the long-term preferences. At last, we apply the target-aware attention mechanism to assign different weights to each channel to generate the session representation.

Our contributions of this work are as follows:

1. A multi-channel D-CRNN utilizes the RNN to explore long-term sequential patterns and leverages CNN to enhance the impact of the short-term dependencies, which is helpful to alleviate the effects of noises.
2. Each channel can capture the long-term and short-term dependencies with different filter sizes, and then adaptively activate channels with the target attentive network to generate the final session representation.
3. We conduct the experiments and analysis on three real-world session datasets. Experimental results demonstrate the effectiveness of D-CRNN.

1.1 Related Work

This section introduces the related work based on pure RNN or CNN and some hybrid models.

RNN-Based SRS. The recurrent architecture of RNN makes it a natural solution for the sequential problems on deep learning-based SRSs. In other words, RNN and its variants (LSTM and GRU) have dominated the studies on SRS. The representative model is proposed by Hidasi et al. [3]. They trained the model with the pairwise ranking loss and achieved an encouraging result compared to the traditional methods. In their following paper, Quadrana et al. [4] designed a parallel RNN architecture to model multi-modal feature representations as data augmentation to improve the personalized recommendation accuracy. Another pure RNN-based model is proposed by Wu et al. [5], which employed the LSTM to solve the personalized recommendation problem.

CNN-Based SRS. Unlike the recurrent architecture in RNN, CNN is well known as a locally connected network to act as "feature extractors". The typical work applying

CNN in the session recommendation task is proposed by Tang and Wang [6]; in their model (named Caser), CNN takes the whole interaction matrix as an image with time and space, then performs horizontal and vertical convolution separately to extract features. To solve the CNN's capable ability for long-term dependencies issues, Yuan et al. [7] introduced the holed convolution by increasing the receptive field to make up the shortcoming. Recently, Yan et al. [8] encoded the sequence embedding as a three-way vector and employ a 2-D convolution to capture the complex long-range dependencies. Yet the issue of long-term dependencies still exists in CNN because continuously increasing the size of the convolutional kernel filter size will make CNN ineffective to capture the short-term dependencies.

Hybrid SRS. Though these SRSs built on a pure model/technique showed their effectiveness, there are still many limitations with these basic models. To this end, some more powerful hybrid models were proposed to address the particular challenges. For example, Li et al. [9] proposed an encoder–decoder model that combines RNN and attention mechanism for the session recommendations. Jannach and Ludewig [10] proposed to enhance the session representation by its k-nearest neighborhood sessions, which is a hybrid model based on the k-nearest neighbors and RNN. Guo et al. [11] proposed a hybrid model based on matrix factorization and RNN for music recommendation. Xu et al. [12] designed a combination of CNN and RNN, in which all item embeddings were first fed into the RNN to generate the hidden state, and then CNN was applied to search the significant local features. This method also suffered from noise issues. Bach et al. [13] decomposed the original sequences into multiple sub-sequence blocks, then applied CNN to generate union features from each block, these local features were regarded as the timestamp input of GRU to compute the probability among all the candidates. This method verified the importance of the local union features in the recommendation task; however, the pooling operation in CNN will miss the item's features information. Recently, several researchers applied graph neural networks (GNNs) [14–16], especially gated graph neural networks (GGNNs) to generate item embeddings [17–19]. Although graph-based methods are helpful in mining high-order relationships, these methods usually require a huge amount of memory to store and update the node features.

Unlike the aforementioned methods, we incorporate RNN and CNN in an innovative way. Our method fed the whole sequence into CNN and generated multiple context-aware subsequences as the dual-channel input separately. Thus, the long-term and short-term dependencies will be effectively captured. We performed extensive experiments to verify the effectiveness of our method.

