2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

Co-refining User and Item Representations with Feature-level Self-attention for Enhanced Recommendation

Zikai Guo*, Deqing Yang*‡, Baichuan Liu*, Lyuxin Xue*, Yanghua Xiao†
‡Corresponding author, *School of Data Science, †School of Computer Science, Fudan University, Shanghai, China
Email: *{zkguo18,yangdeqing,bcliu20,lxxue19}@fudan.edu.cn, †shawyh@fudan.edu.cn

***Abstract*—Self-attention mechanism is primarily designed to capture the correlation (interaction) between any two objects in a sequence. Inspired by self-attention's success in many NLP tasks, some researchers have employed self-attention in sequential recommendation to refine user representations by capturing the correlations between the historical interacted items of a user. However, the user representations in previous self-attention based models are not flexible enough since the self-attention is only applied on user side, restricting performance improvement. In this paper, we propose a deep recommendation model with feature-level self-attention, namely *SAFrec*, which exhibits enhanced recommendation performance mainly due to its two advantages. The first one is that SAFrec employs self-attention mechanism on user side and item side simultaneously, to co-refine user representations and item representations. The second one is that, SAFrec leverages item features distilled from open knowledge graphs or websites, to represent users and items on fine-grained level (feature-level). Thus the correlations between users and items are discovered sufficiently. The extensive experiments conducted over two real datasets (NetEase music and Book-Crossing) not only demonstrate SAFrec's superiority on top-n recommendation over the state-of-the-art deep recommendation models, but also validate the significance of incorporating self-attention mechanism and feature-level representations.**

***Index Terms*—recommender system, self-attention, feature embedding, deep learning**

I. Introduction

In recent years, encouraged by the power of deep neural networks (DNNs for short) in computer image, visions and natural language processing (NLP for short), many researchers have also imported DNNs into recommender systems [1]–[3] to improve recommendation performance. In many deep recommendation models, attention mechanism is widely adopted to refine user representations [4]–[6], thus more precise recommendation is gained.

More recently, the *Transformer* [7] built with *self-attention* which is an advanced version of attention mechanism, has exhibited perfect performance in machine translation. Inspired by it, some researchers also imported self-attention mechanism into sequential recommendation models [8]–[11] to enhance recommendation performance. The attentions in these models are designed to capture the correlations (sequential pattern) between different items in a user's historical interaction sequence, based on which user representations are refined and thus recommendation performance is improved. Although these sequential recommendation models with self-attention have been proven effective, they still have a great room for improvement due to the following issues.

The first issue is that, some self-attention based recommendation models [8], [9] only focus on the item-level sequential patterns, neglecting the correlations between item features which are beneficial to specify a user's fine-grained preference. Specifically, each item in those models is first represented by a single embedding, and then each user is represented by the union constituted by the embeddings of the user's historical interacted items. In fact, such item-level representations do not reveal the correlations/similarities between items sufficiently. As we know, seeking latent relationships between different items in terms of features is the most important principle for many users to filter their favorite items. For example, some users prefer the songs sung by their idols, and some users like to watch the movie of a certain genre. Such item correlations (having the same singer or genre) can be discovered with fine grain if items are represented on feature-level.

The second issue is that, all previous recommendation models with self-attention only employ self-attention mechanism on the user side, causing the limitation of performance improvement. Although some newly proposed models [11], [12] have focused on capturing the feature-level sequential patterns, they still employ self-attention to only refine user representations as previous models [8], [9]. Such operations result in that a user's representation is fixed since his/her historical items are fixed, no matter what candidate items are confronted. Notably, previous attention-based models [4]–[6] have justified that, more precise recommendation results could be obtained if user representations are adjusted with respective to different candidate items.

To address above issues, in this paper we propose a deep recommendation model *SAFrec* towards top-n recommendation of implicit feedbacks [4], [13], [14], which is built with the self-attention mechanism on feature-level representations. On one hand, to capture the correlations between a user's historical items on feature-level, and then generate fine-grained user representations, we first distill item features, also known as *side information*, from open *knowledge graphs* (KGs for short) or websites. Then, each item in SAFrec is represented based on its feature embeddings. One the other hand, SAFrec employs self-attention module on the target user side and the candidate item side simultaneously, which co-refines user representations and item representations. The attentions in SAFrec reflect a user's preference in terms of features

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00

when filtering his/her favorite items. Compared with previous recommendation models with self-attention, SAFrec's self-attention not only captures the relevance between different historical interacted items of a target user, but also captures the correlations between the features of the user and the candidate item. Our experiments justified that such sufficient correlations are useful for achieving more precise top-n recommendation. Furthermore, to improve the effectiveness of SAFrec's self-attention further, we also adopt *multi-head* attention and *batch normalization*.

In summary, we have the following contributions in this paper:

1. We propose a deep recommendation model which employs feature-level self-attention mechanism on both user side and item side to co-refine user representations and item representations, resulting in better recommendation performance than previous self-attention based models.

2. Our work validates that self-attention mechanism is not only competent for sequential tasks such as sequential recommendation, but also can be applied for top-n recommendation which is a more generalized recommendation task.

3. Our extensive experiments not only demonstrate SAFrec's superiority over the state-of-the-art deep models including previous recommendation models, but also justify the rationality of incorporating feature-level self-attention for robust recommendation performance.

The rest of this paper is organized as follows. We elaborate the details of our model in Section 2, followed by our experiment results in Section 3. We introduce related work in Section 4 and conclude our work in Section 5.

II. Methodology

In this section, we introduce our proposed model in detail. Related notations mentioned in the following texts are listed in Table I. In general, we use a bold uppercase to represent a matrix or a cube (tensor), and a bold lowercase to represent a vector.

