IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 14, 2018, accepted January 4, 2019, date of publication January 11, 2019, date of current version February 6, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2892565

HARSAM: A Hybrid Model for Recommendation Supported by Self-Attention Mechanism

DUNLU PENG, WEIWEI YUAN, AND CONG LIU
School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 20093, China
Corresponding author: Dunlu Peng (pengdl@usst.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772342 and Grant 61703278.

ABSTRACT Collaborative filtering is one of the most commonly used methods in recommendation systems. However, the sparsity of the rating matrix, cold start-up, and most recommendation algorithms only consider the users while neglecting the relationship between the products, all of what limit the effectiveness of the recommendation algorithms. In this paper, based on the self-attention mechanism, a deep learning model, named HARSAM, is proposed for modeling user interaction data and learning the user's latent preference expression. HARSAM partitions the user's latent feedback data in different time granularity and employs the self-attention mechanism to extract the correlation among the data in each partition. Moreover, the model learns the user's latent preferences through the deep neural network. Simultaneously, the model learns the item latent representation by making use of the *stacked denoising autoencoder* to model the item's rating data. As the result, the model recommends items to users according to the similarities between user's preference and items. Experiments conducted on the public data demonstrate the effectiveness of the proposed model.

INDEX TERMS SDAE, self-attention mechanism, preference expression, recommendation system.

I. INTRODUCTION

With the rapid popularization of e-commerce, cloud computing and Internet of Thing, data grows exponentially on the Internet. Currently, information overload has been increasingly serious in many applications. Extracting hidden valuable information efficiently and effectively from the complex data has obviously become a crucial issue in the field of big data. The recommendation system [1]–[3] is an often used approach for recommending useful information to users from massive data. Generally, it exploits a recommendation algorithm to recommend the items to users according to their preferences and needs. In the prospective of the algorithm, the recommended items are regarded as the most interesting ones to users. To a certain extent, recommendation systems alleviate the problem of information overload and improve the efficiency of information retrieval, that has been widely recognized by both academia and industry.

In the literature, traditional recommendation algorithms can be classified into three types: collaborative filtering [4]–[6], content-based recommendation algorithms [7]–[9] and hybrid recommendation algorithms [10], [11]. Comparatively, although collaborative filtering is the most widely used one, it has faced two critical challenges from scratch: sparseness of data and the cold start-up. In addition, collaborative filtering algorithm often uses low-level neural network models, which cannot learn the deep features of users and those of items. The content-based recommendation algorithm usually utilizes artificially designed features to extract the relationship between users and items based on the historical behavioral data. This approach not only limits the effectiveness and extensibility of the recommender system, but also ignores the inter-relationship and interaction between the items.

In recent years, deep learning has made breakthroughs in image processing [12]–[14], natural language processing [15]–[17] and other fields. Compared to the traditional recommendation systems, deep learning method has two distinct advantages. On the one hand, deep learning network is a deep-layered nonlinear network that can better extract the deep features of the data. On the other hand, it maps multiple heterogeneous data into a same hidden space and then obtains a unified representation of data. Starting from the point of alleviating the negative impact of data sparsity and cold start in recommendation systems, and considering the relationship between user's interactive items, this paper presents an algorithm called HARSAM, which combines the advantages of self-attention mechanism and deep learning model together, to learn user preferences. The main contributions of the paper are as follows: 1) In the model of

learning preference representation, self-attention mechanism is introduced to concern the internal relationship between user interaction data in different periods, so that the user's preference expression with timeline can be obtained. 2) By composing the separate latent representation of item's rating data and their feature information, the full latent representation of items is generated. To some extent, it overcomes the difficulties of calculating the similarity between items due to data sparsity. 3) We conducted a series of experiments on publicly available datasets to verify our proposed approach. The exmperimental results demonstrate, compared to the comparative aprroaches, our alogrithm can improve both the effectiveness and performance of recommendation.

The rest of the paper is organized as follows: In Section II, the related research in recent years on the recommendation algorithms is introduced. The explanation of terminologies, feature embedding of user's interactive items, and process of learning item latent representation based on SDAE model, are presented and analyzed in Section III. Section IV-A provides the model of HARSAM, which is used to extract the user's latent preference feature based on self-attention mechanism. In Section V, we verify the effectiveness and performance of the our model with experiments. Finally, the conclusion of the paper is drawn in Section VI.

II. RELATED WORK

In the past few decades, researchers have proposed various recommendation algorithms which are widely applied in recommendation systems. For example, GroupLens [18] introduces an automated collaborative filtering recommendation algorithm based on user ratings for recommending videos and news. A content-based recommendation algorithm, provided by Schafer *et al.* [19], makes recommendation in terms of the matching degree between users and items to be predicted, which is calculated from the users' preference feature representation learned from their interactive item features. The hybrid recommendation algorithm described in [20] fuses the results of collaborative filtering and content-based recommendation algorithm.

With the excellent performance of deep learning in extracting deep features of data, more and more researchers have combined deep learning methods with recommended system tasks. In 2007, Salakhutdinov *et al.* [21] pioneer the application of deep learning to solve the recommendation problem, and propose a restricted Boltzmann machine based recommendation model. Strub *et al.* [22] make use of two *Stacked Denoising AutoEncoders* (SDAE) to learn the latent representations of users and items, respectively, in order to predict the missing ratings for recommendation. Song *et al.* [23] present a time granularity based deep semantics structured model through modeling the user's preferences at different time granularities. Bahdanau *et al.* [24] apply attention mechanism in the task of machine translation, which significantly improves the translating accuracy. Recently, attention mechanisms have been exploited to process different learning tasks, such as reading comprehension, recommendation systems, and so on. For instance, Gong *et al.* [25] develop an attention-based convolutional neural network for Hashtag recommendations in Weibo. Yang *et al.* [26] use the attention mechanism to model the correlation between user's comments and item characteristics, and learn latent expressions from users' comments and item features. Furthermore, Vaswani *et al.* [27] introduce self-attention mechanisms into translation tasks to model the intrinsic relationships between the data, which effectively enhances the accuracy of translation.

