2022 IEEE 5th International Conference on Computer and Communication Engineering Technology (CCET) | 978-1-6654-6735-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CCET55412.2022.9906367

A Self-Attentive Interest Retrieval Recommender

Min Wu
School of Software Engineering
Xi'an Jiaotong University
Xi'an, China
e-mail:
epouxdefutaba@stu.xjtu.edu.cn

Chen Li*
School of Software Engineering
Xi'an Jiaotong University
Xi'an, China
e-mail: lynnlc@126.com

Lihua Tian
School of Software Engineering
Xi'an Jiaotong University
Xi'an, China
e-mail: lhtian@xjtu.edu.cn

***Abstract*—Thanks to the attention mechanism, self-attention networks (SANs) have been widely used in sequential recommendation. However, most existing SANs approaches still follow an old fashion generating one single embedding as final representation, which constraints model's capacity. To enrich this kind of representation, sequential recommender uses metadata such as item category to capture user's multi-interests. But this method will not reach its expectation due to item's long-tail property. This property will result a large constant of category cannot be effectively activated by the lack of interaction records. Another drawback is that may also lead to over-parameterization caused by the massive categories. Particularly, we propose a Self-Attentive Interest Retrieval network (SAIR) to explore a context-aware representation from user's behaviors while not fall into over-parameterization. SAIR works in a typical SANs manner, encode the behavior sequence using self-attention, and we propose an interest retrieval module to project the sequences to an interest relevance distribution adaptively. And we leverage an interest-to-interest interaction to generate several context-aware interests embeddings. Then we fuse multi-interest embeddings as final output. Extensive experiments are carried out on three real-world datasets, the results demonstrate that SAIR outperforms other SANs methods and other state-of-the-art algorithms in multiple evaluation metrics.**

Keywords-deep learning; recommendation systems; self-attention networks; information retrieval

I. INTRODUCTION

Recommendation systems have become an indispensable part of the Internet era which are widely used in advertising, e-commerce, social media and other fields. Its core function is to infer how likely user will interact with candidate items base on past actions. The method of click-through rate prediction (CTR) is always the mainstream solution in the recommendation field. It takes advantage of deep network to achieve an automatic feature interaction. However, with the development of recommender, this modeling cannot meet a higher requirement of complex sequential scenarios, because CTR models user representation based on a simple input of item and user.

Therefore, more and more recommenders treat the recommendation as a sequential problem. There are traditional models based on Markov chain, deep models such as GRU4Rec [1], Caser [2], NextItNet [3]. And self-attention networks (SANs) such as SASRec [4], BERT4Rec [5], LightSANs [6]. Undoubtedly, modeling the recommendation problem in a sequential manner is more in line with the real-world scenarios.

However, the majority practice of sequential recommenders is still suffered the vulnerability of a single output representation. This is actual a static modeling of user preferences, which fails to capture user's multi-interests. To explore a multi-interest embedding, SHAN [7] takes advantage of attention mechanism to model the sequence twice to retrieve both long-term and short-term semantics; and CoCoRec [8] makes use of item category to divide sequence to different subsequences to learn the in-category transition pattern. Their research underlines that a sequence may represent a various semantics. Thus, the sequential recommendation should be context-aware to promote model's capacity and to capture user's multi-interests.

Despite their best efforts, they all have limitation in some perspectives. In a way like SHAN can maintain a balance between long-term and short-term semantics, but such consideration still lacks the diversity of representation for long sequence. Besides, like CoCoRec, the usage of metadata makes good use of item category, but it suffers from unreliable sources of metadata, the absence of information, and the over-parameterization in large-scale systems.

To tackle these issues, we propose a Self-Attentive Interest Retrieval network (SAIR). Our method encodes the user behavior sequence in a typical SANs manner. Then we introduce an interest-retrieval module. Our module can adaptively project stacked sequences to interest relevance distribution and conduct an interest-to-interest interaction among sequences. Finally, we apply an aggregation module to integrate multiple embeddings on account of preference to interests. SAIR makes good use of sparse modeling to enrich user's representation and overcome the vulnerability of over-parameterization, the interaction over interests also guarantees a context-aware output.

Our work will be introduced in the following structure. Section 2 will introduce the modeling method of sequential recommendation and multi-interest embedding. Section 3 will introduce our multi-interest modeling method and interest fusion method proposed in this paper. Section 4 will introduce the overall comparison with other baseline. The final section 5 will summarize the contributions of this work and give future experimental directions.

