Knowledge-Based Systems 187 (2020) 104839

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Gated and attentive neural collaborative filtering for user generated list recommendation☆

Chao Yang [a,*], Lianhai Miao [a], Bin Jiang [a], Dongsheng Li [b], Da Cao [a]

[a] College of Computer Science and Electronic Engineering, Hunan University, Lushan Road(s), Yuelu District, 410082, Changsha, Hunan, China
[b] IBM Research, NO. 399 Keyuan Road, Shanghai, 201203, China

ARTICLE INFO

Article history:
Received 11 December 2018
Received in revised form 7 June 2019
Accepted 9 July 2019
Available online 12 July 2019

Keywords:
Recommender systems
Collaborative filtering
Neural network
List recommendation

ABSTRACT

Recommending user generated lists (e.g., playlists) has become an emerging task in many online systems. Many existing list recommendation methods predict user preferences on lists by aggregating their preferences on individual items, which neglects the list-level information (e.g., list attributes) and thus results in suboptimal performance. This paper proposes a neural network-based solution for user generated list recommendation, which can leverage both item-level information and list-level information to improve performance. Firstly, a representation learning network with attention and gate mechanism is proposed to learn the user embeddings, item embeddings and list embeddings simultaneously. Then, an interaction network is proposed to learn user–item interactions and user–list interactions, in which the two kinds of interactions can share the convolution layers to further improve performance. Experimental studies on two real-world datasets demonstrate that (1) the proposed representation learning network can learn more representative user/item/list embedding than existing methods and (2) the proposed solution can outperform state-of-the-art methods in both item recommendation and list recommendation in terms of accuracy.

1. Introduction

In the era of information exploration, users are heavily overwhelmed by the huge volumes of information and products. Fortunately, Recommender systems, which were proposed to address the information overload issue, have been widely applied in various online service providers, e.g., e-commerce platform [1], social media website [2], online news portal [3], etc. An effective recommender system can not only help the service provider to increase the traffic and profit, but can also help customers to find interesting items easily. The interesting items can be recommended to users in different levels, e.g., traditional individual item recommendations [1] and the newly emerging item list recommendations [4,5].

Moving beyond the traditional task of recommending individual items to users, in this work, we focus on a more complicated scenario — recommending user generated lists. During the recent few years, user online behaviors have gradually extended from traditional individual items (e.g., a song or a book) to aggregated items (e.g., a music playlist or a book reading list). Many online service providers, such as Spotify,[1] GoodReads,[2] Douban,[3] etc., provide user generated lists, e.g., music playlists, reading lists, movie watching lists, etc., for users to share among their social friends. As a representative application of aggregated items, user generated lists are usually manually organized by the creators according to specific intents and are exposed to the public by default. Therefore, it will be interesting and useful to recommend these user generated lists to other users, which can not only help other users find their desired item lists but can also facilitate the online service providers to reinforce user engagements [5].

Despite of its value and importance, the problem of user generated list recommendation remains in its infancy and many challenges cannot be properly addressed using existing techniques. (1) The items within a list are not equally important, and the importance of items should vary among different users. Therefore, it is desirable to dynamically adjust the weights of items within a list to capture the complicated meanings of lists. (2) The items within a list are usually gathered according to a specific theme, which may contain some latent attributes. It will

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.07.010.

* Corresponding author.
E-mail addresses: yangchaoedu@hnu.edu.cn (C. Yang), lianhaimiao@gmail.com (L. Miao), jiangbin@hnu.edu.cn (B. Jiang), ldsli@cn.ibm.com (D. Li), caoda0721@gmail.com (D. Cao).

1 https://www.spotify.com
2 https://www.goodreads.com
3 https://www.douban.com

https://doi.org/10.1016/j.knosys.2019.07.010
0950-7051/

be useful to leverage user preferences over the latent attributes to infer their preferences over the lists. (3) Different levels of information, e.g., item-level user preference, list-level user preference and list latent attributes, should be all considered when learning the representations of the user generated lists. (4) A user's preference over a list should manifest his/her preference over the items within the list, and vice versa, so that it will be useful to design user–list recommendation solutions that can benefit from user–item recommendations, and vice versa.

To this end, we propose the GANCF method (short for "**G**ated and **A**ttentive **N**eural **C**ollaborative **F**iltering") to tackle the user generated list recommendation problem. Firstly, we employ the recent advance in neural network modeling, i.e., the attention mechanism, to capture user preferences over individual items and lists. More specifically, three attention networks are proposed to capture different levels of user preferences: (1) an item-level attention network, which can capture user preferences over individual items within a list; (2) a self-attention network, which can learn the latent attributes of each list; and (3) an attribute-level attention network, which can capture the user preferences over individual attributes within each list. Then, the item-level attention network, the attribute-level attention network, and the list embedding are connected via a gated feature fusion network, which can further reinforce the representation learning of the user generated lists. Finally, both user–item and user–list interactions are modeled via a shared interaction learning network, so that the recommendation performances of user–item and user–list can be mutually enhanced. Experiments on two real-world datasets demonstrate that the proposed representation learning network can learn more representative user/item/list embedding compared with existing methods and the shared interaction learning network can achieve higher recommendation accuracy compared with state-of-the-art methods. Our key contributions are summarized as follows:

- We have proposed a neural network-based method to address the representation learning problem of user generated lists, which employs attention mechanism and gated fusion mechanism to facilitate the representation learning. To the best of our knowledge, this is the first work that addresses list recommendation problem under the neural network framework.
- We have proposed a shared interaction learning network to integrate user–item and user–list interactions simultaneously, which can reinforce the performances of both item recommendation and list recommendation.
- Extensive experiments on two real-world datasets have demonstrated the effectiveness of the proposed methods. Meanwhile, we have released the datasets and the implementations, which can facilitate the research community to further explore the important user generated list recommendation problem.[4]

The rest of the paper is organized as follows. After introducing related works in Section 2, we elaborate our proposed methods in Section 3. We then perform experimental evaluation in Section 4. Finally, we conclude the whole paper and give an outlook of future work in Section 5.

2. Related work

We focus on describing the related work on item recommendation, list recommendation and deep learning techniques for recommendation.

2.1. Item recommendation

There are two kind of tasks in item recommendation, The first one with explicit feedback (e.g., user ratings), which directly reflects the preference of users on items, is usually formulated as a rating prediction problem [6–10], the target is to minimize the overall errors between the known ratings and the corresponding prediction scores. Among various item recommendation approaches, matrix factorization has been the frequently praised model due to its simplicity and effectiveness. Biased MF is proposed to further enhance the performance of traditional MF in the problem of rating prediction. Researchers in [2,11–14] introduced additional information like review texts and social relations into MF so as to address the rating sparsity issue. Among numerous MF-based approaches, SVD++ has been proved to be the best single model in terms of fitting user ratings. The second one, which based on implicit feedback (e.g.,view, click) rather explicit ratings, is usually treated as top-N recommendation task [15]. Many approaches have been proposed on the basis of implicit feedback [16–21] Technically, the main difference between the tasks of rating prediction and top-N recommendation lies in the way of model optimization. In particular, the former usually constructs a regression loss function only on the known ratings to optimize, yet the latter needs to take the remaining data (a mixture of real negative feedback and missing data) into consideration which are always ignored by the models for explicit feedback.

2.2. List recommendation

Unlike traditional recommendation tasks that recommend isolated items to users, the list recommendation task focuses on recommending a list of items. Zhu et al. [4] defined the concept of bundle recommendation, in which a bundle refers to a set of items that users consumed together under a specific circumstance (e.g., limit total price, contextual influence, and product compatibility). Furthermore, Liu, Xie and Lakshmanan [22] proposed the LIRE method that aims to recommend items and lists to users in which users' previous interactions with both lists and individual items are considered simultaneously. The relationship between the list and its associated items is modeled via a linear model. After that Sar et al. [23] proposed a novel two-layered framework which can account for linear inter-item interactions as well as additional information. To capture more complex relationships among items within a list, Cao et al. [5] proposed the EMF-joint model, which treats a list as a sentence and items within the list as words, then resort to embedding factorization models to explore the list recommendation problem. Although these works have shown good performances, previous efforts on list recommendation mainly focus on matrix factorization based methods, and fail to explore the powerful neural network based methods, which is the research gap that we aim to bridge in this work.