TABLE I
NOTATIONS.

notation	description
u	the target user
v	the candidate item
$\boldsymbol{e}$	item feature embedding (vector)
$\boldsymbol{U}^0$	original user representation (cube)
$\boldsymbol{U}$	user representation (matrix)
$\boldsymbol{I}$	item representation (matrix)
L	the number of a user's recent interacted items
M	feature number of an item
Z	the number of attention heads
$\boldsymbol{X}, \boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$	representation matrices
$D, D_{K/V}$	embedding dimension
$\boldsymbol{A}$	attention map (matrix)
a_{ij}	affinity score between feature i and feature j
$\boldsymbol{W}^{Q/K/V/N}$	weight matrix
$\boldsymbol{S}_i$	weighted output matrix of the i-th attention head
$\boldsymbol{o}$	output vector of self-attention layer
σ	Sigmoid function
$\hat{y}_{uv}$	predicted probability that user u likes item v

A. Recommendation Task

In this paper, we focus on the top-n recommendation based on users' *implicit feedbacks* [4], [13], [14] which indicate users' preferences for items. In other words, an observed interaction (review or rating) between a user u and an item v is identified as a u's feedback of value 1 reflecting u's interest on v. The unobserved interaction between u and v is identified as a u's feedback of value 0 which does not necessarily imply u dislikes v, because u may not aware of v at all. The objective of our recommendation task can be described as that, given a target user u, the model should recommend top-n items to u from candidate items based on item features and u's preference inferred from u's historical interacted items. To achieve this task, our model tries to compute the probability that u likes the candidate item v, i.e., $\hat{y}_{uv}$. According to $\hat{y}_{uv}$ for each candidate item, u's top-n recommendation list is generated.

B. Model Overview

The framework overview of SAFrec is depicted in Fig. 1, which consists of three layers, i.e, embedding layer, self-attention layer and prediction layer. In the embedding layer, a target user u is first represented by the representations of his/her historical interacted items. Meanwhile, an item's representation consists of its feature embeddings, i.e., each item is represented on feature-level. Next, u's representation and the candidate item v's representation are merged into a big representation matrix $\boldsymbol{X}$, which is the input of the next self-attention layer. In the self-attention layer, an *attention map* (matrix) $\boldsymbol{A}$ is computed, with which $\boldsymbol{X}$ is refined. In addition, we adopt multi-head attention and batch normalization to generate the output of the self-attention layer, which is a vector and denoted as $\boldsymbol{o}$. In the last prediction layer, $\boldsymbol{o}$ is fed into a fully-connected layer to generate the final score $\hat{y}_{uv}$ through a Sigmoid function. In the following introduction, we present the detailed operations in each layer, respectively.

C. Model Details

1) Embedding Layer: The objective of the embedding layer is to generate the representations of the target user and the candidate item, which are the inputs of the next attention layer. In most deep recommendation models including previous self-attention based models [8], [9], [11], [12], a user is represented by the combination of his/her favorite items' representations. Meanwhile, each item is represented by a single embedding (vector) [8], [9] or the union of its feature embeddings [11], [12]. Given the merits of feature-level representations introduced before, an item in SAFrec is represented as the union of its feature embeddings.

Specifically, for each item feature, we first project it into an embedding (vector) $\boldsymbol{e} \in \mathbb{R}^D$ through looking up an embedding matrix. Suppose M features are used to represent items, then the candidate item v is represented by an $M \times D$ matrix as $\boldsymbol{I} = [\boldsymbol{e}_1; \boldsymbol{e}_2; ...; \boldsymbol{e}_M]$.

For the target user u, we consider a fixed number (L) of recently interacted items to represent u. In other words, the L items' representations are united as u's original representation

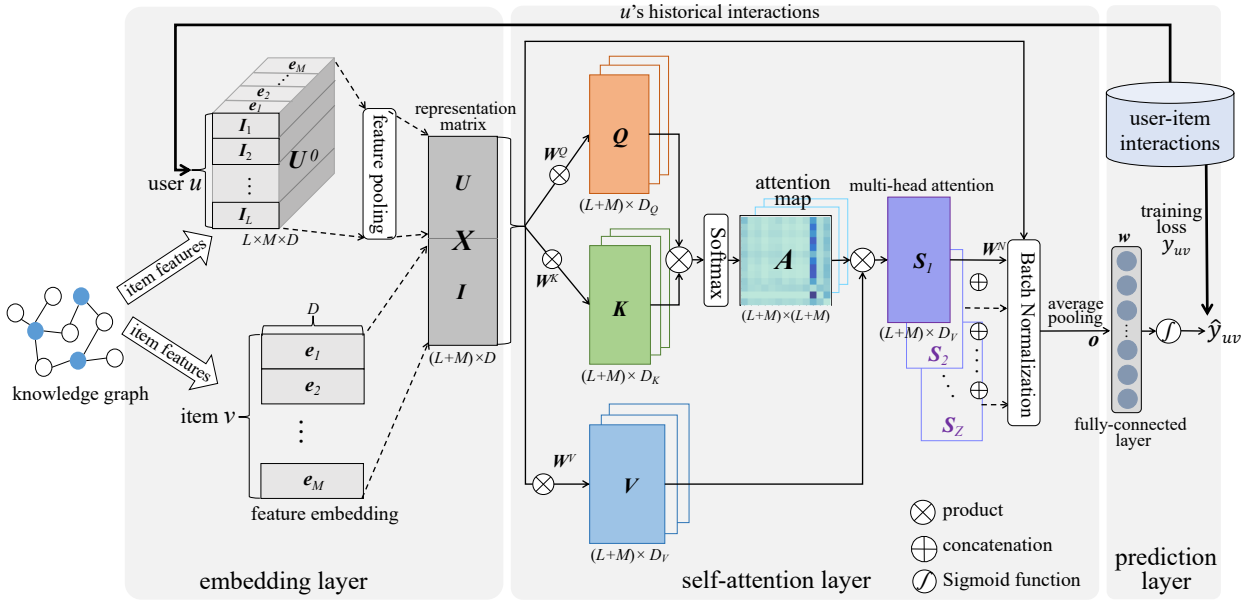


Fig. 1. Framework overview of SAFrec model consisting of three layers.