2 The Proposed Approach

Our proposed D-CRNN incorporates CNN to GRU to learn sequential features. The task of the session-based recommendation is formulated as follows. Let $S = [s_1, s_2, \ldots, s_N]$ denotes the set of all anonymous session sequences, let

$I = \{i_1, i_2 \ldots, i_M\}$ represents all unique items collected from S. An session s can be represented as $s = \{i_1, i_2 \ldots, i_t\}$, where i_t denotes the t timestamp clicked item in s. We fixed the length of the training sequences within L, and repeatedly add vector $\mathbf{0}$ if there are not enough items in a session.

The architecture of D-CRNN is illustrated in Fig. 1. D-CRNN has three main components: (1) convolution layer, (2) GRU layer, and (3) channel selection layer. The convolution layer is employed to transform the original sequence into the GRU input. Then, the GRU layer learns the long-term dependencies from these subsequences. Finally, the target-aware module adaptively selects the channels to generate the final session representation.

Convolutional Layer. Given the session sequence $s = \{i_1, i_2 \ldots, i_t\}$, D-CRNN first map them into a continuous, lower-dimensional space by an embedding lookup table. Then we apply n different sizes of filters to explore the local correlation with multi-scale feature interactions separately. Let $F^k \in \mathbb{R}^{h \times d}$, where $1 \leq k \leq n$, n is the total number of channels, and d represents the item embedding dimension, and $h \in \{1, \ldots, L\}$ is the size of the filter. If $F^k \in \mathbb{R}^{1 \times d}$, CNN can be regarded as a point-level convolution; when $h > 1$, a significant signal feature will be picked up regardless of location, which can be regarded as a union-level representation of the center item and its contextual feature. In each convolution, the filter F will slide from top to bottom over the embedding layer output, and interact with the items i (where $1 \leq i \leq L - h + 1$) to generate the result as follows

$$c_i^k = f(O_{i:i+h-1} \bigodot F^k) \tag{1}$$

Here F^k is the convolutional kernel, and $O_{i:i+h-1}$ is the sub-matrix from the embedding layer, the index of O is the size of the interacting field, $\bigodot$ is the inner product operator, and $f(\cdot)$ denotes the activation function ReLU.

GRU Layer. After the CNN operation, the original embedding matrix has been transformed into a set of fixed length and union-level feature matrices. Now, we

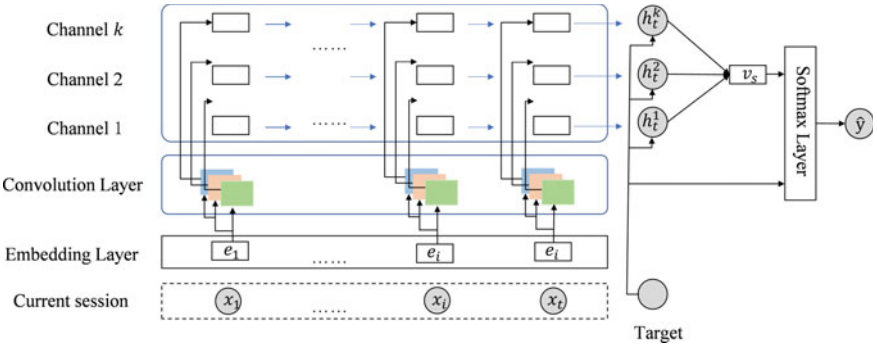

Fig. 1 Framework of D-CRNN model

describe how to feed these union-level features into the RNN layer to capture the long-term dependencies. Specifically, a recurrent network GRU (which is a variant of RNN) will be employed on the CNN output to obtain the hidden state h^k. Here, we choose GRU instead of RNN and LSTM, because the gating mechanism is effective to avoid the exploding/vanishing gradient problem in modeling long-range dependencies and some works showed that GRU performs better than LSTM in the session-based recommendation task [3, 9].