2.3. Deep learning for recommendation

The recent emerging deep learning research have inspired many interesting works on neural network-based recommendation algorithms [9,14,24–28]. most of these works that integrates recommender systems with deep learning methods utilized deep neural networks for modeling auxiliary information, such as image features [29], makeup information [30] and costume aesthetics [31]. The learned features can then be incorporated into collaborative filtering algorithms to improve performance. These works mainly falls into a two-stage mode, in which recommender systems and deep learning methods are implemented separately. Different from previous work, He et al. [18]

4 https://listrecs.wixsite.com/gancf

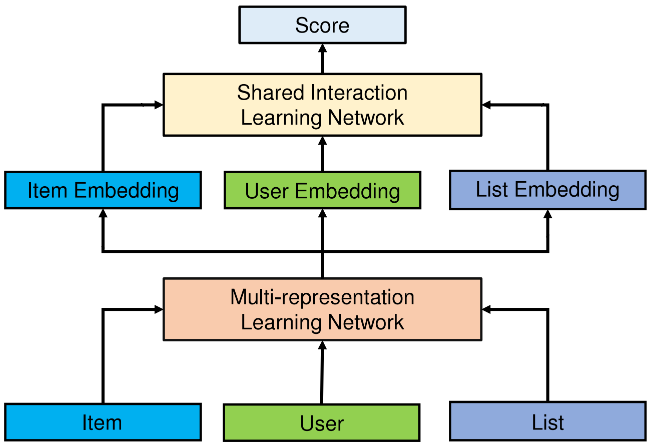


Fig. 1. A high-level illustration of the proposed GANCF method.

cast the well-established matrix factorization algorithm into an overall neural network framework, which combines the traditional inner-product based learner with a series of stacked non-linear transformations. Notably, the Neural Collaborative Filtering (NCF) framework [18] can outperform many traditional approaches. In order to model the correlations between different embedding dimensions, ONCF [32] was proposed to exploit outer productions and 2D convolution layers for learning joint representations of user–item pairs.

Compared with the above existing works, our model is different from them in two aspects. The first aspect is that existing list recommendation models mainly focus on matrix factorization-based methods which are hard to model the non-linear and complex relationship among items, while we employ the self-attention layer to address this problem. The second aspect is that we utilize the shared CNN layers to learn the interaction function between user–item and user–list, and reinforce the performance of both recommendation tasks simultaneously.

3. The proposed methods

In this paper, we propose to address the user generated list recommendation problem under the representation learning (RL) framework [33]. Under the RL paradigm, each entity is represented as an embedding vector, which encodes the inherent properties of the entity (e.g., interests of a user, attributes of a list, etc.) by learning from the data.

As illustrated in Fig. 1, the proposed GANCF method consists of two main components: (1) a multi-representation learning network, which learns the user embeddings, item embeddings and list embeddings simultaneously; and (2) a shared interaction learning network, which performs user–item recommendation and user–list recommendation simultaneously. The details of each component in GANCF will be presented in the following sections.

3.1. Notations and problem formulation

We use bold capital letters (e.g., $\mathbf{X}$) and bold lowercase letters (e.g., $\mathbf{x}$) to represent matrices and vectors, respectively. Non-bold letters (e.g., x) denote scalars and squiggle letters (e.g., $\mathcal{X}$) denote sets. All vectors are in column forms unless otherwise specified.

In the targeted problem, we assume that there are n users $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$, m items $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$ and s lists $\mathcal{L} = \{l_1, l_2, \ldots, l_s\}$. The tth list $l_t \in \mathcal{L}$ consists of a set of items with indexes $\mathcal{C}_t = \{c_{t,1}, c_{t,2}, \ldots, c_{t,|l_t|}\}$, where $v_{c_{t,*}} \in \mathcal{V}$, and $|l_t|$ is the size of the tth list. There are two kinds of observed interaction data among $\mathcal{U}$, $\mathcal{V}$, and $\mathcal{L}$, namely, user–item interactions and user–list interactions. We use $\mathbf{Y} = [y_{ij}]_{n\times m}$ to denote user–item interactions and $\mathbf{R} = [r_{it}]_{n\times s}$ to denote user–list interactions. Then, given a targeted user u_i, our task is to recommend a set of items and lists that u_i will be interested in.

3.2. Multi-representation learning network

Fig. 2 shows the architecture of the proposed multi-representation learning network, which aims to learn user embeddings, item embeddings and list embeddings simultaneously.

3.2.1. Item attention layer

Let $\mathbf{u}_i \in \mathbb{R}^k$ be the embedding vector for user u_i and $\mathbf{l}_t \in \mathbb{R}^k$ be the origin embedding vector for list l_t, where k is the embedding size. The embedding vector of items in list l_t can be written as $\mathbf{v}_{c_{t,1}}, \mathbf{v}_{c_{t,2}}, \ldots, \mathbf{v}_{c_{t,|l_t|}}$, where $\mathbf{v}_{c_{t,*}} \in \mathbb{R}^k$. Here, our goal is to obtain an embedding vector representing a user's preference over individual items. Since different items are of different interests to each user, we use an attention network [34] to model the different levels of the user's preference over individual items. Specifically, for user $\mathbf{u}_i$ and a list of items $\{\mathbf{v}_j\}_{j\in\mathcal{C}_t}$, we first concatenate each item embedding with the same user embedding, then apply multi-layer perceptron (MLP) as the attention network and the softmax function is utilized to normalize the impact weights:

$$\begin{cases} s'(i,j) = \mathbf{h}_t^T \mathrm{ReLU}(\mathbf{W}_t([\mathbf{u}_i, \mathbf{v}_j]) + \mathbf{b}_t) \\ s(i,j) = \dfrac{\exp(s'(i,j))}{\sum_{j\in\mathcal{C}_t}\exp(s'(i,j))} \end{cases}, \tag{1}$$

where $\mathbf{W}_t \in \mathbb{R}^{a\times 2k}$ and $\mathbf{b}_t \in \mathbb{R}^a$ denote the weight matrix and bias vector of the hidden layer, respectively, and a controls the size of the hidden layer. We use ReLU as the activation function and vector $\mathbf{h}_t \in \mathbb{R}^a$ projects the hidden layer into the attentive weight for output. Hence, a user's preference over a list can be calculated as the weighted sum of the items embeddings within the list as follows:

$$\mathbf{e}_{i,t} = \sum_{j\in\mathcal{C}_t} s(i,j)\mathbf{v}_j. \tag{2}$$

3.2.2. Self-attention layer

Items within a list are usually gathered together based on some common themes, which can provide some correlation of the items. However, how to model and capture the relationship among items is non-trivial. Inspired by the recent success that uses self-attention mechanism [35] to learn the relationship among words, we use a self-attention network to learn the relationship among the items within each list.