$\boldsymbol{U}^0$, i.e., $\boldsymbol{U}^0 = [\boldsymbol{I}_1; \boldsymbol{I}_2; ...; \boldsymbol{I}_L]$. Accordingly, $\boldsymbol{U}^0$ is an $L \times M \times D$ cube (tensor) as shown in Fig. 1. For the users with less than L historical interacted items, we use padding to fill the vacant embeddings.

The FM-based recommendation models [1], [2] have proven that capturing feature interactions (correlations) is beneficial to better user modeling, which can be regarded as *feature pooling*. Another advantage of feature pooling is reducing model training cost. In addition, in order to apply self-attention on both user side and item side conveniently, it is necessary to compress $\boldsymbol{U}^0$ into a matrix $\boldsymbol{U}$ through feature pooling. To this end, we use a certain pooling operation to compress each slice (matrix) in $\boldsymbol{U}^0$ into a vector. The pooling operation can be processed in two orthogonal directions. The first one is in feature direction, i.e., pooling all M feature embeddings of each historical item into a single vector. That is,

$$\boldsymbol{U}_{i,:} = Pool_{j=1}^{M}(\boldsymbol{U}_{i,j,:}^0) \tag{1}$$

The second one is in item direction, i.e., pooling the j-th ($1 \leq j \leq M$) feature embeddings of all L historical items into one vector. Hence, we have

$$\boldsymbol{U}_{j,:} = Pool_{i=1}^{L}(\boldsymbol{U}_{i,j,:}^0) \tag{2}$$

In this step, average pooling is a simple and effective choice. For the pooling operation in Eq. 1, we can also adopt the *vanilla attention* as [11], which assigns different weights for different features when merging item feature embeddings. Through our experiments, we have found that the average pooling in feature direction is not only better than the one in item direction, but also better than the vanilla-attention-based operation in feature direction. We will discuss the reasons in the subsequent experiment section.

If Eq. 1 is adopted, $\boldsymbol{U}$ is an $L \times D$ matrix. Then, $\boldsymbol{U}$ and $\boldsymbol{I}$ constitute a big representation matrix $\boldsymbol{X} \in \mathbb{R}^{(L+M)\times D}$, i.e., $\boldsymbol{X}$=$[\boldsymbol{U}; \boldsymbol{I}]$, which contains the information of u and v. For clear presentation, we also name each row in $\boldsymbol{X}$ as a feature embedding in the following texts, where the top L rows are user feature embeddings and the bottom M rows are item feature embeddings.

Note that the inputs of the self-attention in previous sequential models [8], [9], [11] also include position embeddings of each items in the historical sequence, which are used to encode sequential correlations between items. In SAFrec, we neglect position information because we design SAFrec towards top-n recommendation rather than sequential recommendation. It makes SAFrec competent for the recommendation scenario without sequential order or timestamp of historical interacted items.

2) Self-attention Layer: The inputs of generic attention mechanism include *query matrix* $\boldsymbol{Q}$, *key matrix* $\boldsymbol{K}$ and *value matrix* $\boldsymbol{V}$. The standard attention operation is as follows,

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = Softmax(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{D}})\boldsymbol{V} \tag{3}$$

where Softmax function is used to compute the weight between a query i and a value j, corresponding to the interaction (correlation) between query i and the key of value j. $\sqrt{D}$ is a scale factor used to avoid too large values of inner product when the matrix dimension is high.

Self-attention was proposed as a special variant of generic attention, in which $\boldsymbol{Q}$=$\boldsymbol{K}$=$\boldsymbol{V}$ [7]. The first operation in SAFrec's self-attention layer is projecting representation matrix $\boldsymbol{X}$ into three representations of different spaces as below,

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}^Q, \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}^K, \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}^V \tag{4}$$

where $\boldsymbol{W}^Q \in \mathbb{R}^{D\times D_Q}$, $\boldsymbol{W}^K \in \mathbb{R}^{D\times D_K}$ and $\boldsymbol{W}^V \in \mathbb{R}^{D\times D_V}$ are project weight matrices. Accordingly, $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ both have $(L+M)$ rows, and contain the information encoded in the feature-level representations of u and v, i.e., $\boldsymbol{X}$.

The self-attention mechanism in SAFrec is used to capture the correlations (interactions) between different features of the target user and candidate item, which are stored in an *attention map* (matrix) denoted by $\boldsymbol{A} \in \mathbb{R}^{(L+M)\times(L+M)}$. In order to compute the correlations (attentions) in $\boldsymbol{A}$ conveniently, we set $D_Q = D_K$. Formally, we compute $\boldsymbol{A}$ as

$$\boldsymbol{A} = Softmax\big(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{D_K}}\big). \tag{5}$$

Accordingly, $\boldsymbol{A}$ stores the feature correlations of both u and v rather than only u, which is the main difference between SAFrec and previous sequential recommendation models with self-attention [8], [9], [11], [12], and is also one major advantage of SAFrec.