In this work, we design HARSAM, a new hybrid recommendation algorithm, which is based on combining the ability of deep neural network model to extract deep features from complex data with that of using self-attention mechanism to extract internal relations between data. This algorithm measures the user preference of an item based on two factors: One is the similarity between the user and the items to be recommended, and the other is the similarity between the items to be predicted and items the user prefers. Our main idea is that, to improve the accuracy of recommendation, it is necessary to consider the relationships between items as well as their mutability, because of external influences. For example, during the World Cup, users will generally care about football-related goods. Taking this into consideration, therefore, in this paper, we model user interaction data at different time intervals in the way of combining deep neural networks with attention mechanisms. Therefore, the latent preferences of users and the latent representation of items are also learned. In the meantime, considering the problem of poor computability between items, the SDAE model is used to learn latent representations of items from the rating data, combined with latent representations extracted from item features, to collectively describe an item. Experimental results show that our approach improves the effectiveness of the item latent representation and the computability of similarity between items as well.

III. PRELIMINARIES

This section describes the preprocessing of user's interactive item, preliminary embedding information of item features, and learning latent representation of items with SDAE on the rating data. Section III-A explains the definitions required in our work. Section III-B presents the way to preprocess user's interactive items and initially embed item feature information. Finally, in Section III-C, a detailed description of learning latent representation of an item by use of SDAE in our model is provided.

A. DEFINITIONS

Before describing our model and algorithm, some definitions about the data are given as follows:

Rating matrix $R_{m\times n}$: the matrix, which is formed by the item ratings given by users in a recommendation system, contains m users and n items. Each element of the matrix, namely, $r_{ij} \in R_{m\times n}$, represents the ith user ratings for the jth item.

TABLE 1. Notations and their meanings.

Symbol	Descriptions
I	the rough embedding representation of interactive items for one user, $I = \{I_1, I_2, \ldots, I_t\}$
$\overrightarrow{item}$	Rough eigenvector of a single item
V_u^*	User interactive items, $V_u^* = \{V_1^*, V_2^*, \ldots, V_m^*\}$
$R_{t_i}^{m\times l}$	Embedding matrix of user's interactive items in time interval t_i
$R_{m\times n}$	Rating matrix of user's interactive items

Interact items: the items that users interact with, such as clicking, downloading, browsing, rating, etc.

Feature information: the information describing an item, such as Id, name and other attributes. For example, in the experiments of this work, the feature information of the items includes a movie's Id, name, released year, and its category.

Table 1 illustrates the notations as well as their meanings in the paper.

B. PREPROCESSING INPUT DATA

In a recommendation system, the interaction data of a user are items that the user interacts with explicit feedback or implicit feedback. Each item can be represented as a binary tuple $\langle O_i, t_i\rangle$, where, O_i is the item and t_i denotes the time of user i when he interacts with item O_i. Considering the variability of user preferences, before initially embedding the item features, the user's interactive items are divided into T intervals according to the interaction time t_i. It is worth noting that the space of each interval is related to the experimentally selected dataset. We use $T = \left(\left[2^0, 2^1\right), \left[2^1, 2^2\right), \ldots, \left[2^{t-1}, 2^t\right)\right)$ to express the division of the item. Correspondingly, after being divided, interactive items of user i are expressed as $V_i^* = (v_1, \ldots, v_t)$, satisfying $v_i \cap v_j = \emptyset$, $V_i^* = \cup_{t=1}^{t} v_t$.

In each interval, for a given item, all its features are concatenated in the preliminary embedding, and then encoded into a l-length binary vector, which is taken as the rough embedding representation of the item. Formally, suppose there are m_i items appearing in the time interval t_i, the item features can be represented with embedding matrix $R_{t_i}^{m\times l}$, in which each row is an l-length binary vector standing for an item. Thus, the rough embedding representation of a user's interactive items can be expressed as $I = (I_1, \ldots, I_t) = \left(R_1^{m_1\times l}, \ldots, R_t^{m_t\times l}\right)$.

C. MODEL OF SDAE

SDAE, the Stacked Denoising AutoEncoder [28], is an improved AutoEncoder [29] (AE). AE is a simple three-layer neural network structure, and is composed of an input layer, a hidden layer, and an output layer. The training purpose of AE model is to make X and Y as similar as possible, where Y is the output, X is the input and the label used to compute errors during training as well. In addition, the output of hidden layer H after training is the latent representation of X. Due to not using additional tagged data in the process of training, the learned latent representation is often

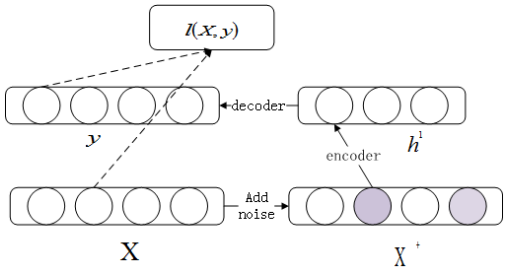


FIGURE 1. Schematic diagram of DAE.

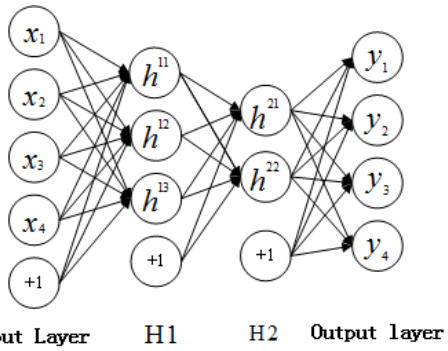


FIGURE 2. Structure of SDAE.

superior to that trained by other annotation model. A common method of improving robustness of the model is to add noise into the input data. As in *Denoising AutoEncoder* (DAE), which is different from AE, noise is added into the input X before training. Obviously, DAE must learn to denoise the data and obtain original input X during training. Briefly, the process of training the model is also a process of learning how to eliminate noise. Fig. 1 depicts the structure of DAE.