Self-attention, also called intra-attention, is an attention mechanism which can automatically capture the internal relationship of a single sequence and compute its representation without any additional information. Self-attention has been successfully applied to various tasks, such as reading comprehension [36], language understanding [37], semantic role labeling [38], etc. We adopt the multi-head attention mechanism proposed by Vaswani et al. [35]. Fig. 3 depicts the architecture of the multi-head attention mechanism. The center of the graph is the scaled dot-product attention, which is a variant of dot-product (multiplicative) attention [39,40]. Compared with the standard additive attention mechanism [41,42] which is implemented by using a one layer feed-forward neural network, the dot-product attention utilizes matrix production which allows faster computation. Given a matrix of n query vectors $\mathbf{Q} \in \mathbb{R}^{n\times k}$, keys $\mathbf{K} \in \mathbb{R}^{n\times k}$ and values $\mathbf{V} \in$

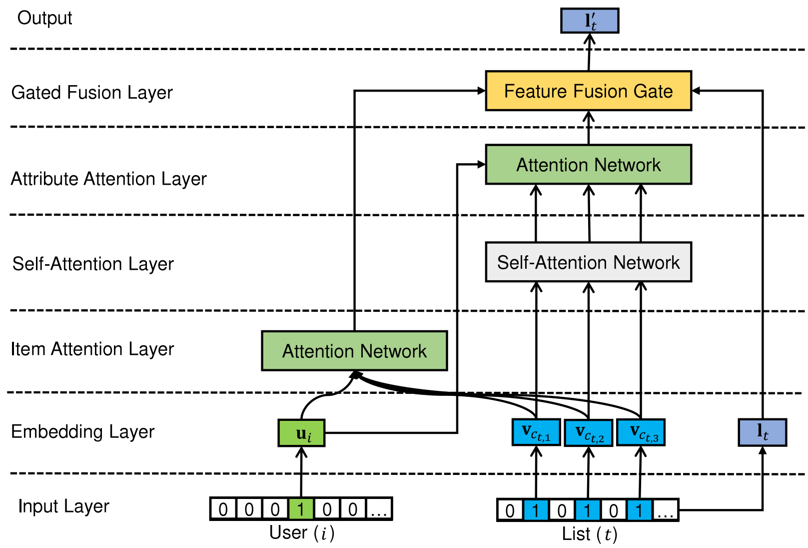


Fig. 2. Illustration of the proposed Multi-representation Learning Network.

$\mathbb{R}^{n\times k}$, the scaled dot-product attention computes the attention scores based on the following equation [35]:

$$Attention(\mathbf{Q},\mathbf{K},\mathbf{V}) = softmax(\frac{\mathbf{QK}^T}{\sqrt{d}})\mathbf{V}, \tag{3}$$

where d is the number of hidden units in the network.

To capture the inherent relationship among items within each list, we first group the items embeddings $\mathbf{v}_{c_{t,1}}, \mathbf{v}_{c_{t,2}}, \ldots, \mathbf{v}_{c_{t,|l_t|}}$ as a matrix $\mathbf{X} \in \mathbb{R}^{|l_t|\times k}$. Then, we map the input matrix $\mathbf{X}$ to queries, keys and values matrices by using different linear projections.

$$\begin{cases} \mathbf{Q} = \mathbf{M}_q\mathbf{X} \\ \mathbf{K} = \mathbf{M}_k\mathbf{X}, \\ \mathbf{V} = \mathbf{M}_v\mathbf{X} \end{cases} \tag{4}$$

where $\mathbf{M}_q, \mathbf{M}_k, \mathbf{M}_v \in R^{k\times|l_t|}$ are the learnable linear projection matrix. It is an important part in our self-attention layer, since $\mathbf{X}$ was projected into different space, and they can be used as input of next step to invoke the self-matching patterns, more empirical analyses will be presented in the experiment section.

Then, h parallel heads are employed to focus on different parts of channels of the value vectors. Formally, for the ith head, we denote the learned linear maps by $\mathbf{W}_i^Q \in \mathbb{R}^{k\times k/h}$, $\mathbf{W}_i^K \in \mathbb{R}^{k\times k/h}$, $\mathbf{W}_i^V \in \mathbb{R}^{k\times k/h}$, which correspond to queries, keys and values, respectively. Then, the scaled dot-product attention is used to compute the relevance between queries and keys, and output mixed representations. The formal description is shown below:

$$\mathbf{M}_i = Attention(\mathbf{QW}_i^Q, \mathbf{KW}_i^K, \mathbf{VW}_i^V). \tag{5}$$

Finally, all vectors produced by parallel heads are concatenated together to form a single vector. A linear map is further used to mix different channels from different heads:

$$\begin{cases} \mathbf{M} = Concat(\mathbf{M}_1, \ldots, \mathbf{M}_h) \\ \mathbf{Y} = \mathbf{MW} \end{cases}, \tag{6}$$

where $\mathbf{M} \in \mathbb{R}^{|l_t|\times k}$ and $\mathbf{W} \in \mathbb{R}^{k\times k}$.

Compared with existing methods which capture item–item relationship by using item positions in the list [22] or local window [43], the self-attention mechanism in this paper shows the following advantages: (1) it can encode more signals without having to consider the order information of the items in the list; (2) it considers the correlation among items in the list, which can capture the global information; and (3) its output matrix is meaningful since each vector can represent the superposition of the interactions between the current item and other items, i.e., it can be regarded as an attribute of a list. More empirical analyses will be presented in the experiment section.

3.2.3. Attribute attention layer

Above the self-attention layer, we propose an attribute attention layer, which targets at extracting a user's preferences over each list based on list attributes. The output matrix $\mathbf{Y}$ of the self-attention network can be divided into $|l_t|$ vectors — $\mathbf{a}_1$, $\mathbf{a}_2$, ..., $\mathbf{a}_{|l_t|}$, which can be seen as the attributes of l. We concatenate each attribute vector with the user embedding $\mathbf{u}_i$ and then design the attribute attention layer as follows:

$$\begin{cases} o'(i,j) = \mathbf{h}_a^T \mathrm{ReLU}(\mathbf{W}_a([\mathbf{u}_i, \mathbf{a}_j]) + \mathbf{b}_a) \\ o(i,j) = \dfrac{\exp(o'(i,j))}{\sum_{j=1}^{|l_t|} \exp(o'(i,j))} \end{cases}, \tag{7}$$

where $\mathbf{W}_a \in \mathbb{R}^{a\times 2k}$ and $\mathbf{b}_a \in \mathbb{R}^a$ denote the weight matrix and bias vector of the hidden layer, respectively. a controls the size of the hidden layer. The vector $\mathbf{h}_a \in \mathbb{R}^a$ projects the hidden layer to the attentive weight for output. Hence, a user's preferences over a list based on list attributes can be calculated as follows:

$$\mathbf{p}_{i,t} = \sum_{j=1}^{|l_t|} o(i,j)\mathbf{a}_j. \tag{8}$$

3.2.4. Gated fusion layer

Based on the learned item-level embeddings, attribute-level embeddings, and list-level embeddings, we can sew them up using various techniques, e.g., element-wise production [2,44], concatenation [18,45] and summation [6,46]. However, the above designs show the following limitations: (1) item-level attention feature and attribute-level attention feature are learned from the same list, which may contain redundant information and (2) the informative part and meaningless part in the features cannot be distinguished.

Targeting at the above limitations, we propose a gated fusion method which can be formally described as follows:

$$\begin{cases} \mathbf{f} = \tanh(\mathbf{W}_{g1}[\mathbf{e}_{i,t}; \mathbf{p}_{i,t}; \mathbf{l}_t] + \mathbf{b}_{g1}) \\ \mathbf{g} = \mathrm{sigmoid}(\mathbf{W}_{g2}[\mathbf{e}_{i,t}; \mathbf{p}_{i,t}; \mathbf{l}_t] + \mathbf{b}_{g2}), \\ \mathbf{l}'_t = \mathbf{g} \odot \mathbf{f} + (1-\mathbf{g}) \odot \mathbf{l}_t \end{cases} \tag{9}$$

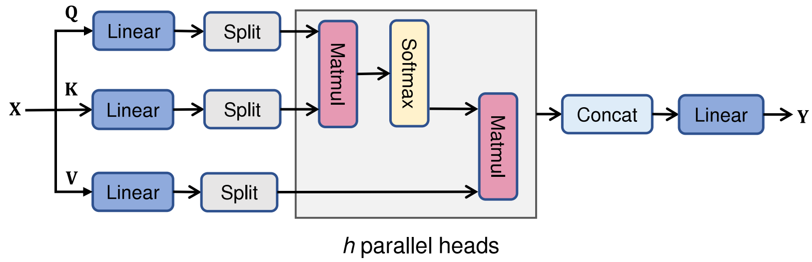


Fig. 3. Architecture of the self-attention network used in this paper.

where $\mathbf{W}_{g1}, \mathbf{W}_{g2} \in \mathbb{R}^{k\times 3k}$ and $\mathbf{b}_{g1}, \mathbf{b}_{g2} \in \mathbb{R}^{k}$ denote the parameter matrices and vectors, respectively. $\mathbf{e}_{i,t}$, $\mathbf{p}_{i,t}$ and $\mathbf{l}_t$ are the user's preferences over item vector, the user's preferences over attribute vector, and the original list embedding, respectively. $\odot$ denotes the element-wise production, $\mathbf{g}$ is the gate applied to the feature vectors, and $\mathbf{l}'_t \in \mathbb{R}^k$ is the multi-level list embeddings which will be fed into the shared interaction learning network.