Specifically, if we define an *affinity score* as

$$a_{ij} = \frac{\boldsymbol{Q}_{i,:}\boldsymbol{K}^\top_{j,:}}{\sqrt{D_K}} \tag{6}$$

where $\boldsymbol{Q}_{i,:}$ is the i-th row of $\boldsymbol{Q}$, then the i-th row and j-th column entry of $\boldsymbol{A}$ is computed as

$$\boldsymbol{A}_{i,j} = Softmax(a_{ij}) = \frac{exp(a_{ij})}{\sum_{j'=1}^{L+M} exp(a_{ij'})}. \tag{7}$$

Such Softmax computation indicates that entry $\boldsymbol{A}_{i,j}$ reflects the relatedness of the j-th feature in $\boldsymbol{K}$ to the i-th feature in $\boldsymbol{Q}$.

Next, we utilize $\boldsymbol{A}$ to adjust all embeddings (vectors) in $\boldsymbol{V}$. Specifically, each feature embedding $\boldsymbol{e}_i \in \mathbb{R}^{D_V} (1 \le i \le (L+M))$ in $\boldsymbol{V}$ is adjusted as

$$\boldsymbol{e}_i = \sum_{j=1}^{L+M} \boldsymbol{A}_{i,j}\boldsymbol{e}_j. \tag{8}$$

It indicates that each feature embedding of u is adjusted based on not only all feature embeddings of u but also all feature embeddings of v, and vice versa, since $\boldsymbol{V}$ contains the projected feature embeddings of both u and v. Accordingly, u's representation and v's representations are co-refined.

If the union matrix of adjusted feature embeddings of u and v is denoted by $\boldsymbol{S} \in \mathbb{R}^{(L+M)\times D_V}$, we have

$$\boldsymbol{S} = Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \boldsymbol{A}\boldsymbol{V} = Softmax\big(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{D_K}}\big)\boldsymbol{V}. \tag{9}$$

The computation of $\boldsymbol{S}$ shows that SAFrec applies self-attention mechanism on both user side and item side to co-refine u's representation and v's representation. It also implies that u's representation will be adjusted again when u is confronted with other candidate items, since the values of $\boldsymbol{A}$ and $\boldsymbol{V}$ both vary.

To improve self-attention's performance, we further import *multi-head attention* which allows the model to jointly attend to the information from different representation subspaces at different positions. Specifically, the i-th head is denoted as the matrix $\boldsymbol{S}_i$ computed as

$$\boldsymbol{S}_i = Attention(\boldsymbol{X}\boldsymbol{W}_i^Q, \boldsymbol{X}\boldsymbol{W}_i^K, \boldsymbol{X}\boldsymbol{W}_i^V) \tag{10}$$

where $\boldsymbol{W}_i^{Q/K/V}$ is the weight matrix in the i-th head. The number of heads is denoted as Z.

At last, we add batch normalization [15] to strengthen the performance of SAFrec's self-attention further. Formally, we have

$$\begin{aligned} \boldsymbol{M} &= Concat(\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_Z)\boldsymbol{W}^N \\ \boldsymbol{O} &= BatchNorm(\boldsymbol{M} + \boldsymbol{X}) \end{aligned} \tag{11}$$

where $Concat(\cdot)$ is matrix-wise concatenation and $\boldsymbol{W}^N \in \mathbb{R}^{(D_V\times Z)\times D}$ is a weight matrix. Matrix $\boldsymbol{M}$ and $\boldsymbol{O}$ have the same size as $\boldsymbol{X}$, i.e., $(L+M)\times D$. In order to make the final output adaptive to the next prediction layer, we use average pooling on each column of $\boldsymbol{O}$ and then adopt dropout operation. Therefore, the final output of the self-attention layer is $\boldsymbol{o} = Dropout\big(avg\_pooling(\boldsymbol{O})\big) \in \mathbb{R}^D$.

3) Prediction Layer: As we stated in Subsection II-A, SAFrec accomplishes top-n recommendation based on making an individual prediction whether a candidate item v deserves to be recommended to the target user u or not. It is actually a classic problem of binary classification. Thus we need to build an effective classifier in SAFrec's prediction layer.

With the final output of the self-attention layer, i.e., $\boldsymbol{o}$, we use a fully-connected layer and Sigmoid function to generate the ranking score $\hat{y}_{uv}$ as below, which is the probability that the target user u likes the candidate item v.

$$\hat{y}_{uv} = \sigma(\boldsymbol{o}\boldsymbol{w} + b) = \frac{1}{1+e^{-(\boldsymbol{o}\boldsymbol{w}+b)}} \tag{12}$$

where $\boldsymbol{w} \in \mathbb{R}^D$ is a weight vector and b is an offset.

D. Model Learning

During the process of model training, the value of each feature embedding is tuned according to the training samples. According to the setting of implicit feedback [13], [14], a training sample is formalized as a triplet denoted by $< u, v, y_{uv} >$ where $y_{uv} \in \{0, 1\}$ indicating whether u likes v. SAFrec's training samples are extracted from observed user-item interactions, i.e., historical interacted item sequences of all users. We recognize a positive sample ($y_{uv} = 1$) only if u has reviewed or rated v. For gathering negative samples ($y_{uv} = 0$), we randomly selected some items which have not been reviewed or rated by u as u's disliked items.

We use the following binary cross-entropy as the loss function of our model training, since binary cross-entropy is a classic objective for training neural classifier,

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) = -\sum_{u,v} \big\{ & y_{uv} \log \sigma\big(\phi[\boldsymbol{U};\boldsymbol{I}]\big) \\ & + (1-y_{uv}) \log\big[1-\sigma\big(\phi[\boldsymbol{U};\boldsymbol{I}]\big)\big]\big\} \end{aligned} \tag{13}$$

where ϕ represents the transformation from $[\boldsymbol{U};\boldsymbol{I}]$ (=$\boldsymbol{X}$) to $\boldsymbol{o}\boldsymbol{w} + b$, which is accomplished by the whole three layers of SAFrec.

