Rethinking Item Importance in Session-based Recommendation

Zhiqiang Pan[1] Fei Cai[1]* Yanxiang Ling[1] Maarten de Rijke[2,3]
[1]Science and Technology on Information Systems Engineering Laboratory,
National University of Defense Technology, Changsha, China
[2]University of Amsterdam, Amsterdam, The Netherlands
[3]Ahold Delhaize, Zaandam, The Netherlands
{panzhiqiang, caifei, lingyanxiang}@nudt.edu.cn, derijke@uva.nl

ABSTRACT

Session-based recommendation aims to predict a user's actions at the next timestamp based on anonymous sessions. Previous work mainly focuses on the transition relationship between items that the user interacted with during an ongoing session. They generally fail to pay enough attention to the importance of the items involved in these interactions in terms of their relevance to user's main intent. In this paper, we propose a Session-based Recommendation approach with an Importance Extraction Module, i.e., SR-IEM, that considers both a user's long-term and recent behavior in an ongoing session. We employ a modified self-attention mechanism to estimate item importance in a session, which is then used to predict user's long-term preference. Item recommendations are produced by combining the user's long-term preference and their current interest as conveyed by the last item they interacted with. Comprehensive experiments are conducted on two publicly available benchmark datasets. The proposed SR-IEM model outperforms start-of-the-art baselines in terms of Recall and MRR for the task of session-based recommendation. In addition, compared to state-of-the-art models, SR-IEM has a reduced computational complexity.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Session-based recommendation, self-attention, item importance.

ACM Reference Format:
Zhiqiang Pan, Fei Cai, Yanxiang Ling and Maarten de Rijke. 2020. Rethinking Item Importance in Session-based Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3397271.3401274

1 INTRODUCTION

Recommender systems help connect people to personalized information in a growing volume of items on offer. Most existing approaches for recommendation focus on a user's interaction history in order to predict their preferences for recommending future items [1, 4]. For cases where historical user-item interactions are unavailable, it is challenging to capture the user's preferences in an accurate manner [2]. For the task of session-based recommendations, we aim to generate recommendations merely based on an ongoing session.

RNNs, attention mechanisms, and GNNs have been widely applied to session-based recommendation. For instance, Hidasi et al. [2] apply a Gated Recurrent Unit (GRU) to model user's sequential behavior in session to capture his instant preference, and Li et al. [5] propose to capture user's main purpose with an attention mechanism. On the basis of NARM, Wang et al. [10] introduce neighbor sessions as auxiliary information to model an ongoing session. In addition, Liu et al. [6] estimate user's general and current interests based on a long-term and short-term memory, respectively. Wu et al. [11] employ Gated Graph Neural Networks (GGNNs) to model the complex transitions between items for producing predictions.

Even though the approaches listed above have all helped to improve the performance of session-based recommendation, they fail to pay enough attention to an important source of information. That is, they can not accurately locate the important items in a session for generating user preferences. After generating item embeddings, the importance of each item is simply determined by its relevance either to the mixture of items in the long-term history [5, 10] or the last single item [11] or a combination [6]. Unavoidably, there are non-relevant items in a session, especially in long sessions, making it hard for models to focus on the important items.

We propose an approach for *session-based recommendation with an importance extraction module*, i.e., **SR-IEM**, that can effectively capture a user's long-term preferences and their current interest. To model a user's long-term preference, we propose an Importance Extraction Module (IEM) that applies a modified self-attention mechanism to extract the importance of each item in an ongoing session. Then, the items are discriminatively combined to predict a user's general preference according to the item importance. To capture a user's current interest, we regard the last item embeddings as an expression of their current interest, which is then combined with the long-term preferences for item recommendation.

Our contributions in this paper are: (1) We propose an Importance Extraction Module (IEM) to accurately obtain the importance of each item for session-based recommendation. The proposed SR-IEM model can simultaneously capture a user's long-term preference and his current interest to make recommendations. (2) We compare the performance of SR-IEM against start-of-the-art baselines on two public datasets and find that it can beat the state-of-the-art

*Corresponding author.