3.3. Shared interaction learning network

The proposed shared interaction learning network is based on the outer product-based neural collaborative filtering (ONCF) framework [32]. ONCF uses an outer product operation on user embeddings and item embeddings to obtain the interaction map, and then feeds the interaction map into a dedicated neural network (e.g., CNN and MLP) to learn the interaction function. The ONCF framework has shown great potential in fitting data and is more flexible than the inner product-based model (e.g., matrix factorization). However, it can only learn either the user–item interaction or the user–list interaction. Therefore, we extend the ONCF method and propose a shared interaction learning network, which can perform simultaneous learning on the user–item interaction function and user–list interaction function.

Fig. 4 shows the architecture of the proposed shared interaction learning network, which is designed to learn the user–item and user–list interaction functions together. More specifically, given a user–item pair (u_i, v_j) or a user–list pair (u_i, l_t), the representation layer first returns the embedding vector for each given entity. Then, the user–item or user–list embeddings are connected by outer production and then the interaction map is fed into the CNN Layers (shared by the two tasks) to obtain the recommendation score.

3.3.1. Interaction map

Assuming that the input is a user–list pair (u_i, l_t), we perform an outer production on their embeddings to obtain the interaction map:

$$\mathbf{H}_0 = \mathbf{u}_i \otimes \mathbf{l}'_t = \mathbf{u}_i \mathbf{l}'^{T}_t, \tag{10}$$

where $\mathbf{H}_0$ is a $k \times k$ matrix, in which each element is evaluated as: $h_{k_1,k_2} = u_{i,k2} l'_{t,k1}$. Using outer production is more beneficial due to the following two reasons: (1) it subsumes matrix factorization (MF) if only considering the diagonal elements in the output matrix; and (2) it models pairwise correlations between embedding dimensions, which provides the model with more signals.

3.3.2. Shared CNN layers

Above the interaction map, we place a stack of CNN layers which can enable the model to capture the nonlinear and higher-order correlations among users, items, and lists. The formal descriptions are as follows:

$$\begin{cases} \mathbf{H}_1 = \mathrm{ReLU}(Conv2d(\mathbf{H}_0)) \\ \mathbf{H}_2 = \mathrm{ReLU}(Conv3d(\mathbf{H}_1)) \\ \ldots\ldots \\ \mathbf{H}_x = \mathrm{ReLU}(Conv3d(\mathbf{H}_{x-1})) \end{cases}, \tag{11}$$

where $Conv2d$ means applying convolution operation over a 2D matrix, $Conv3d$ means applying convolution operation over a 3D tensor. Since the interaction map $\mathbf{H}_0 \in \mathbb{R}^{64\times 64}$ is a matrix, we use 32 convolutional networks of size 2×2 and stride 2 over $\mathbf{H}_0$ to get feature map $\mathbf{H}_1 \in \mathbb{R}^{32\times 32\times 32}$. After that, another 32 convolutional networks of size 2×2 and stride 2 over $\mathbf{H}_1$ are employed to get the feature map $\mathbf{H}_2 \in \mathbb{R}^{32\times 16\times 16}$. Following the same scheme, the output of the last CNN layer is $\mathbf{H}_6 \in \mathbb{R}^{32\times 1\times 1}$.

After obtaining the output of the last CNN layer — $\mathbf{H}_x$ which can be seen as a vector, we can project it to the final recommendation score as follows:

$$\begin{cases} \hat{y}_{ij} = \mathbf{w}^T \mathbf{H}_x, \ \ if \ \ \mathbf{H}_0 = \mathbf{u}_i \otimes \mathbf{v}_j \\ \hat{r}_{it} = \mathbf{w}^T \mathbf{H}_x, \ \ if \ \ \mathbf{H}_0 = \mathbf{u}_i \otimes \mathbf{l}_t \end{cases}, \tag{12}$$

where $\mathbf{w}$ denotes the weights of the prediction layer; $\hat{y}_{ij}$ and $\hat{r}_{it}$ represent the prediction for a user–item pair (u_i, v_j) and a user–list pair (u_i, l_t), respectively.

It is worth mentioning that we propose to share the same CNN layers of two recommendation tasks in our design. The main reasons behind this design are: (1) the list embedding is aggregated from item embeddings, which makes them in the same semantic space by nature and (2) it can be regarded as a multi task feature learning method, which can facilitate the two recommendation tasks by feature sharing [47]. More empirical analyses will be presented in the experiment section.

3.3.3. Optimization

User generated list recommendation is a standard top-N recommendation task. To this end, we consider learning parameters of GANCF with a ranking-based objective. Hence, we opt for Bayesian Personalized Ranking (BPR) [48] objective function, in which we assume the observed user–list interactions should be ranked higher than the unobserved ones. The loss function can be formally described as follows:

$$\begin{cases} L_{user--item} = \sum_{i,j,j' \in O} -ln\sigma(\hat{y}_{ij} - \hat{y}_{ij'}) + \lambda\|\theta\|_{l_2} \\ L_{user--list} = \sum_{i,t,t' \in O'} -ln\sigma(\hat{r}_{it} - \hat{r}_{it'}) + \lambda\|\theta\|_{l_2} \end{cases},$$

where O and O' denote the user–item training set and user–list training set, θ denotes the model parameters, σ is the sigmoid function and λ is the L_2 regularization coefficient. In the training set, each example is a triplet (i, j, j') or (i, t, t'), which means that user u_i has interacted with item v_j or list l_t, but has not interacted with item v'_j or list l'_t before.

4. Experiments

In this section, we conduct extensive experiments to answer the following research questions:

RQ1 Can the proposed GANCF method outperform state-of-the-art item and list recommendation methods?

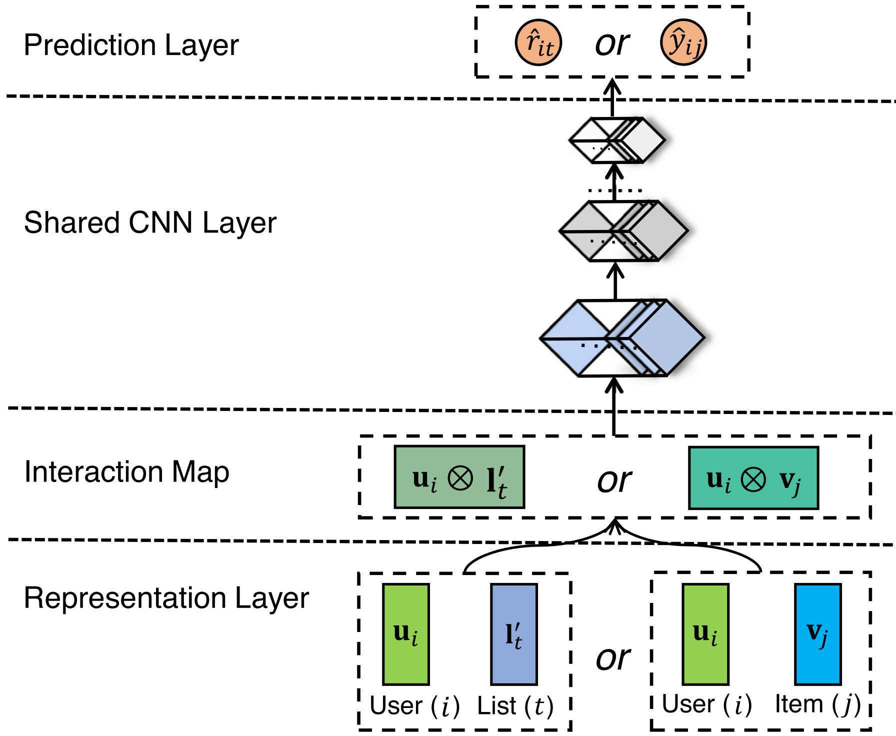


Fig. 4. Illustration of the proposed shared interaction learning network.