In our experiments, we use Adam method [16] to optimize Eq. 13, which is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments.

III. EXPERIMENTS

In this section, we try to answer the following research questions through our extensive experiments:

RQ1: Does SAFrec outperform the state-of-the-art recommendation models?

RQ2: Is it worthwhile to feed SAFrec with feature-level representations?

RQ3: Is employing self-attention mechanism to co-refine user representations and item representations more useful than employing self-attention to only refine user representations?

RQ4: Does SAFrec sufficiently capture the correlations between a user's historical interacted items and his/her next interacted items on feature level?

A. Experiment Setup

1) Experiment Datasets: We conducted our experiments over two datasets, i.e., NetEase music[1] and Book-Crossing [2]. In the two datasets, user reviews on songs and user ratings on books were both regarded as positive feedbacks (interactions). To better compare all models' performance, we filtered out the users only having few interacted items. The related statistics of the datasets are listed in Table II. In addition, we selected singer, composer and album as song features, and selected author, publisher and publication year as book features, thus M=3. Our used datasets and SAFrec's codes have been published on *https://github.com/DeqingYang/SAFrec*.

TABLE II
STATISTICS OF THE TWO EXPERIMENT DATASETS.

domain	# user	# item	# interaction
NetEase Music	3,065	33,138	110,077
Book-Crossing	6,781	50,000	201,941

2) Sample Collection: For each user in the datasets, we randomly selected 5 items he/she has interacted (reviewed or rated) and 50 negative items (not interacted items) to constitute the test samples, since a user's uninterested items are much more than his/her favorite items in real life. The rest interacted items of the user were used as his/her historical interacted items, i.e., the positive samples in the training set. The positive samples in the test set were regarded as the next interacted items of users since our datasets have no timestamps. As other recommendation models' popular setting [4], [13], the ratio of positive items to negative items of a user in the training set is 1:4.

3) Baselines: We further introduce some state-of-the-art deep recommendation models including previous self-attention based models, to be compared with SAFrec.

AFM [1] and **NFM** [2]: These two models are the neural versions of factorization machines (FM) [17] which capture the second-order and high-order (item) feature interactions respectively, to improve recommendation performance.

KE [18]: It is built with the same feature-level embedding layer as SAFrec, but without self-attention module. It uses a multi-layer perceptron (MLP) fed with user representation and item representation to output the final score.

RippleNet [19]: This is a state-of-the-art KG-based model in which the utilized knowledge is just the item features.

SA [8]: In this self-attention based model, each item is first represented by a single embedding (item-level) initialized in random rather than item feature embeddings. Furthermore, its self-attention module is only applied on user side to refine user representations.

FDSA [11]: It also incorporates feature-level representations but only employs self-attention on user side. Note that FDSA and SA were not fed with position embeddings in our experiments since our datasets have no timestamps of user-item interactions.

Furthermore, we use **SAFrec** and **SAFrec**$^{\perp}$ to denote our model with the average pooling in feature direction ($\boldsymbol{U}$ has L rows) and in item direction ($\boldsymbol{U}$ has M rows), respectively. And **SAFrec**^{va} denotes the variant of our model with the vanilla-attention-based merging in feature direction.

4) Performance Metrics: We select some popular metrics of top-n recommendation performance, i.e., MRR (Mean Reciprocal Rank), HR (Hit Ratio), MAP (Mean Average Precision), nDCG (Normalized Discounted Cumulative Gain) [20]. For each model running, we recorded performance scores of top-n recommendations averaged on all test users. To avoid statistics bias, the results of each model reported in the following tables are the average scores of 5 runnings.

B. Experiment Results

1) Hyper-parameter Tuning: We first studied the performance influence of our model's hyper-parameters. Due to space limitation, we only display the results of tuning three important hyper-parameters, i.e., embedding dimension D, the number of considered historical interacted items L, and the number of attention heads Z. Note that we set the rest hyper-parameters to their optimal values when we tuned one hyper-parameter.

TABLE III
TOP-5 NETEASE MUSIC RECOMMENDATION PERFORMANCE OF SAFREC WITH DIFFERENT EMBEDDING DIMENSION D.

D	MRR	HR@5	MAP@5	nDCG@5
50	0.4478	0.2683	0.1945	0.2801
100	0.4476	0.2799	0.2059	0.2900
150	0.4704	0.2887	0.2156	0.3020
200	0.5004	0.3096	0.2333	0.3242
300	**0.5223**	**0.3219**	**0.2517**	**0.3405**
400	0.5127	0.3218	0.2276	0.3383

Table III lists SAFrec's top-5 NetEase music recommendation performance as the function of embedding dimension D. The results show that D=300 is the best setting. Thus, we set D=300 when we ran SAFrec in the following comparison experiments. Furthermore, we set D_K=D_V=120 which have also been proven to be optimal through our empirical studies.

Table IV lists the results of considering different number of historical interacted items (L) when representing a user. The results show that small L can not represent a user abundantly resulting in inferior performance. Howbeit, bigger

[1]https://music.163.com

[2]http://www2.informatik.uni-freiburg.de/∼cziegler/BX/

TABLE IV
TOP-5 NETEASE MUSIC RECOMMENDATION PERFORMANCE OF SAFREC WITH DIFFERENT NUMBER OF CONSIDERED HISTORICAL INTERACTED ITEMS.