II. Related Works

With its ability to model continues items, sequential recommender is the most important method in modern recommendation systems. In this field, Recurrent neural networks are used for its impressive capacity in modeling sequences, but RNN-based methods have been criticized for their parallelism, and they are easier to lose previous information in long sequence. Among convolution works, Caser migrates the practice of image processing to model the sequence embedding in an 'image' way, and also explores to capture user's skip behavior. But variable-length item sequence will undoubtedly affect the expression of convolutional networks.

In recent years, the extensive researches of attention mechanism show its potential for sequential problem. SASRec [4] first applies self-attention to sequential recommendation, it adaptively learns the weights between items through an item-to-item interaction. Attention is initially proposed in computer vision, and has become popular in recent years with widespread applications for pre-train methods like Transformer and BERT. The main idea is that when human beings observe objects, they tend to focus on local hotspots instead of the entire image. This item-to-item interaction ensures there is no obvious loss in the learning from different position, and the attention calculation satisfy the necessary interaction of different features. Therefore, various sequential recommender uses self-attention mechanism to complete the perception of contextual information. For instance, Linformer [9], Transformer4Rec, LightSANs and other methods have appeared. It has been proved that the SANs method has become a novel fashion in recommendation practice.

For modeling user's interests, lots of prior works make their efforts. Traditional CTR prediction models, DIN [10] and DIEN [11] have tried to model the sparse interests of users. Among sequential recommenders, CoCoRec proposes a way to split the item sequence into subsequences according to the item classification. SINE [12] has also attempted to model an item-to-sparse interest relation with multiple conceptual prototypes, but it models merely user's general interest and fail to reveal the interest shift pattern. Besides, [13] argues that a single embedding vector is difficult to meet the multi-semantic problem in a real-world scenario. Therefore, they proposed an embedding method that builds semantic vectors of different concepts for each item. This method obviously enriches the model's capacity to dozens of times, but it also increases size of item's embedding looking-up table dozens of times. However, the above series of methods still well enlighten us the importance of well model the sparse representation of user intents.

III. Proposed Method

Our algorithm proposed is mainly based on the SANs structure for extraction of sequence information. Given user sequence $X^{(u)} = [x_1^{(u)}, x_2^{(u)} \dots x_n^{(u)}]$, our model will output the user's interest representation for next item prediction. The motivation of interest retrieval is to promote the accuracy on next item prediction by enriching the user's representation. But the straight-forward way like modeling with item metadata will suffer from massive item categories, which leads to some poor category representations on absence of sufficient training data. The way like LightSANs and Linformer give a rather acceptable solution. They assume that the user's historical items can be categorized in a small count of latent interests. Therefore, modeling the sequence into a fixed dimension of latent interests is a naturel way to tackle above issues. Based on such assumption, we assume that all items divided by interest will not exceed L latent interests, and for interest retrieval we also assume that the user's preference will not fall into more than K latent interests at instant time, and surely K is smaller than L.

The overall framework of SAIR is shown in Figure 1. The solid line is used to connect the input and output of different layers, and the dotted line after the sequence encoder indicates that the sequence matrix is analyzed in form of a single vector. The network takes the user sequence composed of item-id as the input. Our workflow consists of three modules, firstly we get item embeddings with an embedding layer, then we encode the item representation to sequence embedding using self-attention mechanism. Next, the interest retrieval module will finish the interest retrieval and selection. Finally, the interest fusion module aggregates the selected interests according to user's preferences and outputs a final interest representation.

Below we will introduce each portion in detail.

A. Self-Attention Layers

We regard $\{X^{(u)}\}_{u=1}^{N}$ as an ordered sequence of historical items of user u, where $X^{(u)} = [x_1^{(u)}, x_2^{(u)} \dots x_n^{(u)}]$, n is the number of items that the user interacts with, and this sequence is in time order. $X^{(u)}$ is the input of the network, we firstly get in encoded through an embedding layer. In this process, we involve item category and absolute position. That is typical embedding look-up method, and output $E^{(u)}$,

$$E^{(u)} = H_{id}(X^{(u)}) + H_{cate}(X^{(u)}) + H_{pos}(X^{(u)}). \quad (1)$$

Function H with different subscript means different embedding layers, corresponding to item id, item category and absolute position respectively. Where $H_{id} \in \mathbb{R}^{(MxD)}$, $H_{cate} \in \mathbb{R}^{(TxD)}$, $H_{pos} \in \mathbb{R}^{(NxD)}$.