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401274

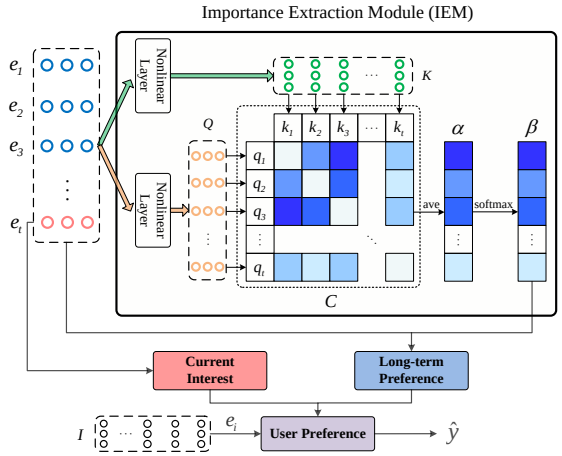


Figure 1: Overview of the proposed SR-IEM model.

models in terms of Recall and MRR. In addition, SR-IEM has a lower computational complexity than competitive neural baselines.

2 APPROACH

Given a session $S = \{x_1, x_2, \ldots, x_t\}$ consisting of t items that a user interacted with, e.g., clicked, the goal of session-based recommendation is to predict the next item from a set of n items $I = \{v_1, v_2, \ldots, v_n\}$ to recommend at time step t+1. Fig. 1 presents an overview of SR-IEM, with three main components, i.e., an importance extraction module (see §2.1), a preference fusion module (see §2.2), and an item recommendation module (see §2.3).

2.1 Importance extraction

To accurately locate the important items in a session for the purpose of modeling user preference, we propose an Importance Extraction Module (IEM) to generate the item importance. We first embed each item x_i in $S = \{x_1, x_2, \ldots, x_t\}$ into a d dimensional representation $e_i \in \mathbb{R}^d$ via an embedding layer. Then, borrowing the merits from the self-attention mechanism [9], we transform the item embeddings $E = \{e_1, e_2, \ldots, e_t\}$ into a different space via a non-linear function to generate the respective *query* Q and *key* K as:

$$Q = \text{sigmoid}(W_q E), \tag{1}$$

$$K = \text{sigmoid}(W_k E), \tag{2}$$

where $W_q \in \mathbb{R}^{d\times d}$ and $W_k \in \mathbb{R}^{d\times d}$ are trainable parameters for the *query* and *key*, respectively; sigmoid is the transformation function to learn information from the item embedding in a non-linear way.

After generating the representation of *query* Q and *key* K, we estimate the importance of each item via a modified self-attention mechanism. First, we compute the similarity of every pair of two items by introducing the affinity matrix C between *query* Q and *key* K as:

$$C = \frac{\text{sigmoid}(QK^\top)}{\sqrt{d}}, \tag{3}$$

where $\sqrt{d}$ is used to scale the attention.

From the affinity matrix, we would like to see that an item is not important if its corresponding similarity scores related to other items are relatively low. A user might interact with such an item by accident or due to curiosity. In contrast, if an item is similar to most items in a session, it may express the user's main preference. That is, the item is relatively important. Inspired by this intuition, we resort to the average similarity between an item and other items in session as the item importance. To avoid high matching scores between identical vectors of *query* Q and *key* K, following [12], we employ a masking operation that masks the diagonal of the affinity matrix. Then, we can assign an *importance score* α_i to each item i:

$$\alpha_i = \frac{1}{t-1} \sum_{j=1, j\neq i}^{t} C_{ij}, \tag{4}$$

where $C_{ij} \in C$. To normalize the scores, a softmax layer is applied to get the final *importance* β of items in the session as:

$$\beta = \text{softmax}(\alpha), \tag{5}$$

2.2 Preference fusion

Through the importance extraction module, we obtain the importance of each item in a session, which indicates the relevance of each item to the user's main purpose. Then, we represent the user's long-term preference z_l by combining the embeddings of items in the session according to their importance as:

$$z_l = \sum_{i=1}^{t} \beta_i e_i. \tag{6}$$

As for the current interest, denoted as z_s, following [6, 11], we directly adopt the embedding of the last item in the session, i.e., $z_s = e_t$. After obtaining a user's long-term preference z_l and his current interest z_s, we combine them into the user's final preference representation z_h that is to be used for item recommendation as:

$$z_h = W_0 [z_l; z_s], \tag{7}$$

where $[\cdot]$ is the concatenating operation. $W_0 \in \mathbb{R}^{d\times 2d}$ transforms the concatenated representation from a latent space $\mathbb{R}^{2d}$ into $\mathbb{R}^d$.