RQ2 Can the self-attention network effectively learn the latent attribute of lists?

RQ3 How effective is the gate mechanism as compared with other aggregation strategies?

4.1. Dataset

We evaluate the proposed method using two real-world datasets — one is crawled from an online social platform Douban and the other is a publicly available music dataset from Netease. The details of the two datasets are shown below.

4.1.1. Douban

This dataset contains movie watching data crawled from Douban,[5] which enables users to select their interested movies or user generated movie watching lists. We retained users consuming at least 10 items and 10 lists. The final dataset contains 14,864 users, 17,008 items, 10,816 lists, 1, 638, 393 user–item interactions, and 775,877 user–list interactions.

4.1.2. Netease

This is a music data from Netease, constructed by [5] for list recommendation. To ensure high data quality, we processed the dataset by retaining users consuming at least 10 items and 10 lists. The final dataset contains 16,047 users, 41,923 items, 19,265 lists, 907,667 user–item interactions, and 460,011 user–list interactions.

4.2. Evaluation criteria

We adopt the leave-one-out evaluation protocol, which has been widely used to evaluate the performance of top-N recommendation [18,49]. Specifically,for each user in the dataset, we hold out his/her latest item (list) interaction as the testing positive sample, and then pair it with 99 items (lists) that the user did not consume before as the negative samples. We adopt the random sample strategy to select the negative samples [50]. Each method then generates predictions for these 100 user–item (user–list) interactions. To evaluate the performance, we adopted two popular metrics — Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). HR@k is a recall-based metric, measuring whether the testing item is located within the top-k position (1 for yes and 0 otherwise). NDCG@k assigns higher scores to the items within the top-k positions of the ranking list.

As we all know, it is not enough to judge the quality of a recommender system simply by measuring accuracy. To make a comprehensive understanding of our model, we would measure the serendipity, diversity, and novelty of our model. Following [51,52], the definition of these metrics are shown below.

First, the serendipity is calculated by

$$Serendipity(R) = \frac{||R \backslash R_{unexp}| \bigcap R_{useful}|}{|R|}, \tag{13}$$

where R is the set of recommendation items (lists) generated for the user, R_{unexp} is the items (lists) set generated by the primitive model, which means the items (lists) recommended to a user is unexpected for this user., and R_{useful} means the items (lists) that the user consumed. The higher scores denote higher serendipity of recommendation result.

Second, the diversity is measured by

$$Diversity(R) = \frac{\sum_{i \in R} \sum_{j \in R \backslash \{i\}} dist(i,j)}{|R|(|R-1|)}, \tag{14}$$

where R is the set of recommended items (lists) and $dist(i,j)$ is the distance between items (lists), because our model follows the representation learning paradigm, the distance could be measured by cosine similarity. The higher scores denote lower diversity of recommendation result.

Third, the novelty is given by

$$Novelty(R) = \frac{\sum_{i \in R} -log_2 p(i)}{|R|}, \tag{15}$$

[5] http://movie.douban.com

where R is the set of recommended items (lists), due to novel items are identified with the long tail items, that is, the part of the items are only viewed or purchased by a small part of the users. Therefore, the $p(i)$ is typically defined as the item's popularity in the dataset. The higher scores denote higher novelty of recommendation result.

4.3. Implementation details

We implemented our method based on PyTorch.[6] For hyper-parameter tuning, we randomly sampled one interaction with items and one interaction with lists for each user as the validation set. For the initialization of the embedding layer, we randomly initialized their parameters with a Gaussian distribution — $\mathcal{N}(0, 0.1)$. For the shared CNN layers, we used the kaiming initialization strategy [53]. We used the Adam optimizer for all gradient-based methods, where the mini-batch size and learning rate were searched in $[128, 256, 512, 1024]$ and $[0.001, 0.005, 0.01, 0.05, 0.1]$, respectively. For fair comparison, we set the embedding size as 64 for all methods. For the attention network, we empirically set the size of the first hidden layer with the dimension of 32 which is the half of the embedding size, and choose ReLU as the activation function. For self-attention network, we set the number of multi-heads as 8 which is a default setting in [35]. For serendipity metric, we choose the MF-BPR as our primitive model. We repeated each experiment 10 times and reported the average results.

4.4. Compared methods

We compare GANCF with the state-of-the-art item recommendation methods, list recommendation methods and different variants of our method described as follows:

4.4.1. State-of-the-art methods

We compare GANCF with existing state-of-the-art methods on both item recommendation and list recommendation as follows.

- **MF-BPR** [48], which optimizes the standard MF model with the pairwise ranking loss. It performs item recommendation and list recommendation by leveraging user–item and user–list interactions, respectively.
- **LIRE** [22], which considers users' previous interactions with both individual items and user generated lists. It weights items within lists based on both position of items and personalized list consumption pattern. It also applies the BPR loss for optimization and is a strong competitor in jointly recommending lists and their contained items.
- **EMF-joint** [5], which combines factorization methods and embedding-based methods. It treats the lists as sentences and items as words in language models. It also applies the BPR loss for optimization and is a state-of-the-art method in jointly recommending lists and items.
- **NeuMF** [18] is the state-of-the-art deep neural network method. It stacks multiple fully connected layers above the inner products of feature embeddings to capture higher-order and nonlinear cross features. user–item interactions and user–list interactions are utilized to perform the item recommendation and list recommendation, respectively.
- **ConvNCF** [32] is a recent work that exploits the outer production between user and item embeddings for representation learning. It uses multiple stacked 2D convolutional layers for learning features from the outer product matrix. Item recommendation and list recommendation are conducted by utilizing user–item interactions and user–list interactions, respectively.

[6] http://www.pytorch.org

Table 1
Top-N recommendation accuracies of both item recommendation and list recommendation on Douban dataset.

	Douban			
	Item		List	
	HR@10	NDCG@10	HR@10	NDCG@10
MF-BPR	0.4741	0.2687	0.4359	0.3018
LIRE	0.4831	0.2742	0.4428	0.3116
EMF-joint	0.4910	0.2816	0.4491	0.3271
NeuMF	0.5009	0.2831	0.4521	0.3596
ConvNCF	0.5093	0.2898	0.4586	0.3692
GANCF-single	0.5135	0.2967	0.4600	0.3704
GANCF-avg	0.5179	0.3017	0.4614	0.3721
GANCF	**0.5259***	**0.3129***	**0.4688***	**0.3803***

*Indicates that the improvements over all other methods are statistically significant for $p < 0.05$.

Table 2
Top-N recommendation accuracies of both item recommendation and list recommendation on Netease dataset.

	Netease			
	Item		List	
	HR@10	NDCG@10	HR@10	NDCG@10
MF-BPR	0.5150	0.2923	0.4487	0.2601
LIRE	0.5251	0.3043	0.4568	0.2628
EMF-joint	0.5312	0.3102	0.4691	0.2683
NeuMF	0.5400	0.3183	0.4774	0.2703
ConvNCF	0.5548	0.3211	0.4828	0.2793
GANCF-single	0.5595	0.3264	0.4890	0.2803
GANCF-avg	0.5644	0.3301	0.4914	0.2851
GANCF	**0.5716***	**0.3345***	**0.4987***	**0.2889***

*Indicates that the improvements over all other methods are statistically significant for $p < 0.05$.

4.4.2. Variant models

To analyze the effectiveness of attention mechanism and shared CNN layers in GANCF, we propose the following variants of GANCF.