L	MRR	HR@5	MAP@5	nDCG@5
3	0.4310	0.2868	0.2068	0.2968
5	0.4893	0.3034	0.2252	0.3153
10	**0.5223**	**0.3219**	**0.2517**	**0.3405**
15	0.4867	0.2938	0.2151	0.3059
20	0.4276	0.2746	0.1973	0.2844

TABLE V
TOP-5 NETEASE MUSIC RECOMMENDATION PERFORMANCE OF SAFREC WITH DIFFERENT NUMBER OF ATTENTION HEADS, I.E., Z.

Z	MRR	HR@5	MAP@5	nDCG@5
1	0.4298	0.2703	0.1942	0.2766
3	0.4899	0.3072	0.233	0.3208
5	**0.5223**	**0.3219**	**0.2517**	**0.3405**
8	0.5217	0.3161	0.2433	0.3349

L does not lead better performance because many users are not found having enough historical interacted items in our datasets. In this scenario, more paddings are used to fill user representations, incurring more noises. According to the tuning results, we set L=10 in NetEase music recommendation.

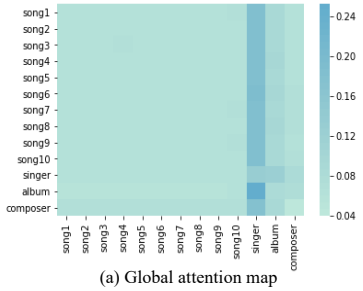
Since we import multi-head attention into SAFrec to further enhance the model's capability, an interesting question is how many heads are optimal? To answer this question, we evaluated SAFrec's performance under different Zs. Table V lists SAFrec's top-5 NetEase music recommendation performance as the function of Z. The results show that using 5 attention heads is the best setting for SAFrec's performance.

2) Effectiveness on Top-N Recommendation: To answer RQ1, we compared all models' performance on music recommendation and book recommendation. Table VI lists all models' top-3/5 recommendation performance where we set L=3 in book recommendation since user-item interactions in Book-Crossing is more sparer. The results shows that SAFrec outperforms its competitors remarkably.

SAFrec's superiority over SA gives yes answer to both RQ2 and RQ3. The reason of SA's inferior performance is two-fold. One reason is SA only uses item-level representations. The other reason is SA only applies self-attention mechanism to refine user representations. Although FDSA also incorporates feature-level representation as SAFrec, it is defeated by SAFrec implying that employing self-attention on both user side and item side is more effective than just on user side.

SAFrec's advantage over SAFrec$^{\perp}$ and SAFrec^{va} justifies the average pooling in feature directions is a better strategy to compress $\boldsymbol{U}^0$ into $\boldsymbol{U}$. One possible reason of SAFrec's superiority over SAFrec$^{\perp}$ is that, a user in SAFrec is represented by more embeddings (in $\boldsymbol{U}$) since L is bigger than M in general. Although vanilla-attention operation is more complicated than average pooling and considers different weights for different features to represent an item, SAFrec also outperforms SAFrec^{va}. It is possibly because that, the self-attention mechanism in SAFrec can capture the correlations between the candidate item's features and a user's historical items, which are used to assign different weights for different features to refine user/item representations. As a result, the vanilla-attention operation in generating $\boldsymbol{U}$ may disturb the self-attention's effectiveness.

Although KE and RippleNet both incorporate knowledge, i.e., item features, they are inferior than SAFrec. It justifies that employing self-attention to generate adaptive user/item representations can improve recommendation performance further. SAFrec's superiority over AFM and NFM shows that the feature correlations (interactions) captured by SAFrec's self-attention are more useful than those captured by AFM and NFM in terms of recommendation performance.



(a) Global attention map

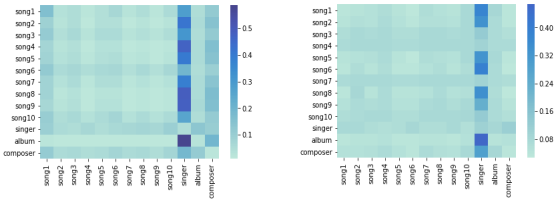

(b) User a's attention map (c) User b's attention map

Fig. 2. The self-attention map (matrix) of all NetEase users and two special users.

3) Insights into Self-Attention Map: To answer RQ4, We exhibit the correlations between a user's next interacted item and his/her historical interacted items which are captured by SAFrec's self-attention. To this end, we visualize the heat map of attention matrix $\boldsymbol{A}$ computed by SAFrec in macro-level and micro-level, respectively. Due to space limitation, we only show the investigation results of NetEase music recommendation.

a) Macro-level Analysis: In order to exhibit the correlations between a user's next reviewed songs and his/her historical favorite songs, we first filtered out 712 users who have more than 10 positive songs in the training set since we set $L = 10$. The reason of such filtering is to avoid the influence of paddings in user representations. We took each test user's positive songs as the next reviewed songs, except for the 10 songs regarded as the user's historical favorite songs. After model training, we fed each user u along with each of u's next reviewed songs into SAFrec, to compute $\boldsymbol{A}$ where multi-head attention is not used. Then, u's attention map is the average of all $\boldsymbol{A}$s computed w.r.t. each of u's next reviewed songs. Fig. 2(a) is the average heat map of all users' attention maps. According to Eq. 7, $\boldsymbol{A}$ is not a symmetrical matrix, and the i-th column in $\boldsymbol{A}$ reflects the importance of the i-th feature to other features including 10 historical favorite songs. Accordingly, the column of singer has the highest attention values, showing that the singers of a user's next reviewed songs are most correlated with his/her historical reviewed

TABLE VI
PERFORMANCE COMPARISONS OF TOP3/5 RECOMMENDATION ON MUSIC RECOMMENDATION AND BOOK RECOMMENDATION.