2.3 Item recommendation

Once the user's preference representation in a session has been generated, we use it to produce item recommendations by calculating the probabilities for all items in the candidate item set I. We first compute the user's preference score $\hat{z}_i$ for each item v_i in the candidate item set I by a multiplication operation as:

$$\hat{z}_i = z_h^\top e_i, \tag{8}$$

where z_h is obtained by Eq. (7) and e_i is the embedding of item v_i. Then a softmax layer is applied to the preference scores to generate a normalized probability of each item to be recommended as:

$$\hat{y} = \text{softmax}(\hat{z}), \tag{9}$$

where $\hat{z} = (\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_n)$. Finally, the items with the highest scores in $\hat{y}$ will be recommended to the user.

To train our model, we employ the cross-entropy as the optimization objective to learn the parameters as:

$$L(\hat{y}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i), \tag{10}$$

where $y_i \in y$ reflects the appearance of an item in the one-hot encoding vector of the ground truth, i.e., $y_i = 1$ if the i-th item is the target item; otherwise, $y_i = 0$. Finally, we apply the Back-Propagation Through Time algorithm to train SR-IEM.

Table 1: Statistics of the datasets used in our experiments.

Statistics	YOOCHOOSE	DIGINETICA
# clicks	557,248	982,961
# training sessions	369,859	719,470
# test sessions	55,898	60,858
# items	16,766	43,097
Average session length	6.16	5.12

3 EXPERIMENTS

Research questions. (RQ1) Can the proposed SR-IEM model beat the competitive baselines? (RQ2) How does SR-IEM perform compared to the baselines under various session lengths? (RQ3) How does IEM perform on distinguishing the importance of items in a session compared to other importance extraction methods?

Model summaries. We answer our research questions by comparing the performance of SR-IEM against eight baselines for session-based recommendation: (1) Three traditional methods, i.e., S-POP, Item-KNN [8] and FPMC [7]; (2) Five neural models, i.e., GRU4REC [2], NARM [5], STAMP [6], CSRM [10] and SR-GNN [11].

Datasets and parameters. The datasets we use for evaluation are two public benchmark e-commerce datasets, i.e., YOOCHOOSE[1] and DIGINETICA.[2] We use the same preprocessing of the datasets as in [5, 6, 11]. The statistics of the datasets are listed in Table 1. Following [3], we set the maximum session length L to 10, indicating that for long sessions, we only consider the 10 most recent items. The dimension of the item embeddings is set to $d = 200$. We use the Adam optimizer with an initial learning rate 10^{-3} and a decay factor 0.1 for every 3 epochs. The batch size is set to 128 and $L2$ regularization is applied to avoid overfitting by setting as 10^{-5}.

Evaluation metrics. Like [6, 10], we evaluate SR-IEM and the baselines using Recall@N and MRR@N; we set N to 20 in our experiments.

4 RESULTS AND DISCUSSION

4.1 Overall performance

To answer RQ1, we compare SR-IEM to baselines in terms of Recall@20 and MRR@20. The results are presented in Table 2. First of all, for the baselines, we see that the neural models generally outperform the traditional methods. For instance, SR-GNN performs best on YOOCHOOSE in terms of both metrics while it loses against CSRM on DIGINETICA in terms of Recall@20. SR-GNN is able to explore complex transitions of items to generate accurate user preferences by applying a GGNN while CSRM incorporates neighbor sessions on the basis of NARM, leading to better performance than other baselines. Thus, we choose CSRM and SR-GNN for comparisons in later experiments.

Next, we zoom in on the performance of SR-IEM. In general, SR-IEM outperforms all baselines on both datasets in terms of both metrics. For instance on YOOCHOOSE, SR-IEM shows a 2.49% improvement in terms of MRR@20 against the best baseline SR-GNN, which is higher than the corresponding 0.82% improvement in terms of Recall@20. In contrast, on DIGINETICA, the corresponding improvement in terms of Recall@20 is relatively higher than MRR@20. This may be attributed to the size of item set. Thus, SR-IEM is able to boost the ranking of target items for the cases with relatively few candidate items while it is even more effective on hitting the target item for cases with relatively many candidate items.

Table 2: Model performance. The results of the best baseline and the best performer in each column are underlined and boldfaced, respectively. ▲ denotes a significant improvement of SR-IEM over the best baseline using a paired t-test ($p < 0.01$).