- **GANCF-single** uses two separated CNN layers to learn user–item interactions and user–list interactions, respectively, which reduces to the ConvNCF method.
- **GANCF-avg** removes the attribute attention network and employs the average strategy to get the user's preferences over items and attributes, i.e., each item or attribute contributes equally to the user's preferences.

4.5. Recommendation performance comparison (RQ1)

Tables 1 and 2 show the performance comparison w.r.t. HR@10 and NDCG@10 on the two real-world datasets. From these tables, we have the following observations:

(1) Our GANCF model achieves the best performance on the two datasets for both tasks. In fact, the result shows improvements over MF-BPR, LIRE, EMF-joint, NeuMF, ConvNCF of 10.96%, 8.86%, 7.11%, 5.00% 3.26% in HR@10, and 16.45%, 14.11%, 11.11%, 10.52% 8.07% in NDCG@10 respectively in item recommendation task. Meanwhile in the list recommendation task, the improvement are also significant. This validates the effectiveness of the proposed GANCF solution.

(2) The performance of neural network-based solutions (i.e., NeuMF, ConvNCF, GANCF-single, GANCF-avg, GANCF) are superior to that of factorization based approaches LIRE, MF-BPR and embedding hybrid model EMF-joint. This demonstrates the superiority of neural networks, especially their great ability in modeling the high-order interactions among users, lists, and items.

Table 3
Top-N recommendation diversity, serendipity, novelty, of both item recommendation and list recommendation on Douban dataset. The best result for each metric is highlighted in bold. All the p-values between our model and each of the baselines are much smaller than $p < 0.05$.

	Douban					
	Item			List		
	Serendipity	Diversity	Novelty	Serendipity	Diversity	Novelty
MF-BPR	–	8.445	5.605	–	8.055	5.322
LIRE	0.0117	7.201	8.294	0.0105	6.199	7.621
EMF-joint	0.0192	6.996	**9.21**	0.0139	6.064	**8.33**
NeuMF	0.0289	6.055	5.136	0.0196	7.705	5.711
ConvNCF	0.0352	6.089	4.605	0.0281	6.895	5.351
GANCF-single	0.0394	6.123	4.681	0.029	5.421	5.217
GANCF-avg	0.0441	6.141	4.598	0.0311	5.229	4.993
GANCF	**0.0501**	**6.038**	4.207	**0.0329**	**5.11**	4.829

Table 4
Top-N recommendation diversity, serendipity, novelty, of both item recommendation and list recommendation on Netease dataset. The best result for each metric is highlighted in bold. All the p-values between our model and each of the baselines are much smaller than $p < 0.05$.

	Netease					
	Item			List		
	Serendipity	Diversity	Novelty	Serendipity	Diversity	Novelty
MF-BPR	–	9.917	7.211	–	8.543	5.871
LIRE	0.0089	8.716	8.317	0.012	5.338	7.513
EMF-joint	0.0213	6.394	**10.593**	0.0235	5.571	**7.993**
NeuMF	0.0276	**5.021**	6.33	0.0451	6.821	6.552
ConvNCF	0.0343	5.105	5.122	0.069	6.736	5.917
GANCF-single	0.0417	5.117	4.819	0.0731	5.493	5.022
GANCF-avg	0.0439	5.109	4.502	0.0846	5.229	4.686
GANCF	**0.0512**	5.038	4.667	**0.0967**	**4.987**	4.833

(3) the performance of GANCF is superior to that of GANCF-avg, which indicates that the items within a list are indeed not equally important when consumed by different users and this kind of variances can influence user's preference over list. For example, A fan of Comedy movies may collect the movie list named "Chinese Kungfu Films", but a fan of war movies may not collect this movie list, because many Chinese Kungfu films have comedy elements but lack of war elements.

(4) The experimental results of GANCF is superior to GANCF-single. This admits the significance of modeling user–item and user–list interaction in the parameter shared network. Since, the list embedding is aggregated from item embeddings, which makes them in the same semantic space by nature. Moreover, this can augment the training of user–list interaction function with user–item interaction data and vice versa, which facilitates the two tasks reinforcing each other.

(5) Compared with ConvNCF, GANCF-single achieves higher accuracy on both datasets with respect to both metrics. The statistically significant improvements confirm that better representations can indeed be learned by the proposed representation learning network. Furthermore, it indicates that the earlier user, item and list interaction (before feed into the interaction learning network) the more representative embedding vector we can get.

Tables 3 and 4 show the performance comparison w.r.t. serendipity, diversity and novelty on the two real-world datasets. From these tables, we have the following observations:

(1) Our GANCF model achieves the best performance in serendipity on the two datasets for both tasks. It means that neural network-based methods outperform factorization based approaches and embedding hybrid models once again. This demonstrates that neural networks may help the recommender systems to find some items (lists) to make user feel "surprise" .

(2) There is no obvious winner in item diversity on both datasets. This is mainly because, the distance between items measured by cosine similarity and different methods have different embedding vectors, which would cause the various result. However, we can find that in list diversity on both datasets, GANCF, EMF-joint and LIRE perform better than ConvNCF, NeuMF, MF-BPR. It might be because GANCF, LIRE and EMF-joint could utilize the relationship among items within a list, however, convNCF and NeuMF neglect these information. It proves for list recommendation, considering the correlation between items within a list could improve the diversity of recommending results.

(3) We could find that EMF-joint achieves the best performance and our GANCF solution gets bad result in the novelty metric on the two datasets for both tasks. One plausible explanation is that our method belongs to collaborative filtering-based method. This method would recommend items to user by capturing user–item interaction information, which means the more user–item interactions, the more accurate recommendation results. However, for the long-tail items, there is not much user–item interaction, our model does not perform well on this metric. In fact, this problem belongs to cold-start item problem in recommender systems, however, this paper does not aim to address this problem, we would leave this study in our future work.

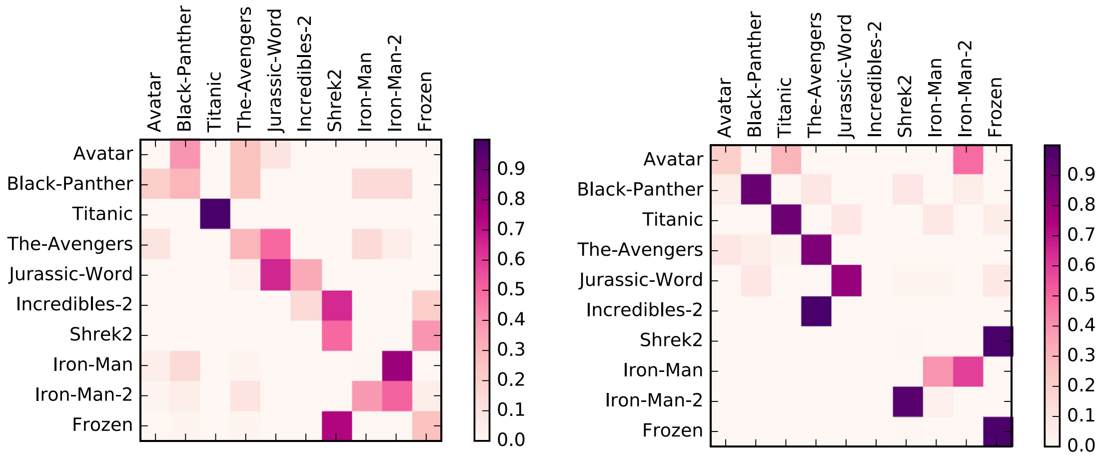
4.6. Self-attention analysis (RQ2)

To better understand the effectiveness of the self-attention mechanism, we conduct a qualitative experiment here. More specifically, we select one list from Douban dataset as an example in this case study. The items in the list are showed in Fig. 5, and we visualized the self-attention weights in self-attention layer on head 4 and head 5 by two heatmaps.