Each vector in embedding matrix $E^{(u)}$ is now an item representation. And we follow the practice like SASRec, using a scaled self-attention to model item representation into sequence representation. The equation is,

$$\hat{E} = \mathrm{softmax}\left(\frac{QK}{\sqrt{D}}\right)V \quad (2)$$

where $Q = E^{(u)}W^Q$, $K = E^{(u)}W^K$, $V = E^{(u)}W^V$. And we activate $\hat{E}$ with feed-forward function $F = \mathrm{FFN}(\hat{E})$, where $\mathrm{FFN}(\cdot)$ denotes as

$$\mathrm{LayerNorm}(\mathrm{Linear}(\mathrm{Relu}(\mathrm{Dropout}(\mathrm{Linear}(\cdot))))). \quad (3)$$

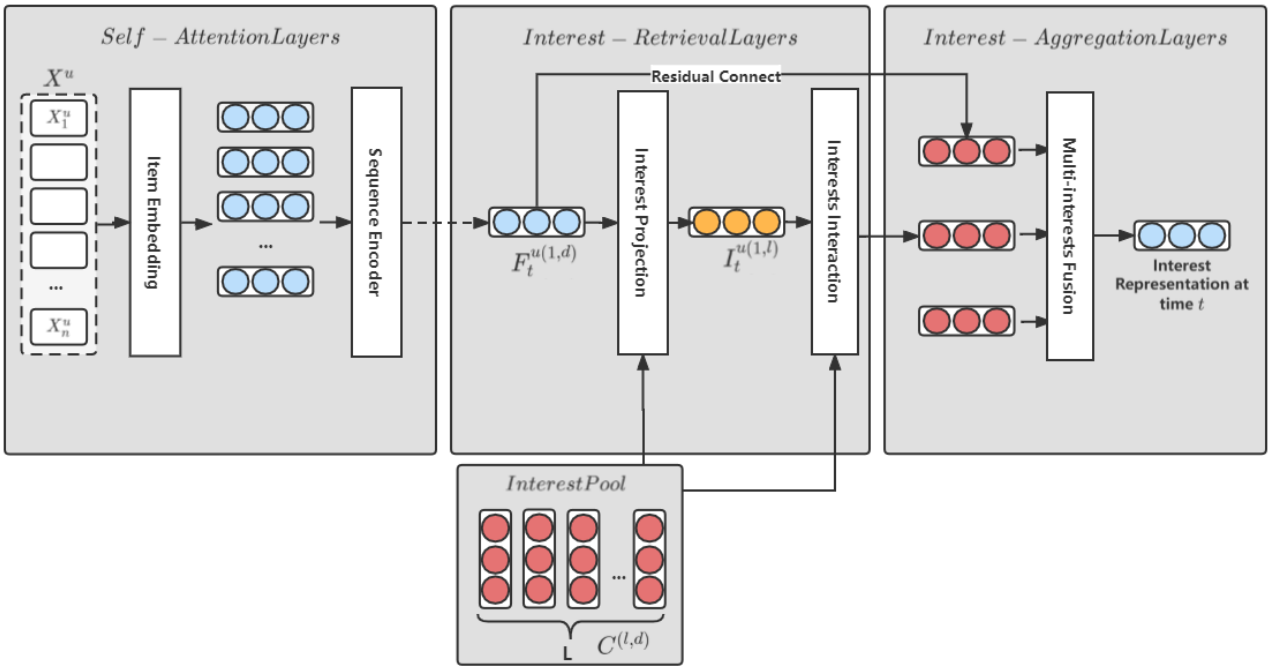

Figure 1. Overall framework of SAIR

After this layer of self-attention, we make the conversion of $\widehat{E} \rightarrow F$. A single vector in matrix F at position t is now represent a compress sequence information.

B. *Interest Retrieval Layer*

With the compressed sequences representation $F \in \mathbb{R}^{(NxD)}$, it now involves how to extract user's interests. To achieve this goal, we propose a learnable projection function and an interest interaction layer in this module.