Method	YOOCHOOSE		DIGINETICA	
	Recall@20	MRR@20	Recall@20	MRR@20
S-POP	30.44	18.35	21.06	13.68
Item-KNN	51.60	21.81	35.75	11.57
FPMC	45.62	15.01	31.55	8.92
GRU4REC	60.64	22.89	29.45	8.33
NARM	68.32	28.63	49.70	16.17
STAMP	68.74	29.67	45.64	14.32
CSRM	69.85	29.71	<u>51.69</u>	16.92
SR-GNN	<u>70.57</u>	<u>30.94</u>	50.73	<u>17.59</u>
SR-IEM	**71.15▲**	**31.71▲**	**52.35▲**	**17.64**

In addition, we analyze the computational complexity of SR-IEM as well as the best baselines, i.e., CSRM and SR-GNN. For CSRM and SR-GNN, the computational complexity is $O(td^2 + dM + d^2)$ and $O(s(td^2 + t^3) + d^2)$, respectively, where t denotes the session length and d is the dimension of item embeddings. Here, M is the number of incorporated neighbor sessions in CSRM and s is the number of training steps in GGNN. For SR-IEM, the computational complexity is $O(t^2d+d^2)$, which mainly comes from the importance extraction module $O(t^2d + d^2)$ and from the other components $O(d^2)$. As $t < d$ and $d \ll M$ [10], the complexity of SR-IEM is clearly lower than that of SR-GNN and CSRM. To confirm this empirically, we present the training and test times of SR-IEM as well as CSRM and SR-GNN in Table 3. We find that SR-IEM has clearly lower time costs than CSRM and SR-GNN. This indicates that compared to the baselines, SR-IEM can perform best in terms of both recommendation accuracy and computational complexity, making it practicable for potential applications.

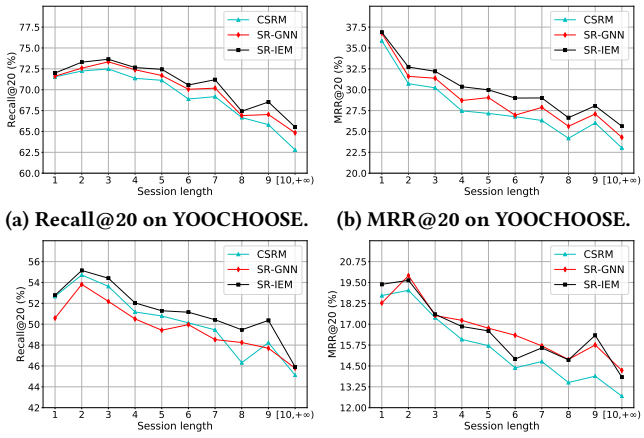
Table 3: Computational complexity and efficiency. We set the training and test time of SR-IEM to 1 unit, respectively. Then, we can find the relative time cost of each corresponding model against SR-IEM.

Method	Complexity	YOOCHOOSE		DIGINETICA	
		training	test	training	test
CSRM	$O(td^2 + dM + d^2)$	4.91	18.62	4.63	19.32
SR-GNN	$O(s(td^2 + t^3) + d^2)$	3.12	2.89	2.56	2.75
SR-IEM	$O(t^2d + d^2)$	1.00	1.00	1.00	1.00

4.2 Impact of session length

To answer RQ2, we plot the results of SR-IEM, CSRM and SR-GNN in terms of Recall@20 and MRR@20 in Fig. 2. As for Recall@20,

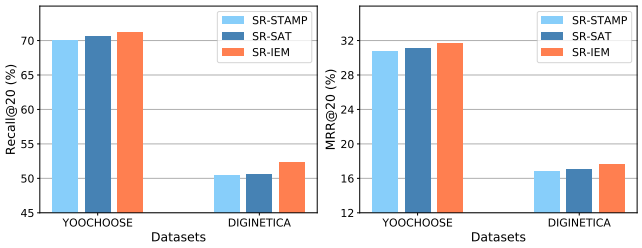
[1] http://2015.recsyschallenge.com/challege.html
[2] http://cikm2016.cs.iupui.edu/cikm-cup



(a) Recall@20 on YOOCHOOSE. (b) MRR@20 on YOOCHOOSE.

(c) Recall@20 on DIGINETICA. (d) MRR@20 on DIGINETICA.

Figure 2: Model performance under varying session lengths.