We can see from the results that (1) different heads have different self-attention weights. This is intuitive because item embeddings are projected into different subspaces in the self-attention layer, which would discover different self-matching patterns. Just like people looking in the mirror, from the different angle, what they would see in the mirror is different. Meanwhile, this also proves the benefits of utilizing self-attention. According to aggregating the self-matching patterns in different subspaces, we can get a more comprehensive list representation. (2) Self-attention is capable of finding long-range relationships, i.e., the relationship between two items can be captured without the distance constraints. For instance, in Fig. 5(b), Avatar has a strong impact on Iron-Man-2 which is far away from Avatar within the list. This kind of property indicate that the global information in the list can be captured. Furthermore, This empirical evidence validates our motivation on utilizing self-attention within a list. (3) From Fig. 5(a), we can find that Avatar, Black Panther and The Avengers are close to each other, it is mainly because all these films are belong to science fiction type, and some user may put them together to form a list in nature. This empirical evidence indicate that when armed with self-attention, neural network have the ability to discover some inner relationship, i.e., item latent attributes can be captured in our list recommendation task.

4.7. Effect of gate mechanism (RQ3)

The recommendation performance comparison indicates that GANCF obtains the best accuracy, demonstrating the effectiveness of the integrated end-to-end solution. To further understand the effectiveness of the feature fusion strategy in learning list embeddings, we make minor changes on GANCF. More specifically, we replace Gated Fusion Layer with other operations, e.g., summation, element-wise production and concatenation. For notational convenience, we use GANCF-S to denote the method "GANCF



(a) Visualization on head 4 (b) Visualization on head 5

Fig. 5. An example of different self-attention heads, which shows that item–item relationship can be captured.

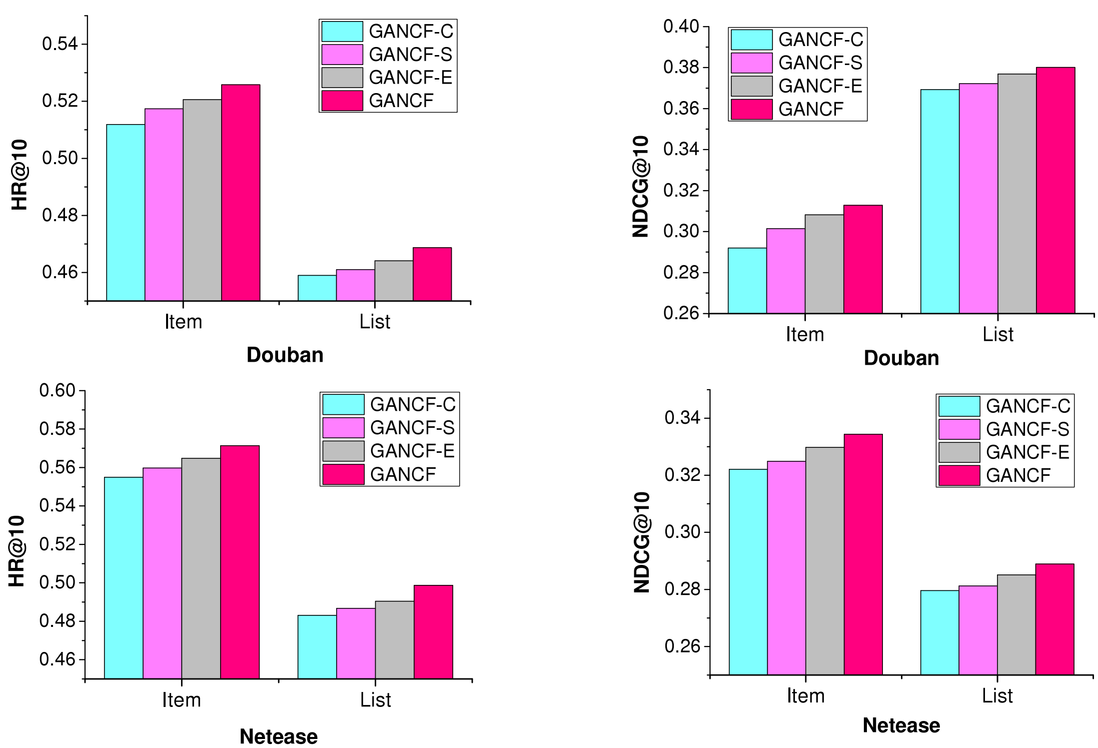


Fig. 6. The performance comparison of different aggregation strategies (GANCF-S, GANCF-E, GANCF-C, and GANCF) w.r.t. HR@10 and NDCG@10 on both datasets.

with summation", GANCF-E to denote the method "GANCF with element-wise production", and GANCF-C to denote "GANCF with concatenation".

Fig. 6 shows the results of GANCF and the other three variants. We can see from the results that

(1) GANCF outperforms other variants by a large margin on both datasets, confirming the effectiveness of the gated mechanism. More specifically, the positive effect of gate mechanism in filtering the redundant information at the fusion stage. The more intuitive explanation is that the gate mechanism can be seen as a gatekeeper, each feature go through this gate would be checked, then it would keep the informative part and drop the meaningless part in these feature automatically. (2) The performances of GANCF-S and GANCF-E are superior to that of GANCF-C. It is mainly because GANCF-C just concatenate each feature directly, however, GANCF-E and GANCF-S have some interaction among each feature. It demonstrates that even utilize some simple method to explicitly modeling the correlations among different features are useful. In summary, the gated mechanism is beneficial to the targeted list embedding problem. This empirical evidence provides support for our motivation of utilizing gated mechanism.

5. Conclusion and future work

This work addresses the user generated list recommendation problem from the perspective of neural representation learning, in which we mainly focus on two key problems: (1) how to obtain semantically meaningful representations of the user generated lists and (2) how to model the interactions between the users and the lists. We propose a neural network-based solution named GANCF, which addresses the list representation learning problem with a multi-representation learning network and addresses the interaction learning problem with a shared interaction learning network. Specifically, by leveraging the multi-representation learning network, we can learn the list representation, item representation and user representation simultaneously; by leveraging shared interaction learning network, it is capable of learning the complicated interactions among users, items, and lists. Extensive experiments on two real-world datasets demonstrate that the proposed representation learning network can learn more representative user/item/list embeddings than existing methods and GANCF can outperform state-of-the-art methods in both item recommendation and list recommendation in terms of accuracy.

In the future, we plan to extend our work in the following two directions. First, we will explore the utility of social network [54,55], since it contains valuable signal on how users share their generated lists to their friends and how users trust each other. Second, we are interested in realizing list recommender systems in an online fashion. The interests of user on items evolve over time, and so do the interests of lists. As it is computationally prohibitive to retrain a recommender model in real-time, it would be extremely helpful to do online learning [56,57]. Along this line, we are particularly interested in leveraging reinforcement learning methods to provide online recommendation.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 61702176 and Hunan Provincial Natural Science Foundation of China under Grant No. 2017JJ3038.

References

[1] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, K. Gai, Deep interest network for click-through rate prediction, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2018, pp. 1059–1068.

[2] X. Wang, X. He, L. Nie, T.-S. Chua, Item silk road: Recommending items from information domains to social users, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2017, pp. 185–194.

[3] H. Wang, F. Zhang, X. Xie, M. Guo, Dkn: Deep knowledge-aware network for news recommendation, in: Proceedings of the 2018 World Wide Web Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2018, pp. 1835–1844.

[4] T. Zhu, P. Harrington, J. Li, L. Tang, Bundle recommendation in ecommerce, in: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM, 2014, pp. 657–666.

[5] D. Cao, L. Nie, X. He, X. Wei, S. Zhu, T.-S. Chua, Embedding factorization models for jointly recommending items and user generated lists, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2017, pp. 585–594.

[6] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2008, pp. 426–434.

[7] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, ACM, 2001, pp. 285–295.

[8] M. Mao, J. Lu, G. Zhang, J. Zhang, Multirelational social recommendations via multigraph ranking, IEEE Trans. Cybern. 47 (12) (2016) 4049–4061.