Previous research suggests that deep learning is good at extracting the deep features of data. SDAE, a combination of deep learning model and DAE, not only retains the advantages of the DAE model, but also has the ability of extracting deep features of data. Fig. 2 shows a SDAE model with double hidden layers. Compared with DAE model, SDAE model only increases the number of hidden layers, and the output of the hidden layer H_2 is the latent representation of the input data. The learning model of SDAE is formulated as follows:

$$h^1 = f\left(W^1 X^+ + b^1\right) \tag{1}$$

$$h^2 = f\left(W^2 h^1 + b^2\right) \tag{2}$$

$$X^\wedge = f\left(W^3 h^2 + b^3\right) \tag{3}$$

In above formulas, X^+ is the noise-incremented representation of the input data X. h^1, h^2 and $X^\wedge$ denote the output of Hidden Layers H_1, H_2 and Output Layer, respectively. $f(\cdot)$ is the activation function, in which, W and b are weight matrix and bias used in the neural network.

Algorithm 1 Algorithm of Learning Item Presentation With SDAE Model

Input: Rating matrix $R_{m\times n}$, epochs=1000, $\varphi = 0.01$, sttddev=0.3

Output: The list of item latent representation It

1: Training Begin:
2: SGD=SGD($lr = 0.004$)
3: SDAE=SDAE(Optimizer=SGD,Afunction=Sigmoid)
4: $It = []$
5: **for** j in 1:epochs **do**
6: Len=R.Shape()[1]
7: **for** i in 0:Len-1 **do**
8: $X = R[:i]$
9: X^+ =Gaussiannoise(sttddev,X)
10: $X^\wedge$,S_r =SDAE(X^+)
11: $L_{SDAE} = L\left(X, X^\wedge\right) + \varphi\left(\sum_{i=1}^{l}\left\|W^i\right\|_F^2\right)$
12: SGD(L_{SDAE})
13: Save(SDAE)
14: **for** i in 0:len-1 **do**
15: S_r =SDAE($R[:i]$)
16: It.append(S_r)
17: **return** It

The error function for the input and output of the model is defined as :

$$L\left(X, X^\wedge\right) = X\log X^\wedge + (1-X)\left(1-\log X^\wedge\right) \tag{4}$$

To prevent the overfitting of the model, the most common method is to reconstruct the error function by adding regular terms into it. Thus, we rewrite the function as:

$$L_{SDAE} = L\left(X, X^\wedge\right) + \varphi\left(\sum\nolimits_{i=1}^{l}\left\|W^i\right\|_F^2\right) \tag{5}$$

in which, l is the number of layers in the model, and φ is the parameter of regular item.

Taking advantages of its excellent performance on mining the deep features of data, in this work, an SDAE model is used to learn the latent representation with item's rating data. In order to speed up convergence of the model, before computing, we normalize the item's rating data into the interval of [0, 1].

The input of the model includes rating matrix $R_{m\times n}$, activation function (*Sigmoid*), times of time (*epochs*), *Stochastic Gradient Descent* (SGD) algorithm, learning rate (lr), values of regular terms φ, gaussian noise (*sttddev*), etc. The output is the item's latent representation which is generated from the item's rating data. Algorithm 1 gives a detailed description of the model.

As shown in the description of the algorithm, it starts with some initialization, such as setting learning rate lr for the SGD algorithm (Line 2), and assigning SGD, the weights and bias, and the activation function to SDAE model (Line 3). To store the output h^2 of the hidden lary H_2, in Line 4, an empty set is generated. The forward computation of SDAE is described in Lines 6 ∼12, in particular, Line 8 gets the rating for each item, and the noise is added into it in Line 9. The forward propagation computation is accomplished in Line 10 by using Formulas (1∼3). In Lines 11∼ 12, the SGD, optimized by the error functions of Equation (4) and (5), is employed to adjust the parameters for the backward propagation. Clearly, Lines 5 ∼ 13 train once the model on the input data. The latent representation of items is finally generated on the item's rating data, Lines 14∼ 17), in terms of the trained SDAE model.

IV. HARSAM MODEL

In this section, we concentrate on the description of using our proposed HARSAM model, a deep neural network which is improved with self-attention mechanism, to learn the latent preference representation for both users and items. In addition, we discuss in detail the method of generating the list of recommendation.

A. HARSAM MODEL

In real life, user preferences are not fixed, for example, when users focus on some certain items, they will inevitably ignore the others. Due to this, in this work, we propose the HARSAM model, which uses the self-attention to model the internal relationships among items in each period. Fig. 3 describes the structure of the model.

From the figure, we observe that the HARSAM model consists of two parts: the left part and the right part. The left part is used to learn user's latent preference representation S_u, and the right part is response for extracting the representation of item features S_v, with a fully connected neural network.

There are four stages of learning the representation of user's latent preferences, including *embedding the item data*, *extracting item features*, *modeling with self-attention*, and *learning the representation of user's latent preferences*.

In Section III-B, we discussed specially the method of user's interactive items preprocessing and initial embedding of features, whose output is regarded as the rough embedding representation of a user's interactive items and input into the HARSAM model, denoted as I in the figure. This task is implemented in the first stage of our model, that is, *embedding the item data* .

In the stage of *extracting the item features*, the rough embedding representation of a user's interactive items I, is mapped into a d-dimensional space through a fully-connected neural network, which is computed with the Relu function:

$$I^d = f_{Relu}\left(W^d I + b\right) \tag{6}$$

In our work, $f_{Relu}(.)$ is employed as the activation function for every layer of the neural network.