Enlighten by Linformer and LightSANs [9, 6], we propose a learnable projection function after our self-attention layer to generate a relevance distribution of encoded sequences and latent interests. First, we assume that the user will have no more than L latent interests, so we choose to project each vector on hidden unit dimension which means D different dimension will be reduced to L, the matrix shape change from $(N \times D)$ to $(N \times L)$. And here we introduce our interest pool matrix $C \in \mathbb{R}^{(LxD)}$ to realize this interest projection. We can compute the sequence-to-interest relevance distribution $C_a \in \mathbb{R}^{(NxL)}$ as

$$C_a = F \cdot C^T . \quad (4)$$

$C_a = [C_a^1, C_a^2 \dots C_a^n]$, where C_a^t represents the user sequence's preference score on L interests at time t, and each hidden unit is the distance calculated by inner product of the corresponding sequence vector and the latent interest vector.

Despite our modeling of sparse interests looks like quite similar with prior work SINE [12], we both make use of sparse conceptual prototypes. However, our difference lies in two aspects: 1) SINE models K interests base on a general representation of sequence. But our method applies interest projection based on stacked sequences which means we have K interest embeddings at each position. 2) SINE leverage an item-to-interest interaction with its set of prototypes to calculates the contribution of each item to each interest. And the output is based on item-id embedding. But ours explore an interest-to-interest interaction base on a distribution matrix to fully perceive migration of user's interests.

Unlike previous work [13] model sequence with all semantic concepts, which makes each sequence embedding has too many vectors to represent massive concept. However, this kind of method falls into a suboptimum solution, cause user only focuses on certain interests instead of all at a certain moment. Thus, our method retrieves user's interest with an interaction layer. To capture the latent pattern of interest migration, we firstly apply another self-attention to the relevance distribution matrix C_a. The equation is:

$$\widehat{C_a} = \text{Attention}(C_a W^Q, C_a W^K, C_a W^V). \quad (5)$$

And in order to eliminate the noise of unimportant interests and maintain a small user focus, we retrieve K most valuable interests from the user preference matrix $\widehat{C_a}$. That is,

$$C_k = \text{top}(\widehat{C_a}, k). \quad (6)$$

The vector in $C_k \in \mathbb{R}^{(NxK)}$ consist of the indices of interests selected from different sequences, and we can then generate user multi-interest embedding by look-up into our interest pool, the parameter matrix C. That is,

$$I_k = \text{Embedding}(C_k, C) \quad (7)$$

where $I_k = [I_k^1, I_k^2 \dots I_k^N]$, $I_k^t \in \mathbb{R}^{(KxD)}$represents the user's k preference interest vectors at time t. With this interest retrieval module, we have converted the original sequence

representation to the interest representation by the transformation from $F \rightarrow I_k$.

C. Interest Aggregation Layers

In fact, the k sparse representations extracted only come from a rather small parameter matrix C. And its purpose is to express L latent embeddings as sparse as possible instead of directly representing the final sequence information. Meanwhile, we consider the item-id as the most valuable info that cannot be dropped. Under these two concerns, we let the k interest representations residually connected by the sequence representation. That is

$$\widehat{I_k^t} = \text{Residual}(I_k^t,\ F^t) = \left[I_k^{t(1)} + F^t,\ I_k^{t(2)} + F^t, \dots, I_k^{t(K)} + F^t\right]. \quad (8)$$

For activation, we activate the k interest embeddings with its own feed-forward networks respectively like what multi-head attention did. That is, $I_{active} = \text{FFN}(\widehat{I_k})$. In this way, the k interest representations at time t after activation are denoted as $I_{active}^t = \left[I_a^{t(1)}, I_a^{t(2)}, \dots, I_a^{t(k)}\right]$.

To aggregate interest embeddings. We make full usage of the interest relevance distribution C_a. The value of hidden unit is the relevance score represents user's preference over sparse interests. Thus, we apply softmax operation on these selected interests. And the weight matrix denotes as

$$W_i^t = \frac{\exp(c_a^i)}{\sum_j^{C_k} \exp(c_a^j)}, i \in [1, k]. \quad (9)$$

And we calculate the user finally fused representation with this weighted sum equation (10) to get $e_u^t \in \mathbb{R}^D$.