Different from the traditional RNN, where the input x_t demonstrates the item interacted at timestamp t, our input vector $x_t \in \mathbb{R}^d$, $1 \leq t \leq L - h + 1$ denotes the tth convolution operation in the CNN, as described in the convolution layer. We use GRU with the gating mechanism to remember the former long-distance item state information. The detail process of the update state is as follows.

$$z_t = \sigma\left(W_z c_t^k + V_z h_{t-1}^k + b_z\right) \tag{2}$$

$$r_t = \sigma\left(W_r c_t^k + V_r h_{t-1}^k + b_r\right) \tag{3}$$

$$\widetilde{h}_{t-1}^k = \tanh\left(W_h c_t^k + V_h(r_t * h_{t-1}^k) + b_h\right) \tag{4}$$

$$h_t^k = (1 - z) * h_{t-1}^k + z_t * \widetilde{h}_{t-1}^k \tag{5}$$

where $*$ denotes the Hadamard product, σ and tanh are the activation functions, and W, V, and b are trainable parameters in the current channel.

Channel Selection Layer. Previous work usually aggregates the multi-channel final hidden states h_t^m $(1 \leq m \leq k)$ of the k channels with a mean-value attention mechanism, max pooling, or mean pooling. In this paper, we propose to dynamically combine multi-channel embedding to construct final session embeddings by a target-ware attention mechanism. All candidate items are regarded as the targets in this model. Specifically, we use a target attention module to compute weighted value between all channels and each target item $v_i \in I$, its embedding $h_{v_i} \in R^d$:

$$\beta_{i,m} = \text{softmax}(e_i, m) = \frac{\exp(h_t^m W h_{v_i})}{\sum_{m=1}^{k} \exp(h_t^m g_{v_i})} \tag{6}$$

where W is the nonlinear parameters. Then, we aggregate all channels with the weighted factor $\{\beta_{i,m}\}$ to build a dual-channel embedding s_h:

$$s_h = \sum_{m=1}^{k} \beta_{i,m} * h_t^m \tag{7}$$

where $\beta_{i,m}$ is the concentration weight of the target item h_{v_i}. That is, $\beta_{i,m}$ weights the target item to construct the embedding v_C that probably outputs the target item h_{v_i}.

Model Optimization. Given the session embedding s_h, we compute the recommendation probability distribution between all candidate item h_{v_i} and s_h:

$$\hat{y}_i = \frac{\exp\left(h_{v_i}^T s_h\right)}{\sum_{j=1}^{|V|} \exp\left(h_{v_i}^T s_h\right)} \tag{8}$$

where $\hat{y}_i$ is the recommendation score of all candidates at the next time. At last, we optimize the cross-entropy as the objective function and minimizing the loss to train the parameters:

$$\text{Loss}\ (\hat{y}) = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \tag{9}$$

3 Experiments and Analysis

Datasets. We adopted two datasets, namely Yoochoose, Diginetica. In particular, the Yoochoose dataset is collected from the ecommerce website Yoochoose.com. The Diginetica dataset is users' transaction data log.

For Yoochoose, we evaluate the D-CRNN on the most commonly used subdataset 1/64, 1/4. Similar to Li et al. [9], sessions from the latest week were used for testing, and others considered as the training data. In addition, given a session $S = [v_1, v_2, \ldots, v_n]$, we split all the data sequence to produce the training session and the predict item, i.e., $([v_1], v_2), ([v_1, v_2], v_3), \ldots, ([v_1, v_2, \ldots, v_{n-1}], v_n)$. The statistics about the three datasets are shown in Table 1.

Experimental Setup. The number of the batch size is 512; the dimension of embedding size is 50; the hidden size of GRU is 50. The maximum length our model can deal with is 59. We choose three channels with corresponding filter sizes $f = [1, 2, 4]$. We train and optimize our method with Adam. The learning rate $\text{lr} = 0.005$ and decays by 0.1 after every 3 epochs. Dropout layers are used to avoid overfitting: one is after the embedding layer with $\text{dropout} = 0.3$, and the other is after the GRU layer with $\text{dropout} = 0.5$.