domain	Model	MRR	HR@3	HR@5	MAP@3	MAP@5	nDCG@3	nDCG@5
NetEase Music	AFM	0.4655	0.2904	0.2564	0.2322	0.1878	0.2908	0.2744
	NFM	0.4726	0.2657	0.2390	0.2182	0.1745	0.2836	0.2613
	KE	0.4841	0.3351	0.3131	0.2764	0.2333	0.3383	0.3222
	RippleNet	0.4940	0.3258	0.2994	0.3012	0.2516	0.3186	0.3001
	SA	0.4870	0.3353	0.2588	**0.3114**	0.2512	0.3373	0.3165
	FDSA	0.4924	0.3325	0.3043	0.2701	0.2217	0.3370	0.3165
	SAFrec$^{\perp}$	0.4641	0.2986	0.2639	0.2432	0.1922	0.3059	0.2801
	SAFrec^{va}	0.4669	0.3138	0.2862	0.2514	0.2074	0.3169	0.2971
	SAFrec	**0.5223**	**0.3624**	**0.3219**	0.3083	**0.2517**	**0.3706**	**0.3405**
Book-Crossing	AFM	0.7244	0.5028	0.4284	0.4436	0.3459	0.5253	0.4679
	NFM	0.5258	0.3838	0.3502	0.3298	0.2109	0.4212	0.4012
	KE	0.7102	0.5007	0.4208	0.4401	0.3444	0.5224	0.4664
	RippleNet	0.7383	0.5289	0.4387	0.4423	0.3512	0.5337	0.4767
	SA	0.5974	0.3542	0.3002	0.3251	0.2629	0.3737	0.3311
	FDSA	0.6285	0.4227	0.3751	0.3442	0.2732	0.4330	0.3975
	SAFrec$^{\perp}$	0.7178	0.5051	0.4345	0.4428	0.3479	0.5268	0.4722
	SAFrec^{va}	0.7204	0.5016	0.4278	0.4413	0.3437	0.5251	0.4678
	SAFrec	**0.7459**	**0.5558**	**0.4744**	**0.4914**	**0.3874**	**0.5770**	**0.5152**

songs. It implies that singer is the most significant feature to generate precise music recommendation results. In addition, $\boldsymbol{A}_{12,11}$'s big value indicates singer's significant correlation to album. It is unsurprising since an album only has fixed singer(s). Comparatively, $\boldsymbol{A}_{11,12}$ is not so big because a singer generally owns several albums.

b) Micro-level Analysis: Furthermore, we also investigated two special users (a and b) and depicted their attention maps in Fig. 2(b) and 2(c), respectively. These two maps also indicate that singer is still the most significant feature for them. However, besides singer, user a pay more attention to composer while user b focus more on album. Such results validate that many users select their favorite in terms of different features, which poses a great challenge to the design of personalized recommendation models.

IV. RELATED WORK

a) Attention-based Models: The attention mechanism in DNNs is inspired by the intuition of the visual attention found in humans. It learns to pay attention only to the most important parts of the target. Attention mechanism has been widely employed in various tasks such as image captioning [21] and machine translation [22]. The idea behind attention is that each output depends on relevant parts of inputs (with different weights) that the model should focus on successively. In recent years, attention mechanism has been imported into recommender systems successfully [1], [5], [23]. For example, AFM [1] learns the weight of each feature interaction for content-aware recommendation. The authors in [4], [5] used attention network to generate adaptive user representations based on the similarity/correlation between the candidate item and historical items. More recently, Google researchers proposed a purely attention-based sequence-to-sequence model Transformer [7], which has been proven to exhibit better performance and efficiency than RNN/CNN-based models on machine translation. They used such self-attention mechanism to capture complex structures in sentences, and thus retrieve relevant words in input sequence for generating the next word in output sequence. Inspired by this work, the authors in [8], [9], [11], [12] also employed self-attention module into a sequential recommendation model. Specifically, [8], [9] only adopt item-level representations which restrict recommendation performance. Although [11], [12] import feature-level representations as our SAFrec, they and the former two self-attention-based models use self-attention to capture the correlations between different items (or behaviors) in a user's historical interaction sequence, and then to refine user representations rather than the candidate items' representations. Therefore, there is still space for improving these models.

b) Deep Recommendation Models: In recent years, many researchers have employed various DNNs into recommender systems for broad recommendation tasks. Traditional recommendation algorithms such as CF and matrix factorization, are proven to be improved by DNN-based models. For example, [24] proposed a novel AutoEncoder (AE) framework for CF. [25] proposed deep matrix factorization, and NFM [2] is a neural version of FM [17]. NCF model [13] integrates generalized matrix factorization model and MLP to predict CF-based implicit feedback. Inspired by CNN's power in computer vision processing, the authors in [26] strived to bridge the semantic gap in music by training CNNs to predict latent factors from audio. In addition, various DNN-based encoders and decoders are also imported, including Bayesian aSDAE [27] which is utilized to enhance CF-based recommendation [28]. For sequential recommendation, GRU4Rec [29] employs GRU [30] and KSR [31] utilizes a key-value memory network. DIN [32] is also a deep recommendation model with feature-level representations, but it does not employ self-attention mechanism.

c) KG-based Recommendation: Existing KG-based recommender systems can be categorized into three classes: embedding-based method, path-based method and unified method [33]. In embedding-based methods, the KG information about items and users is first encoded into low-rank

embeddings by a KG embedding model, and then a user's preference on an item is computed by a function mapping the user embedding and the item embedding, which can be inner product or DNN. This class includes DKN [5], CKE [14], KSR [31] and KTGAN [34]. KE [18] and our SAFrec also belong to this class. Path-based methods build a user-item graph and leverage the connectivity patterns among the entities in the graph for recommendation. The graph is often recognized as the heterogeneous information network since it contains multiple types of nodes and edges. This class includes the models and algorithms in [35]–[37]. To fully exploit KGs for better recommendations, unified methods have been proposed which is based on the idea of embedding propagation. These methods refine the entity embeddings with the guidance of the structure information in KGs, integrating the merits of the aforementioned two methods. The representative models of this class include KGAT [38] and RippleNet [19] etc.