$$e_u^t = \sum_{i=1}^{k} W_i^t \cdot I_a^{t(i)} \quad (10)$$

D. Loss Function

SAIR follows the practice like [4, 12] to train the model. Given the sequence $X^u \in \mathbb{R}^N$ of a specific user u, our model generates the representation matrix $E^u \in \mathbb{R}^{(NxD)}$ after calculation. We use the inner product of embeddings used as the prediction score, so our prediction function is written as

$$P(x_u^t | x_u^1, \dots, x_u^{t-1}) = \langle \text{Item}_{\text{label}},\ e_u^t \rangle. \quad (11)$$

During training, we sample one negative label randomly for each positive label, and we use the cross-entropy loss function to evaluate the prediction loss is

$$\mathcal{L}_{ce} = -\sum_{X^u \in X} \sum_{t \in [1,2,\dots,N]} \left[\log(\langle \text{Item}_{\text{lab}}, e_u^t \rangle) + \log(1 - \langle \text{Item}_{\text{neg}}, e_u^t \rangle)\right]. \quad (12)$$

In addition, to optimize the expression of C. DecovLoss [18] is introduced to calculate the loss of interest matrix C. DecovLoss is designed to adjust the parameter distribution by calculating the cross-covariance between the hidden units. We denote this loss $\mathcal{L}_d = \frac{1}{2}\left(\|Z\|_F^2 - \|\text{diag}(Z)\|_2^2\right)$, where $Z = \frac{1}{D}(C - \bar{C})(C - \bar{C})^T$, C is the interest pool matrix, $\bar{C}$ is its mean matrix of C.

The final loss function is written as $\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_d$, as the objective function of our model.

IV. Experiment and Analysis

A. Experiment Environment

In comparison with SOTA, we consider the baseline from the SANs method and general method in the sequential recommendation. We implement all network using an open-source recommendation framework RecBole.

B. Dataset and Implement Details

Our choice focus on dataset with high sparsity. We use three real-world benchmark datasets, including the movielens dataset at two scales and the anime dataset from animation websites. Statistics of datasets is shown in Table I.

For the preprocessing of the dataset, all interaction records with a non-negative rating are regarded as valid data. Meanwhile, the category information is retained, and the items missing category are set to the [UNK] category.

We apply leave-one-out strategy for evaluation Additionally, a uniform negative sampling is performed on all datasets for training. We employ the Sample-softmax strategy for validation and testing, sampling number is set to 100.

For evaluation criteria, normalized discounted cumulative gain (NDCG@K) and hit ratio (HR@K) are applied. And K is taken from [5; 10; 20].

C. Baseline Model

We choose the competitor from the SANs method and other state-of-the-art recommenders. The following is a brief introduction to the competitor baseline:

1). SASRec [4], the first application of attention on sequential recommender after the attention mechanism has been widely used.

2). BERT4Rec [5] argue the regular treatment of sequence from left to right has its limitation. Thus, it models sequence with a bi-direction encoder.

3). Caser [2] treat the user sequence matrix as an image. It models sequence with two convolutional layers. And Caser leverage a skip behavior to learn a latent sequential pattern.

4). LightSANs [6] improves the original self-attention method by reducing the dimension of Q and K through linear transformation before the attention calculation. So, it developpe the item-to-item interaction of self-attention to interest-to-interest interaction.

TABLE I. Statistics of the Datasets

Dataset	*ML-1m*	*ML-20m*	*Anime*
#User	6041	138494	73515
#Items	3883	26745	11200
#Actions	1000038	20000263	7813737
#Category	18	21	82
#Avg.Length	163.4	144.41	96.51
#Sparsity	95.58	99.46	99.05

TABLE II. OVERALL COMPARISON OF PERFORMANCE ON NDCG@K AND HR@K(%). THE BEST PERFORMANCE AND SECOND PERFORMANCE ARE DENOTED IN BOLD AND UNDERLINED RESPECTIVELY