Table 1 Statistics of the datasets in our experiments

Datasets	Yoochoose 1/64	Yoochoose 1/4	Diginetica
All clicks	557,248	8,326,407	982,961
Train clicks	369,859	5,917,746	719,470
Test clicks	55,898	55,898	60,858
Num of items	16,766	29,618	43,097
Avg. items	6.16	5.71	5.12

Baselines. We compare and analyze the proposed model with seven representative baselines, including conventional methods and deep learning algorithms:

1. S-POP: S-POP recommends the next-click based on item popularity of the current session.
2. BPR-MF [20]: BPR-MF predicts the next item by decomposing the original user–item interaction matrix and used a pairwise objective function to optimize the matrix factorization.
3. GRU4REC [3]: GRU4REC is the representative work of RNN-based methods in the session-based recommendation task.
4. Caser [6]: Caser is a pure model based on CNN to solve sequential problems.
5. FPMC [21]: FPMC incorporates the Markov chain into matrix factorization to solve the personalized recommendation problem.
6. STAMP [22]: STAMP is a hybrid model based on MLP and attention mechanism.
7. NARM [9]: NARM is a hybrid model based on RNN and attention mechanism.

Evaluation Metrics and Experimental Setup. Two widely metrics are used to evaluate the method performance: Mean Reciprocal Rank (MRR)@10, and Recall@10.

Performance Comparison. Table 2 presents the performance of all methods, where we highlighted in boldface of the best result and underlined the best results of the benchmark to make the comparison clearer. From Table 2, we have the following findings:

1. Although S-POP directly regard the most popular item in the current session as the next one without any complicated statistic or deep learning method, it is not the worst, and is even better than BPR-FM and FPMC, especially on Diginetica. The common problem in FPMC and BPR-FM is that they both ignore the contextual information in the current sequence and cannot deal with the dynamic change of user interests.

Table 2 Performance comparison of D-CRNN with baseline methods

	Yoochoose 1/64		Yoochoose 1/4		Diginetica	
	Recall@10	MRR@10	Recall@10	MRR@10	Recall@10	MRR@10
S-POP	15.31	13.09	14.80	12.79	9.43	7.28
BPR-MF	22.93	10.24	29.30	13.89	4.21	1.89
FPMC	36.12	18.54	37.44	20.05	15.01	6.20
GRU4REC	52.43	24.53	55.49	26.05	17.93	7.73
CASER	59.09	28.08	57.29	28.33	32.64	13.92
STAMP	52.96	25.17	57.67	28.32	33.98	14.26
NARM	57.83	27.42	57.98	28.51	35.44	15.13
Our method	59.79	28.90	60.11	28.94	37.52	15.86

2. Deep learning methods (with the basic model or the hybrid models) consistently performed better than the traditional methods. Compared to the traditional method, deep learning methods are good at capturing complicated correlations. Among these methods, Gre4REC and CASER are the pure models based on RNN and CNN, while others are hybrid models. It demonstrates that a hybrid method is usually a more powerful model when solving solve the same issue. GRU is also a powerful tool for sequential patterns than MLP. It is worth mentioning that although CASER cannot capture the sequential patterns, it also achieves a comparable result. The main reason is that most of the sessions are short due to the limitation of session duration, and items in the sequences are not strictly ordered, so CNN performs better than RNN.
3. D-CRNN achieved the best results in terms of Recall and MRR, outperforming conventional methods and basic neural network models; especially on the Diginetica dataset, D-CRNN improves by 5.8% and 4.8% respectively in terms of Recall@10 and MRR@10. This ascertains the assumption of our method that the operation of CNN can filter the irrelevant features, thus making RNN performs well.

4 Conclusion

In this paper, we designed a hybrid model based on RNN and CNN, which can leverage the advantage of both RNN and CNN and overcome the shortcoming of the single model for the session recommendation. Moreover, the extensive experiment showed the importance of the union-level features and sequential patterns in the sequence prediction task. In the next work, we plan to apply the recent hot graph neural networks to further exploit high-order relationships to improve the accuracy of the recommendation.