V. Conclusion

In this paper, we propose a deep recommendation model with feature-level self-attention, namely SAFrec. In order to learn more flexible and adaptive representations of users and items for enhanced recommendation, SAFrec employs a self-attention mechanism of feature-level on user side and item side simultaneously. Compared with the self-attention built in previous sequential recommendation models, SAFrec's self-attention captures the correlations between the candidate item and the target user on feature-level, achieving recommendation performance gains. The extensive evaluations over two real recommendation datasets justify SAFrec's superiority over the state-of-the-art deep recommendation models including the recommendation models with self-attention and KG-based recommendation models.

Acknowledgements

This paper was supported by National NSF of China No. U1636207, Shanghai Science and Technology Innovation Action Plan No. 19511120400.

References

[1] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T. S. Chua, "Attentional factorization machines: Learning the weight of feature interactions via attention networks," in *Proc. of IJCAI*, 2017.

[2] X. He and T. S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proc. of SIGIR*, 2017.

[3] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "A dynamic recurrent model for next basket recommendation," in *Proc. of SIGIR*, 2016.

[4] X. He, Z. He, J. Song, Z. Liu, Y. G. Jiang, and T. S. Chua, "Nais: Neural attentive item similarity model for recommendation," *IEEE TKDE*, pp. 1–1, 2018.

[5] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," in *Proc. of WWW*, 2018.

[6] C. Shen, D. Yang, and Y. Xiao, "A deep recommendation model incorporating adaptive knowledge-based representations," in *Proc. of DASFAA*, 2019.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. of NIPS*, 2017.

[8] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," 2018.

[9] S. Zhang, Y. Tay, L. Yao, and A. Sun, "Next item recommendation with self-attention," 2018.

[10] X. Huang, S. Qian, Q. Fang, J. Sang, and C. Xu, "Csan: Contextual self-attention network for user sequential recommendation," in *Proceedings of ACM International Conference on Multimedia*, 2018.

[11] T. Zhang, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, D. Wang, G. Liu, and X. Zhou, "Feature-level deeper self-attention network for sequential recommendation," in *Proc. of IJCAI*, 2019.

[12] Z. C., B. J., and S. J. et al., "Atrank: An attention-based user behavior modeling framework for recommendation," in *Proc. of AAAI*, 2018.

[13] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. S. Chua, "Neural collaborative filtering," in *Proc. of WWW*, 2017.

[14] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. of KDD*, 2016.

[15] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network," in *Proc. of ICML*, 2015.

[16] J. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. of ICLR*, 2015.

[17] S. Rendle, "Factorization machines," in *Proc. of ICDM*, 2010.

[18] D. Yang and Z. W. et al., "Knowledge embedding towards the recommendation with sparse user-item interactions," in *Proc. of ASONAM*, 2019.

[19] W. Hongwei, Z. Fuzheng, W. Jialin, M. Zhao, L. Wenjie, X. Xie, and G. Minyi, "Ripple network: Propagating user preferences on the knowledge graph for recommender systems," in *Proc. of CIKM*, 2018.

[20] K. Jarvelin and J. Kekalainen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems*, vol. 20, pp. 422-446, 2002.

[21] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proc. of CVPR*, 2015.

[22] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2015.

[23] J. Chen, H. Zhang, X. He, W. Liu, W. Liu, and T. S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention," in *Proc. of SIGIR*, 2017.

[24] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *Proc. of WWW*, 2015.

[25] H. J. Xue, X. Y. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. of IJCAI*, 2017.

[26] A. V. den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Proc. of NIPS*, 2013.

[27] P. Vincent, H. Larochelle, Y. B. I. Lajoie, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *JMLR*, vol. 11, pp. 3371 - 3408, 2010.

[28] X. Dong, L. Yu, ZhonghuoWu, Y. Sun, L. Yuan, and F. Zhang, "A hybrid collaborative filtering model with deep structure for recommender systems," in *Proc. of AAAI*, 2017.

[29] B. Hidasi and A. K. et al., "Session-based recommendations with recurrent neural networks," in *Proc. of ICLR*, 2016.

[30] C. Kyunghyun, V. M. Bart, B. Dzmitry, and B. Yoshua, "On the properties of neural machine translation: Encoder-decoder approaches," *Computer Science*, 2014.

[31] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *Proc. of SIGIR*, 2018.

[32] G. Zhou, C. Song, X. Zhu, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proc. of KDD*, 2018.

[33] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He, "A survey on knowledge graph-based recommender systems," *CoRR*, vol. abs/2003.00911, 2020.

[34] D. Yang, Z. Guo, Z. Wang, J. Jiang, Y. Xiao, and W. Wang, "A knowledge-enhanced deep recommendation framework incorporating gan-based models," *ICDM*, pp. 1368–1373, 2018.

[35] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: a heterogeneous information network approach," in *WSDM*, 2014, pp. 283–292.

[36] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Porc. of KDD*, 2017, pp. 635–644.

[37] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model," in *Porc. of KDD*, 2018, pp. 1531–1540.

[38] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," *Porc. of KDD*, 2019.