During the stage of *modeling with self-attention*, by use of self-attention mechanism, we respectively extract the intrinsic relationships among items in t intervals. The calculation of attention mechanism in the i-th interval is expressed as

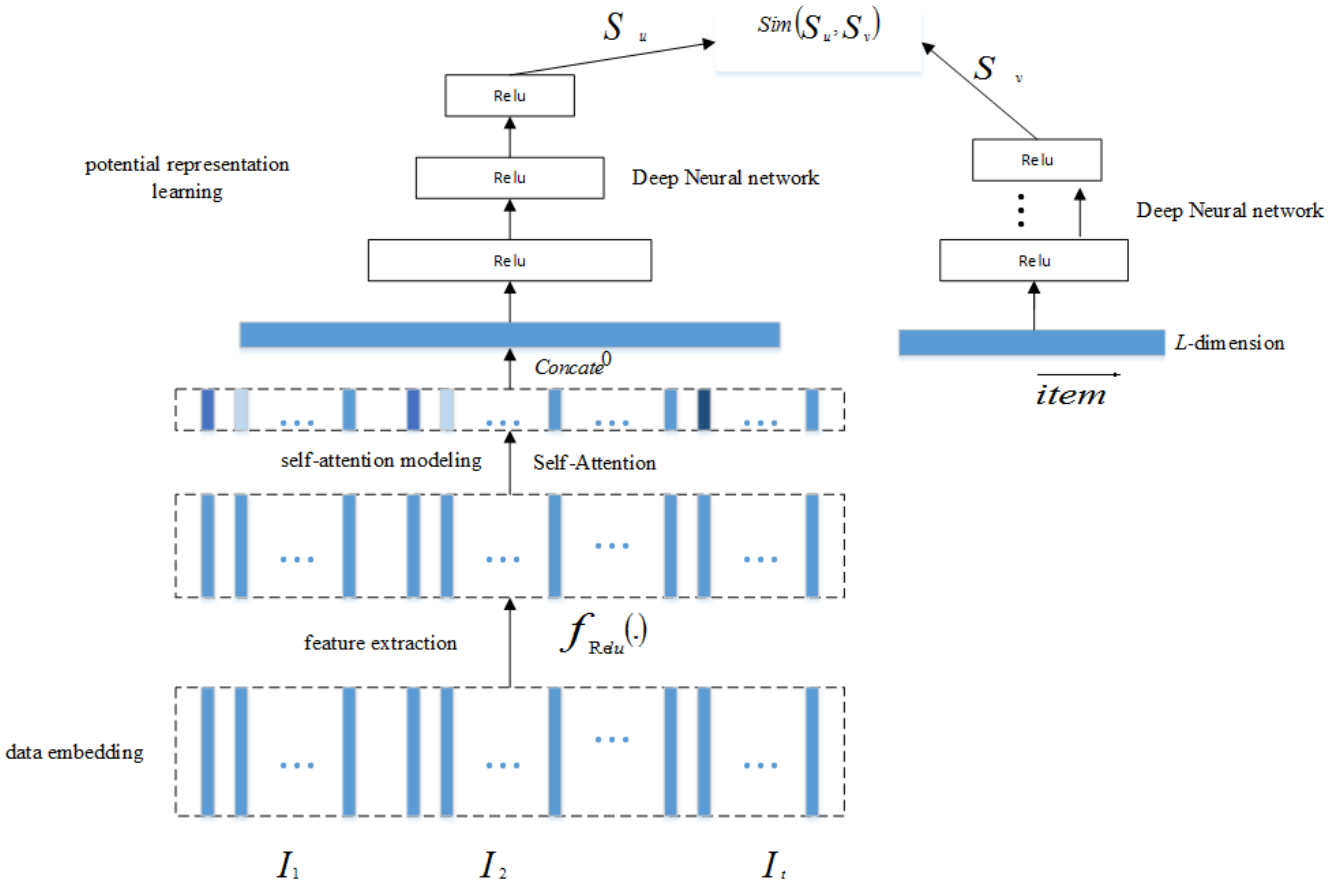


FIGURE 3. Structure of HARSAM model.

follows:

$$I_i^z = f_{Relu}\left(W^z I_i^d + b\right) \tag{7}$$

$$A_z = softmax\left(I_i^z W^A \left(I_i^d\right)^T\right) \tag{8}$$

$$I_i^z = A_z I_i^z \tag{9}$$

in which, A_z is the attention weight matrix of items in the interval. The final result of I_i^z is the z-dimensional feature representation of items in the i-th interval. In following, we symbolize the calculation of attention mechanism as f_{Self} (.). Accordingly, we formulate the items in t intervals as :

$$I^z = \left(f_{Self}\left(I_1^d\right), \ldots, f_{Self}\left(I_t^d\right)\right) \tag{10}$$

The stage of *learning latent representation* is to concatenate the feature representation of each item, and input the result into a three-layer fully connected neural network to compute the latent representation of the user. The process of computation is described as:

$$S_u = f_{Relu}\left(\ldots\left(f_{Relu}\left(Concate^0\left(I^z\right)\right)\right)\right) \tag{11}$$

where, $Concate^0\left(I^z\right)$ denotes the concatenation of the user's interactive item representations. S_u is the latent feature representation of the user.

The right part, which extracts item feature representations by making use of a multi-layer fully connected neural network, can be expressed as:

$$S_v = f_{Relu}(\ldots(f_{Relu}(\overrightarrow{item}))) \tag{12}$$

Here, $\overrightarrow{item}$ is the rough eigenvector of a single item, and S_v is the latent representation of item features.

The HARSAM model, whose input is the set of user's interactive items plus a set of items that the user explicitly prefers, is to maximize the similarity between the user preference feature representation S_u, and the item feature representation S_v. To reach this goal, formally, we can optimize the model with the following loss function:

$$L_{(AS-SADDL)} = -log\frac{(S_u)^T S_v}{\|S_u\| \|S_v\|} \tag{13}$$

B. GENERATING THE RECOMMENDATION LIST

According to the previous discussion, applying the SDAE model on the item's rating data, we can obtain the item latent representation S_r, and with the model of HARSAM, we can get the latent feature representation of user preferences S_u and the latent feature representation of items S_v, from the data of user's interactive items. Hence, the complete latent

representation of an item is composed of the two parts, S_r and S_v.