[9] Q. Zhang, D. Wu, J. Lu, G. Zhang, Cross-domain recommendation with probabilistic knowledge transfer, in: International Conference on Neural Information Processing, Springer, 2018, pp. 208–219.

[10] X. Yu, Y. Chu, F. Jiang, Y. Guo, D. Gong, SVMs classification based two-side cross domain collaborative filtering by inferring intrinsic user and item features, Knowl.-Based Syst. 141 (2018) 80–91.

[11] J. McAuley, J. Leskovec, Hidden factors and hidden topics: understanding rating dimensions with review text, in: Proceedings of the 7th ACM Conference on Recommender Systems, ACM, 2013, pp. 165–172.

[12] L. Shi, W.X. Zhao, Y.-D. Shen, Local representative-based matrix factorization for cold-start recommendation, ACM Trans. Inf. Syst. 36 (2) (2017) 22.

[13] Y. Sun, N.J. Yuan, X. Xie, K. McDonald, R. Zhang, Collaborative intent prediction with real-time contextual data, ACM Trans. Inf. Syst. 35 (4) (2017) 30.

[14] W. Wang, G. Zhang, J. Lu, Member contribution-based group recommender system, Decis. Support Syst. 87 (2016) 80–93.

[15] X. Li, M. Jiang, H. Hong, L. Liao, A time-aware personalized point-of-interest recommendation via high-order tensor factorization, ACM Trans. Inf. Syst. 35 (4) (2017) 31.

[16] I. Bayer, X. He, B. Kanagal, S. Rendle, A generic coordinate descent framework for learning from implicit feedback, in: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 1341–1350.

[17] R. He, J. McAuley, VBPR: Visual Bayesian personalized ranking from implicit feedback, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, AAAI, 2016, pp. 144–150.

[18] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 173–182.

[19] X. He, H. Zhang, M.-Y. Kan, T.-S. Chua, Fast matrix factorization for online recommendation with implicit feedback, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2016, pp. 549–558.

[20] S. Kabbur, X. Ning, G. Karypis, Fism: factored item similarity models for top-n recommender systems, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, pp. 659–667.

[21] M. Polato, F. Aiolli, Boolean kernels for collaborative filtering in top-N item recommendation, Neurocomputing 286 (2018) 214–225.

[22] Y. Liu, M. Xie, L.V. Lakshmanan, Recommending user generated item lists, in: Proceedings of the 8th ACM Conference on Recommender Systems, ACM, 2014, pp. 185–192.

[23] O. Sar Shalom, N. Koenigstein, U. Paquet, H.P. Vanchinathan, Beyond collaborative filtering: The list recommendation problem, in: Proceedings of the 25th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2016, pp. 63–72.

[24] C. Wu, J. Wang, J. Liu, W. Liu, Recurrent neural network based recommendation for time heterogeneous feedback, Knowl.-Based Syst. 109 (2016) 90–103.

[25] X. Chen, Y. Zhang, Q. Ai, H. Xu, J. Yan, Z. Qin, Personalized key frame recommendation, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2017, pp. 315–324.

[26] A. Van den Oord, S. Dieleman, B. Schrauwen, Deep content-based music recommendation, in: Advances in Neural Information Processing Systems, 2013, pp. 2643–2651.

[27] F. Zhang, N.J. Yuan, D. Lian, X. Xie, W.-Y. Ma, Collaborative knowledge base embedding for recommender systems, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 353–362.

[28] R. Yin, K. Li, J. Lu, G. Zhang, Enhancing fashion recommendation with visual compatibility relationship, in: The World Wide Web Conference, ACM, 2019, pp. 3434–3440.

[29] X. Geng, H. Zhang, J. Bian, T.-S. Chua, Learning image and user features for recommendation in social networks, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2015, pp. 4274–4282.

[30] T. Alashkar, S. Jiang, S. Wang, Y. Fu, Examples-rules guided deep neural network for makeup recommendation, in: Proceedings of the 31th AAAI Conference on Artificial Intelligence, AAAI, 2017, pp. 941–947.

[31] W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, Z. Qin, Aesthetic-based clothing recommendation, in: Proceedings of the 2018 World Wide Web Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2018, pp. 649–658.

[32] X. He, X. Du, X. Wang, F. Tian, J. Tang, T.-S. Chua, Outer product-based neural collaborative filtering, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 2227–2233.

[33] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.

[34] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, T.-S. Chua, Attentional factorization machines: Learning the weight of feature interactions via attention networks, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3119–3125.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[36] T. Shen, T. Zhou, G. Long, J. Jiang, C. Zhang, Bi-directional block self-attention for fast and memory-efficient sequence modeling, in: ICLR, 2018.

[37] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, C. Zhang, Disan: Directional self-attention network for RNN/CNN-free language understanding, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, AAAI, 2018, pp. 5446–5455.

[38] Z. Tan, M. Wang, J. Xie, Y. Chen, X. Shi, Deep semantic role labeling with self-attention, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, AAAI, 2018, pp. 4929–4936.

[39] M.T. Luong, H. Pham, C.D. Manning, Effective approaches to attention-based neural machine translation, 2015, arXiv preprint arXiv:1508.04025.

[40] C. Yang, M. Jiang, B. Jiang, W. Zhou, K. Li, Co-attention network with question type for visual question answering, IEEE Access 7 (2019) 40771–40781.

[41] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: ICLR, 2015.

[42] C. Yang, H. Zhang, B. Jiang, K. Li, Aspect-based sentiment analysis with alternating coattention networks, Inf. Process. Manage. 56 (3) (2019) 463–478.

[43] D. Liang, J. Altosaar, L. Charlin, D.M. Blei, Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence, in: Proceedings of the 10th ACM Conference on Recommender Systems, ACM, 2016, pp. 59–66.

[44] Y. Zhang, Q. Ai, X. Chen, W.B. Croft, Joint representation learning for top-n recommendation with heterogeneous information sources, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, ACM, 2017, pp. 1449–1458.

[45] T. Bai, J.-R. Wen, J. Zhang, W.X. Zhao, A neural collaborative filtering model with interaction-based neighborhood, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, ACM, 2017, pp. 1979–1982.

[46] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, T.-S. Chua, Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2017, pp. 335–344.

[47] P. Gong, J. Ye, C. Zhang, Multi-stage multi-task feature learning, J. Mach. Learn. Res. 14 (1) (2013) 2979–3010.

[48] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, AUAI Press, 2009, pp. 452–461.

[49] J. Huang, W.X. Zhao, H. Dou, J.-R. Wen, E.Y. Chang, Improving sequential recommendation with knowledge-enhanced memory networks, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM, 2018, pp. 505–514.

[50] D. Cao, X. He, L. Miao, Y. An, C. Yang, R. Hong, Attentive group recommendation, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM, 2018, pp. 645–654.

[51] D. Kotkov, S. Wang, J. Veijalainen, A survey of serendipity in recommender systems, Knowl.-Based Syst. 111 (2016) 180–192.

[52] M. Kaminskas, D. Bridge, Diversity, serendipity, novelty, and coverage: A survey and empirical analysis of beyond-accuracy objectives in recommender systems, ACM Trans. Interact. Intell. Syst. 7 (1) (2017) 2.

[53] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2015, pp. 1026–1034.

[54] Z. Cheng, J. Shen, Just-for-me: an adaptive personalization system for location-aware social music recommendation, in: Proceedings of International Conference on Multimedia Retrieval, ACM, 2014, pp. 185–193.

[55] X. Song, L. Nie, L. Zhang, M. Akbari, T.-S. Chua, Multiple social network learning and its application in volunteerism tendency prediction, in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2015, pp. 213–222.

[56] Z. Cheng, S. Jialie, S.C. Hoi, On effective personalized music retrieval by exploring online user behaviors, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2016, pp. 125–134.

[57] W. Jiang, G. Wang, M.Z.A. Bhuiyan, J. Wu, Understanding graph-based trust evaluation in online social networks: methodologies and challenges, ACM Comput. Surv. 49 (1) (2016) 10.