Dataset	*Metrics*		*BERT4Rec*	*SASRec*	*Caser*	*LightSANs*	*SAIR*	*.Impv*
ML-1M	NDCG	@20	0.5996	0.6112	0.5879	0.6145	**0.6213**	1.1%
		@10	0.5779	0.5968	0.5754	0.5945	**0.6088**	2%
		@5	0.5466	0.5582	0.541	0.5621	**0.5732**	1.9%
	HR	@20	0.8664	0.9035	0.8724	0.8828	**0.9107**	0.8%
		@10	0.7811	0.827	0.7906	0.8046	**0.8334**	0.8%
		@5	0.6843	0.7461	0.6889	0.7045	**0.754**	1%
ML-20M	NDCG	@20	0.7539	0.7567	0.7442	0.7893	**0.8176**	3.6%
		@10	0.7433	0.7496	0.7378	0.782	**0.8126**	3.9%
		@5	0.723	0.7298	0.7059	0.7618	**0.798**	4.7%
	HR	@20	0.9485	0.9622	0.943	0.9889	**0.9908**	0.1%
		@10	0.9058	0.9349	0.9254	0.9641	**0.9713**	0.7%
		@5	0.8445	0.8746	0.8816	0.9091	**0.9269**	1.9%
Anime	NDCG	@20	0.5799	0.7803	0.7537	0.8348	**0.8532**	2.2%
		@10	0.5611	0.7753	0.7363	0.8305	**0.8501**	2.4%
		@5	0.5172	0.7601	0.7294	0.8196	**0.8413**	2.6%
	HR	@20	0.9229	0.9695	0.9535	0.9647	**0.9854**	1.6%
		@10	0.8494	0.95	0.9417	0.9481	**0.9734**	2.4%
		@5	0.7152	0.9043	0.8839	0.9146	**0.9466**	3.5%

D. Experiment Result

1) Overall comparison

For a fair comparison, the max length of input sequence is fixed to 150 for all models. We apply Adam optimizer with the learning rate fixed as 0.001. If the training procedure requires negative sampling, we use a uniform negative sampling. The size of hidden unit is fixed to 64, and we use a mini-batch of 256 to train all models. Our best model selector is NDCG@10, early-stop is set to 20 epochs. For our model SAIR, our crucial hyper-parameters are set as follows, the size of interest pool L is 10 and the number of interests K is 3 for ML-1m dataset; L is 20 and K is 8 for ML-20m dataset; L is 30 and K is 10 for Anime dataset.

The overall performance comparison is shown in Table II. We can observe that our proposed method SAIR is exceed a lot in all evaluation metrics compared with all baselines. This is benefited from SAIR's modeling user's diverse interests. In this way, we enrich user's representation with several interest embeddings. Especially, SAIR has achieved a greater improvement than SASRec on datasets with more item categories. For instance, the Anime dataset has nearly 100 categories and long sequence length of interactions. With interaction sequence get longer the recommendation scenario become more complex. SAIR is rather suitable for such large scaled dataset. And for NDCG indicator, the promotion of our method tends to be increased with the decrease of @K value. This trend shows both in ML-20m and Anime datasets, while NDCG is sensitive with the order of recommended items. This reveal that SAIR's recommendation results are more likely to find user's real interest item and place them at the top of recommendation list.

Especially compared with SASRec, the backbone of SANs method, SAIR gain a meaningful improvement. As on anime dataset, SAIR achieve an improvement of 9.6% (NDCG@10) compared to SASRec. SAIR performs a sequence-to-interest relevance distribution projection based on the extracted sequence information. And SAIR models the interest interaction with the attention mechanism and top-K operation. As a result, SAIR generate a much more powerful representation by aggregate diverse interest embeddings. As well, our method is also an enhancement to simple item-id representations. The empirical experiments have proved the advantages of SAIR over SASRec on multiple datasets. The modeling of diverse interests has effectively improved the representation ability of the algorithm, especially on larger and more complex datasets.

2) Ablation study

To verify the effectiveness of proposed modules. We introduce two variants SAIR-label and SAIR-full. SAIR-label models the interests with item categories by replacing our interest pool. SAIR-full remove the interest interaction layer to model directly with all prototypes. Both variants are observed with a performance drop, while SAIR-full has a drop of 4.3% on Anime dataset. This phenomenon suggests our interest retrieval module helps to eliminate the noises while modeling user's multiple interests. We also find performance degrades further with a more complex dataset.