Let us take top-k recommendation as an example to illustrate the process of recommending items with our model. We do the recommendation in two steps: (1) generate the k-candidate items C_{k_1} according to the pairwise similarity between user preference features and unrated items, denoted as Sim_1, and the k-candidate items C_{k_2} according to the pairwise similarity between user preference items and unrated items, denoted as Sim_2; (2) Combine C_{k_1} and C_{k_2} to compute the final top-k items according to a comprehensive similarity Sim, which is a harmonic function of Sim_1 and Sim_2.

For Sim_1, we calculate it with following formula:

$$Sim_1 = \frac{(S_u)^T S_v}{\|S_u\| \|S_v\|} \tag{14}$$

Then, the unrated items are sorted in descending order of Sim_1, and the top-k items are selected as the candidates C_{k_1}.

To compute the pairwise similarity between the unrated items and the user preference items, Sim_2, the equation is employed :

$$Sim_2 = \left(\frac{(S_r^i)^T S_r^j}{\|S_r^i\| \left\|S_r^j\right\|} \right) + \left(\frac{(S_v^i)^T S_v^j}{\|S_v^i\| \left\|S_v^j\right\|} \right) \tag{15}$$

Similarly, the first k unrated items with the highest similarity are chosen as the candidates C_{k_2}, which is merged with C_{k_1} into the final candidates C_k, denoted as, $C_k = C_{k_1} \bigcup C_{k_2}$.

The next step is to decide the best k items in C_k for the recommendation. We utilize the comprehensive similarity Sim, which is calculated with a harmonic function of Sim_1 and $\overline{Sim_2}$, the average of Sim_2. Suppose that user i has q preferential items, the comprehensive similarity Sim is calculated as follows:

$$\overline{Sim_2} = \frac{1}{q} \sum_{i=1}^{q} Sim_2 \tag{16}$$

$$Sim = \varepsilon \left(\overline{Sim_2}\right) + (1 - \varepsilon)\, Sim_1 \tag{17}$$

where ε is an adjustable parameter, and $\varepsilon \in [0, 1]$. The final items recommended to the user are the top-k items in C_k, which are ranked according to the comprehensive similarity Sim.

On the basis of the aforementioned idea, we summarize the whole process of the top-k recommendation with our model as follows:

1) Input the item set V, user interactive items V_u^*, user preference items V^p, and user unrated items V^u. Among these sets, the expressions $V_u^* \in V$ and $V^p \in V_u^*$ hold.
2) In the use of our HARSAM model, the user latent preference feature representation S_u as well as the latent feature representation of each item S_v is obtained.
3) Employ the SDAE model to extract the latent representation of each item S_r in terms of the rating data of items.
4) Compute the pairwise similarity between the user and the unrated items V^u by exploiting Formula (14), and select the top-k items to construct the candidate set C_{k_1}. Identically, calculate the pairwise similarity between user preference items in V^p with the user unrated items V^u with Formula (15), and select the top-k items to form the candidate set C_{k_2}. The final candidate set C_k is generated by combining the two sets C_{k_1} and C_{k_2}, i.e., $C_k = C_{k_1} \bigcup C_{k_2}$
5) Based on Formulae (16) and (17), the comprehensive similarity Sim is computed, according to which the items in C_k are ranked, and the top-k items in the rank are picked as the final recommended items to the user.

TABLE 2. Statistical information of the datasets.

Datasets	Users	Movies	Rating	Sparser
MovieLens-100k	943	1,682	100,000	6.30%
MovieLens-1M	6,040	3,706	1,000,000	4.46%
Book-Crossing	278,858	271,379	1,149,780	0.0015%

V. EXPERIMENTS

To verify the performance of the model developed in this work, we conducted a series of experiments on the data of MovieLens[1]-100k, MovieLens-1M and Book-Crossing,[2] which are publicly available on the Web. The experiments investigated our model from two aspects. The first one is to find out how the coefficient ε affects the effectiveness of our model, and decide what the exact value of ε is the best for our datasets. The second one is to compare our HARSAM models with some comparative models such as SDAE, item-based model and collaborative filtering models, on the recommendation effects.

A. EXPERIMENTAL SETTINGS

Table 2 summarizes the datasets we employed in the experiments. As their names show, the MovieLens-100k dataset contains 100,000 ratings from 943 users on 1,682 movies, and the MovieLens-1M dataset contains 1 million ratings from 6,040 users on 3,706 movies. The last dataset, Book-Crossing dataset, contains 1,149,780 ratings from 278,858 users on 271,379 books. We formed the user-item matrixes with these ratings, and used them as input to the SDAE model. By statistics, we found the sparsity of the matrix generated from MovieLens datasets are about 6.30% and 4.46%, and that of matrix derived from Book-Crossing is about 0.0015%. The MovieLens datasets show a movie is often labeled as one or more genres (there are 19 genres in total). The category information of each movie is described by a binary vector of length 19, where the bit 1 denotes that the movie belongs to the category, otherwise the corresponding bit is 0. Besides, some additional information about a movies is also provided in the datasets, such as its name, its release time and so on.