References

1. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780
2. Cho K, van Merriënboer B, Bahdanau D, Bengio Y (2014) On the properties of neural machine translation: encoder–decoder approaches. In: Proceedings of SSST-8, eighth workshop on syntax, semantics and structure in statistical translation, pp 103–111
3. Hidasi B, Karatzoglou A, Baltrunas L, Tikk D (2016) Session-based recommendations with recurrent neural networks. In: 4th international conference on learning representations (poster)
4. Quadrana M, Karatzoglou A, Hidasi B, Cremonesi P (2017) Personalizing session-based recommendations with hierarchical recurrent neural networks. In: Proceedings of the 11th ACM conference on recommender systems, pp 130–137
5. Wu CY, Ahmed A, Beutel A, Smola AJ, Jing H (2017) Recurrent recommender networks. In: Proceedings of the 10th ACM international conference on web search and data mining, pp 495–503

6. Tang J, Wang K (2018) Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the 11th ACM international conference on web search and data mining, pp 565–573
7. Yuan F, Karatzoglou A, Arapakis I, Jose JM, He X (2019) A simple convolutional generative network for next item recommendation. In: Proceedings of the 12th ACM international conference on web search and data mining, pp 582–590
8. Yan A, Cheng S, Kang WC, Wan M, McAuley J (2019) CosRec: 2D convolutional neural networks for sequential recommendation. In: Proceedings of the 28th ACM international conference on information and knowledge management, pp 2173–2176
9. Li J, Ren P, Chen Z, Ren Z, Lian T, Ma J (2017) Neural attentive session-based recommendation. In: Proceedings of the 2017 ACM conference on information and knowledge management, pp 1419–1428
10. Jannach D, Ludewig M (2017) When recurrent neural networks meet the neighborhood for session-based recommendation. In: Proceedings of the 11th ACM conference on recommender systems, pp 306–310
11. Guo L, Yin H, Wang Q, Chen T, Zhou A, Quoc Viet Hung N (2019) Streaming session-based recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp 1569–1577
12. Xu C, Zhao P, Liu Y, Xu J, Sheng VSSS, Cui Z, Zhou X, Xiong H (2019) Recurrent convolutional neural network for sequential recommendation. In: The world wide web conference 2019, pp 3398–3404
13. Bach NX, Long DH, Phuong TM (2020) Recurrent convolutional networks for session-based recommendations. Neurocomputing 411:247–258
14. Yu F, Zhu Y, Liu Q, Wu S, Wang L, Tan T (2020) TAGNN: target attentive graph neural networks for session-based recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 1921–1924
15. Pan Z, Cai F, Chen W, Chen H, de Rijke M (2020) Star graph neural networks for session-based recommendation. In: Proceedings of the 29th ACM international conference on information & knowledge management, pp 1195–1204
16. Wang Z, Wei W, Cong G, Li X-L, Mao X-L, Qiu M (2020) Global context enhanced graph neural networks for session-based recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, pp 169–178
17. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th conference on uncertainty in artificial intelligence, pp 452–461
18. Rendle S, Freudenthaler C, Schmidt-Thieme L (2010) Factorizing personalized Markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on world wide web, pp 811–820
19. Liu Q, Zeng Y, Mokhosi R, Zhang H (2018) STAMP: short-term attention/memory priority model for session-based recommendation. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp 1831–1839
20. Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th conference on uncertainty in artificial intelligence, pp 452–461
21. Rendle S, Freudenthaler C, Schmidt-Thieme L (2010) Factorizing personalized Markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on world wide web, pp 811–820
22. Liu Q, Zeng Y, Mokhosi R, Zhang H (2018) STAMP: short-term attention/memory priority model for session-based recommendation. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp 1831–1839