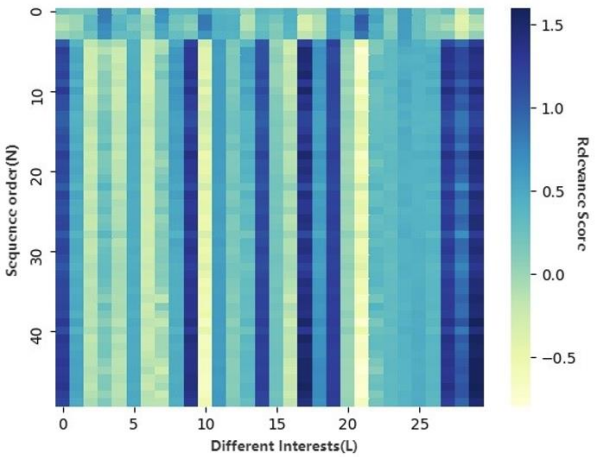

Figure 2. Heat map of distribution matrix.

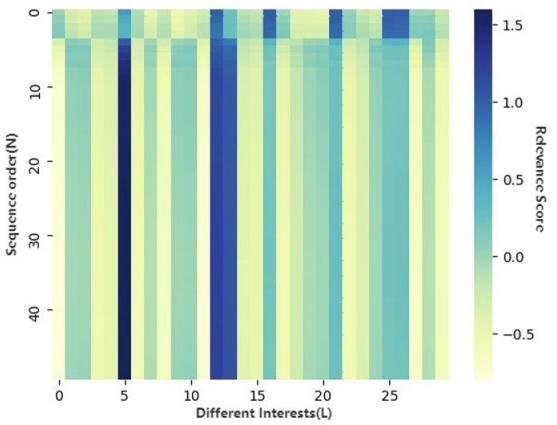

Figure 3. Heat map of distribution processed by interaction layer.

And a case study is carried out to prove the effectiveness of interaction in a more intuitive way. Shown as Figure 2 and Figure 3, they are heat maps of distribution matrix before the interaction and after the interaction. Obviously, Figure 3 has less noise compared to Figure 2, and user's interest distribution is more concentrated after the interest interaction. Besides, we also observe an obvious shift of interests in the first few items, that suggests our method can capture the migration of interest in long sequence.

V. CONCLUSION

In this paper, based on the SANs approach, we propose a novel recommender SAIR that introduce attention mechanism to model diverse interests from sequences. Compared with general approaches, we input additional item classification information into the embedding layer for producing a context-aware representation. For interests modeling, we design two modules to accomplish the interest retrieval from sequences and the integration of the multiple interest embeddings. Our methodology successfully improves user's representation by modeling user's multi-interests. And our method is resilient to over-parameterization. With empirical experiments on three real-world datasets, SAIR obtains an improvement on all evaluation metrics compared to state-of-the-art methods.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under Grant No.61901356 and the HPC Platform of Xi'an Jiaotong University.

REFERENCES

[1] HIDASI, Balázs, et al. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939, 2015.

[2] TANG, Jiaxi; WANG, Ke. Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the eleventh ACM international conference on web search and data mining. 2018. p. 565-573.

[3] YUAN, Fajie, et al. A simple convolutional generative network for next item recommendation. In: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining. 2019. p. 582-590.

[4] KANG, Wang-Cheng; MCAULEY, Julian. Self-attentive sequential recommendation. In: 2018 IEEE International Conference on Data Mining (ICDM). IEEE, 2018. p. 197-206.

[5] SUN, Fei, et al. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM international conference on information and knowledge management. 2019. p. 1441-1450.

[6] FAN, Xinyan, et al. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021. p. 1733-1737.

[7] YING, Haochao, et al. Sequential recommender system based on hierarchical attention network. In: IJCAI International Joint Conference on Artificial Intelligence. 2018.

[8] CAI, Renqin, et al. Category-aware collaborative sequential recommendation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021. p. 388-397.

[9] Wang, S., Li, B. Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-attention with linear complexity. arXiv preprint arXiv:2006.04768.

[10] ZHOU, Guorui, et al. Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. 2018. p. 1059-1068.

[11] ZHOU, Guorui, et al. Deep interest evolution network for click-through rate prediction. In: Proceedings of the AAAI conference on artificial intelligence. 2019. p. 5941-5948.

[12] TAN, Qiaoyu, et al. Sparse-interest network for sequential recommendation. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining. 2021. p. 598-606.

[13] Liu, N., Tan, Q., Li, Y., Yang, H., Zhou, J., & Hu, X. (2019, July). Is a single vector enough? exploring node polysemy for network embedding. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 932-940).

[14] COGSWELL, Michael, et al. Reducing overfitting in deep networks by decorrelating representations. arXiv preprint arXiv:1511.06068, 2015.