[1]https://grouplens.org/datasets/movielens
[2]http://www2.informatik.uni-freiburg.de/ cziegler/BX/

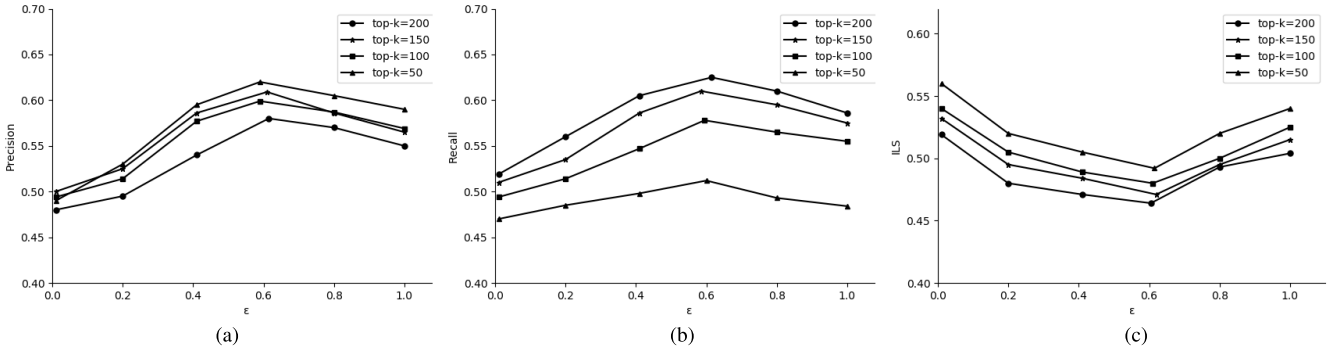


FIGURE 4. Impacts of ε on the model performance. (a) Impact of ε on Precision. (b) Impact of ε on Recall. (c) Impact of ε on ILS.

TABLE 3. Possible cases of a unrated item.

User preferences	Recommended	Not Recommended
like	True-Positive	False-Negative
not like	False-Positive	True-Negative

Some attributes for users and books are also provided in Book-Crossing datasets In the experiments, we set 80% of the data as the training set, and 20% as the test set. The 10-fold crossover average was taken as the final result to evaluate the performance.

We employed the widely applied information evaluation metrics, including precision $P(k)$, recall $R(k)$, ILS, MAP(Mean Average Precision), NDCG(Normalized Cumulative Discounted Gain) as well as the computing time t, as the evaluation indicators.

When making a recommendation, for a given item that has not been rated by the user, there are possibly four cases it may be in: (1) being recommended and the user likes it – True-Positive(N_{tp}), (2) being recommended but the user does not like it – False-Positive (N_{fp}), (3) not being recommended but the user likes it – False-Negative (N_{fn}) and (4) not being recommended and the user does not like it – True-Negative (N_{tn}). Table 3 summarizes these four possible scenarios.

Specifically, in our experiments, we used the average value of each indicator as the actually used metrics, which are evaluated as follows:

$$P(k) = \frac{1}{M}\sum_{i=1}^{M}\frac{N_{tp}}{N_{tp}+N_{fp}} \tag{18}$$

$$R(k) = \frac{1}{M}\sum_{i=1}^{M}\frac{N_{tp}}{N_{tp}+N_{fn}} \tag{19}$$

where M is the number of users in the dataset, and k is the number of items for recommendation.

ILS is generally used to judge the diversity of items within the recommended list, which is evaluated as follows:

$$ILS\,(L) = \frac{\sum_{b_i\in L}\sum_{b_j\in L, b_i\neq b_j} S\left(b_i, b_j\right)}{\sum_{b_i\in L}\sum_{b_j\in L, b_i\neq b_j} 1} \tag{20}$$

where b_i, b_j are the items in the recommendation list L, $S\left(b_i, b_j\right)$ indicates the pairwise similarity of the items.

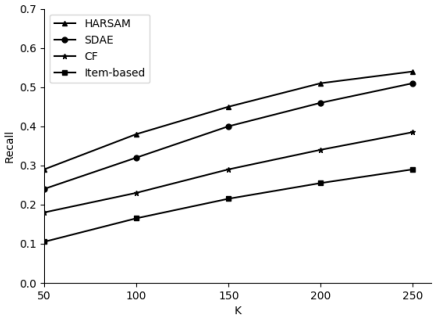


FIGURE 5. Recall on Movielens-100K.

All the experiments were implemented on a PC server with a Tesla M60 GPU, 8-core CPU, as well as 16GB of memory. In terms of appropriate times of tests, we found the values of the hyperparameters were suitable for the following settings: The regularization coefficient φ was set to 0.01, the learning rate of random gradient descent lr was set to 0.004, the noise increase rate $sttddev$ was set to 0.03, and the initial embedding length of item features was set to 1,943.

B. ANALYSIS OF EXPERIMENTAL RESULTS

The comparative methods we selected in the experiments were the *Collaborative Filtering* recommendation algorithm (CF) [21], item content based recommendation algorithm (item-based) [30] and SDAE recommendation algorithm (SDAE) [28]. We comprehensively measured their $R(k)$ and ILS, which are formulated with Equations (19) and (20), and the computing time, MAP, NDCG as well.

1) SELECTING THE VALUE OF ε

According to Formula (17), we found that the values of Sim, the comprehensive similarity between the user and the item, is clearly affected by the value of ε. For example, if ε is set to 0, the algorithm will only consider the user preference feature and the unrated item. Inversely, if ε is set to 1, it means that the user preference items and the unrated items are considered in the model. In Fig. 4, the experimental results show how the precision, recall rate, and ILS vary with the weight ε.

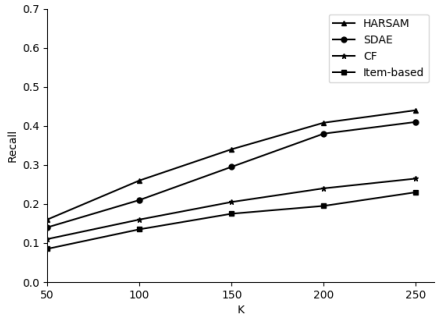


FIGURE 6. Recall on Movielens-1M.

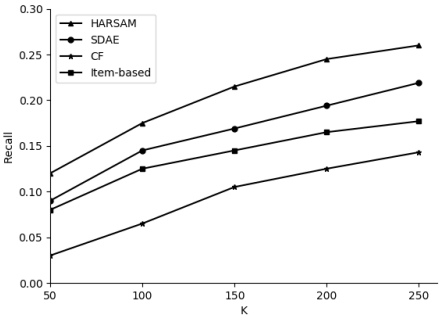


FIGURE 7. Recall on Book-Crossing.