(a) Recall@20. (b) MRR@20.

Figure 3: Model performance of various importance extraction methods used in our framework on two datasets.

we see that as the session length increases, the performance of the three models on YOOCHOOSE and DIGINETICA increases first and then shows a continuous downward trend. The improvement of SR-IEM over CSRM and SR-GNN is more obvious for sessions lengths 4–7 than for lengths 1–3. When the session length is too short, IEM is not able to distinguish the item importance very well. As the length increases, the effectiveness of IEM goes up.

For MRR@20, all models display a consistent downward trend on YOOCHOOSE and DIGINETICA as the session length increases. SR-IEM outperforms CSRM and SR-GNN at all lengths on YOOCHOOSE while losing to SR-GNN for some cases on DIGINETICA, e.g., at lengths 4 and 5. In addition, on both two datasets, the MRR@20 scores show a sharper decrease than the Recall@20 scores. The difference in Recall@20 and MRR@20 scores on the two datasets may be due to the fact that non-relevant items have a bigger negative impact on MRR@20 than on Recall@20.

4.3 Analysis on importance extraction module

To answer RQ3, we replace IEM in SR-IEM with two alternatives and make comparison. We denote the variant models as (1) SR-STAMP: replace IEM with an attention mechanism proposed by [6]; here the mixture of all items and the last item in session is deemed as "query." (2) SR-SAT: utilize a self-attention mechanism [9] to distinguish the item importance, and then aggregate them using an average pooling strategy [12]. The results are shown in Fig. 3.

In general, SR-IEM achieves the best performance in terms of Recall@20 and MRR@20 on both datasets. SR-SAT outperforms SR-STAMP. This could be due to the fact that SR-SAT considers the item-item relationship in a session by modeling the contextual signal, which helps to capture user's preference for generating correct item recommendations. However, SR-STAMP only takes a mixture of all items and the last item to determine the item importance, thus failing to accurately represent user's preference. In addition, it is difficult for both SR-SAT and SR-STAMP to eliminate non-relevant items in a session, which results in a negative effect on the recommendation performance. In contrast, the proposed IEM module can effectively locate important items and assign a relatively high weight to them for user preference modeling, in a way that avoids being distracted by other items in the session.

5 CONCLUSIONS AND FUTURE WORK

We have proposed an Importance Extraction Module for Session-based Recommendation (SR-IEM) that incorporates a user's long-term preference and his current interest for item recommendation. A modified self-attention mechanism is applied to estimate item importance in a session for modeling a user's long-term preference, which is combined with user's current interest indicated by the last item to produce the final item recommendations. Experimental results show that SR-IEM achieves considerable improvements in terms of Recall and MRR over state-of-the-art models with reduced computational costs compared to competitive neural models. Next, we would like to apply the Importance Extraction Module to other tasks, e.g., dialogue systems and question answering.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China under No. 61702526, the Defense Industrial Technology Development Program under No. JCKY2017204B064, and the Innovation Center for AI (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Xiangnan He, Kuan Deng, Xiang Wang, et al. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. *arXiv preprint arXiv:2002.02126* (2020).
[2] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, et al. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR'16*.
[3] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM'18*. 197–206.
[4] Chenliang Li, Xichuan Niu, Xiangyang Luo, et al. 2019. A Review-Driven Neural Model for Sequential Recommendation. In *IJCAI'19*. 2866–2872.
[5] Jing Li, Pengjie Ren, Zhumin Chen, et al. 2017. Neural Attentive Session-based Recommendation. In *CIKM'17*. 1419–1428.
[6] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, et al. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *KDD'18*. 1831–1839.
[7] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW'10*. 811–820.
[8] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, et al. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW'01*. 285–295.
[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. 2017. Attention is All you Need. In *NeurIPS'17*. 5998–6008.
[10] Meirui Wang, Pengjie Ren, Lei Mei, et al. 2019. A Collaborative Session-based Recommendation Approach with Parallel Memory Modules. In *SIGIR'19*. 345–354.
[11] Shu Wu, Yuyuan Tang, Yanqiao Zhu, et al. 2019. Session-Based Recommendation with Graph Neural Networks. In *AAAI'19*. 346–353.
[12] Shuai Zhang, Yi Tay, Lina Yao, et al. 2018. Next Item Recommendation with Self-Attention. *arXiv preprint arXiv:1808.06414* (2018).