To select the best value of ε for our datasets, we varied the value of ε from 0 to 1.0 with the step of 0.2. Fig. 4 indicates the performance of the algorithm grows as the increase of ε, and ε is around 0.6, all of the three indicators reach the best values. When the value of ε rises further, the performance of the algorithm will decline. Based on this observation, in subsequent experiments, we employ 0.6 as the value of ε.

2) COMPARISON OF RECALL

After selecting the proper values for the hypeparameters, the next experiment we conducted was to compare our model to the comparative approaches with different k. During the experiments, we measured the values of $R(k)$, ILS, MAP and NDCG of our model as well as that of the comparative approaches by setting the values of k from 50 to 250 with the step of 50. The results are shown in Figs. 5∼ 8 and Table 4.

The recall values in Formula 19 reflect the completeness of recommended items. Figs. 5∼7 depict the average recall of our HARSAM model with that of the comparative approaches. As shown in the figures, although as k increases from 50 to 250, the performance of all the four models is improved overtly till it becomes stable, and it is obversable that HARSAM grows much faster at recall than the other models. We viewed the recall rate of the models evaluated with MovieLens-100k is higher than that with the MovieLens-1M and Book-Crossing, that is due to use-item matrix generated from MovieLens-1M and Book-Crossing are sparser than that derived from MovieLens-100k. Overall, experimental results on the three datasets indicate that the proposed HARSAM mode is superior to the comparative models for recommending satisfying items.

3) COMPARISON OF ILS

In recommendation systems, the ILS is often used to indicate the similarity between recommended items. In general, smaller ILS values mean wider range and higher diversity of the recommendation list. The result of the experiments is shown in Fig. 8. They demonstrate that as the value of k increases, all models show a decline at first and then reach a stable ILS. However, the proposed HARSAM model decreases more significantly and always keeps a minimum ILS value. This also implies that it performs better than the comparative models in the diversity of recommendation. This is because the HARSAM model proposed in this work takes the advantage of attention mechanism to model the user's interactive items in different periods. The mechanism considers not only the intrinsic relationship between items, but also the user's preference in different periods of time. Thus, it substantiates that introducing attention attention mechanism into the recommendation model enhances significantly the diversity of recommended results.

4) COMPARISON OF MAP AND NDCG

In recommendation systems, MAP is often used to measure the mean precision of recommendation and NDCG is applied to evaluate the ranking correctness. Table 4 presents the

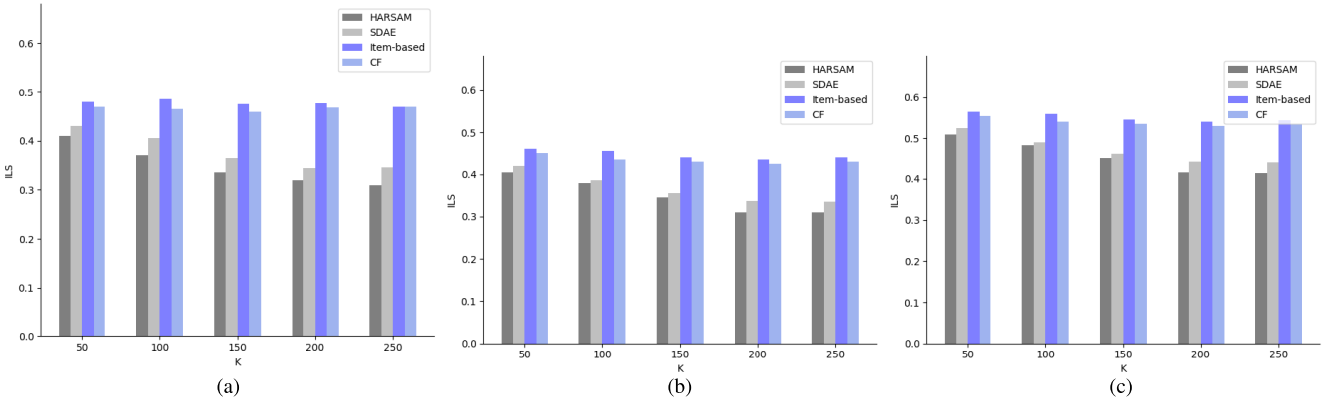


FIGURE 8. Performance comparison of four models based on ILS. (a) ILS on Movielens-100K (b) ILS on Movielens-1M (c) ILS on Book-Crossing.

TABLE 4. MAP and NDCG of different models on the three datasets.

Models	MovieLens-100k		MovieLens-1M		Book-Crossing	
	NDCG@20	MAP	NDCG@20	MAP	NDCG@20	MAP
Item-based	0.5729	0.5261	0.5122	0.5102	0.2310	0.2198
CF	0.6134	0.5382	0.5644	0.5211	0.2110	0.1998
SDAE	0.6543	0.5722	0.6347	0.5534	0.3124	0.2978
HARSAM	0.6869	0.5931	0.6622	0.5748	0.3260	0.3148

MAP and NDCG@20 of the models implemented on the three datasets. It illustrates, both on MAP and NDCG@20, the HARSAM model outperforms the three comparative models. This observation elucidates that the HARSAM model gives the most possible user's preferred items a higher rank in the recommended list, that can largely improve the user experience. This experiment bears out that introducing attention mechanism into the model enables it to obtain more effective user latent representation and also improve the recommendation performance.

VI. CONCLUSION

The recommendation system, which alleviates information overload by improving the efficiency of information retrieving, is widely used to filter information for users. The model proposed in this paper combines the item's rating data and the item's feature information to learn the latent representation of the item. In addition, the HARSAM model, which introduces the self-attention mechanism into deep neural network, was developed to model user interaction data. The model takes into account the interacting relationship between items as extracting the latent representation of the item, that increases the effectiveness of learned latent representation. To some extent, the proposed model improves the computability of item data by easing the cold start issue. Experimental results show the proposed model outperforms the comparative ones at all measurements, such as recall, ILS, NDCG and MAP. In our future work, we will try to connect knowledge graph with our model, which enriches the information of items and user data, to make more improvement to the effectiveness of recommendations.

REFERENCES

[1] M. Gao, K. Liu, and Z. Wu, "Personalisation in Web computing and informatics: Theories, techniques, applications, and future research," *Inf. Syst. Frontiers*, vol. 12, no. 5, pp. 607–629, 2010.

[2] H.-L. Xu, X. Wu, X.-D. Li, and B. P. Yan, "Comparison study of Internet recommendation system," *J. Softw.*, vol. 20, no. 2, pp. 350–362, 2009.

[3] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 403, no. 3, pp. 56–58, 1997.

[4] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proc. Int. Conf. World Wide Web*, 2017, pp. 193–201.

[5] R. Chen, Q. Hua, Y.-S. Chang, B. Wei, L. Zhang, and X. Kong, "A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks," *IEEE Access*, vol. 6, pp. 64301–64320, 2018.

[6] H. Li, X. Diao, J. Cao, and Q. Zheng, "Collaborative filtering recommendation based on all-weighted matrix factorization and fast optimization," *IEEE Access*, vol. 6, pp. 25248–25260, 2018.

[7] R. J. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," in *Proc. 5th ACM Conf. Digit. Libraries*, 2000, pp. 195–204.

[8] B. Maria and P. Z. Ivana, "Tuning machine learning algorithms for content-based movie recommendation," *Intell. Decis. Technol.*, vol. 9, no. 3, pp. 233–242, 2015.

[9] T. De Pessemier, K. Vanhecke, S. Dooms, and L. Martens, "Content-based recommendation algorithms on the hadoop mapreduce framework," in *Proc. WEBIST*, 2011, pp. 237–240.

[10] W. Chen, Z. Niu, X. Zhao, and Y. Li, "A hybrid recommendation algorithm adapted in e-learning environments," *World Wide Web*, vol. 17, no. 2, pp. 271–284, 2014.

[11] W. Haiming, Z. Peng, T. Lu, H. Gu, and N. Gu, "Hybrid recommendation model based on incremental collaborative filtering and content-based algorithms," in *Proc. CSCWD*, Apr. 2017, pp. 337–342.

[12] U. Javaid and J. A. Lee. (2018). "Capturing variabilities from computed tomography images with generative adversarial networks." [Online]. Available: https://arxiv.org/abs/1805.11504

[13] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3128–3137.

[14] M. J. Zhang and C. Stefano, "Evolutionary computation and evolutionary deep learning for image analysis, signal processing and pattern recognition," in *Proc. GECCO*, 2018, pp. 1221–1257.

[15] J. Sulam, V. Papyan, Y. Romano, and M. Elad. (2018). "Multi-layer convolutional sparse modeling: Pursuit and dictionary learning." [Online]. Available: https://arxiv.org/abs/1708.08705

[16] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.

[17] F. Lorenzo and M. Z. Fabio. (2017). "Symbolic, distributed and distributional representations for natural language processing in the era of deep learning: A survey." [Online]. Available: https://arxiv.org/abs/1702.00764

[18] W. You and Y. E. Shui-Sheng, "A survey of collaborative filtering algorithm applied in E-commerce recommender system," *Comput. Technol. Develop.*, vol. 16, no. 9, pp. 70–72, 2006.

[19] J. B. Schafer, J. Konstan, and J. Riedl, "Recommender systems in e-commerce," in *Proc. ACM Conf. Electron. Commerce*, 1999, pp. 158–166.

[20] M. Balabanović and Y. Shoham "Fab: Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[21] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted Boltzmann machines for collaborative filtering," in *Proc. Int. Conf. Mach. Learn.*, 2007, pp. 791–798.

[22] F. Strub and J. Mary, "Collaborative filtering with stacked denoising autoencoders and sparse inputs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[23] Y. A. M. Song Elkahky and X. He, "Neural machine translation by jointly learning to align and translate," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2016, pp. 909–912.

[24] D. Bahdanau, K. Cho, and Y. Bengio. (2014). "Neural machine translation by jointly learning to align and translate." [Online]. Available: https://arxiv.org/abs/1409.0473

[25] Y. Gong and Q. Zhang, "Hashtag recommendation using attention-based convolutional neural network," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 2782–2788.

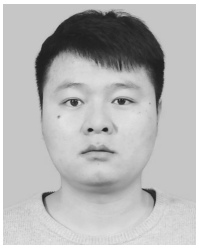
[26] F. Yang, L. Jin, W. Yang, Z. Feng, and S. Zhang, "Handwritten/printed receipt classification using attention-based convolutional neural network," in *Proc. Int. Conf. Frontiers Handwriting Recognit.*, Oct. 2016, pp. 384–389.

[27] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[28] Y. Suzuki and T. Ozaki, "Stacked denoising autoencoder-based deep collaborative filtering using the change of similarity," in *Proc. Int. Conf. Adv. Inf. Netw. Appl. Workshops*, Mar. 2017, pp. 498–502.

[29] M. Angshul and J. Anant, ''Cold-start, warm-start and everything in between: An autoencoder based approach to recommendation,'' in *Proc. IJCNN*, 2017, pp. 3656–3663.
[30] A. van den Oord, S. Dieleman, and B. Schrauwen, ''Deep content-based music recommendation,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 2643–2651.

DUNLU PENG received the Ph.D. degree in 2006. He is currently a Professor in computer science with the University of Shanghai for Science and Technology. He has published more than 100 conference or journal papers in computer science. His research interests include data analytics, natural language process, machine learning, recommendation systems, and information fusion.



WEIWEI YUAN is currently pursuing the master's degree in computer science with the University of Shanghai for Science and Technology. His research interests include recommendation systems, natural language processing, deep learning, and data mining.

CONG LIU received the Ph.D. degree in computer application from East China Normal University, Shanghai, China, in 2013. He is currently a Lecturer with the Department of Computer Science and Engineering, School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, China. His research interests include evolutionary computation, machine learning, and image processing.