Enhanced graph recommendation with heterogeneous auxiliary information

Fulian Yin^{1,2} · Meiqi Ji² · Yanyan Wang² · Zebin Yao² · Xiaoli Feng² · Sitong Li²

Received: 10 August 2021 / Accepted: 1 January 2022 / Published online: 24 January 2022
© The Author(s) 2022

Abstract

The boom in the field of movies and TV programs, which is a kind of information overload, may lead to poor user experience and are detrimental to the healthy development of the industry, hence personalized program recommendation is crucial. Since program names, labels, and synopsis are highly condensed languages, to enable better semantic representations for personalized recommendations and enrich the completeness requirements of data resources, we propose an enhanced graph recommendation with heterogeneous auxiliary information (EGR-HA), focusing on auxiliary information knowledge representations, and graph neural network-based node updates. Firstly, multi-source heterogeneous auxiliary information knowledge is fused to supplement semantics of program and user to obtain initial representations that contain rich semantics, then user and program node embedding representations are aggregated in multiple layers through graph neural networks to model higher-order interaction history information and realize user and program representation update; finally, user viewing prediction is performed based on deep networks to realize personalized program recommendation. The final experiment results in indicators, such as normalized discounted cumulative gain (NDCG), hit rate (HR) and root mean square error (RMSE), verified the effectiveness of this method by comparing with various methods.

Keywords Graph neural network · Program recommendation · Auxiliary information · Program representation · User representation

Introduction

At present, the problem of information overload in the movies and TV programs field is serious, and it is increasingly difficult for users to find interesting programs from a wide range of programs, which makes the user experience worse, and does not take advantage of the healthy development of the movies and TV programs industry, thus personalized program recommendation for film and TV program has come into being [8,44].

Traditional recommendation algorithms mainly contain collaborative filtering (CF) and content-based (CB) methods. CF recommendations predict users' interests by collect-

ing preference information from multiple users [19,39,40], including user-based collaborative filtering [19] and item-based collaborative filtering [39]. However, the scoring matrix is usually very sparse in many applications, leading to significant degradation in the recommendation performance of collaborative filtering-based approaches. CB recommendation is used to recommend items by the similarity between items [2,23,29]. Its similarity is obtained based on the attributes of the item and the user. However, it is not possible to understand the potential interests of users and lacks diversity [29]. At the same time, there are other parts of recommendation methods, such as decision tree [35], clustering [1,55], and Bayesian [36]. The above traditional methods basically perform recommendation research through user and program interaction data, ignoring more deep-seated features.

Then with the rise of deep neural networks, and neural network-based recommendation algorithms have begun to gradually appear, such as Wipe&Deep [7], DeepFM [14], NFM [16] and AFM [52]. These methods build the relationships between users and programs through neural networks

✉ Sitong Li
sitongli@cuc.edu.cn

¹ State Key Laboratory of Media Convergence and Communication, Communication University of China, Beijing 100024, China

² School of Information and Communication Engineering, Communication University of China, Beijing 100024, China

Fig. 1 Users and programs auxiliary information and high-level interaction features

to make recommendations, but they are not very good at solving problems such as data sparsity and cold starts. Therefore afterwards, to enrich the data required for recommendations, auxiliary information is widely used as an important semantic source [11,31,38,46]. Also considering the structural complexity of users and programs, graph neural network [9,20,43,45,49] are now gaining a lot of attention because they can better consider the interaction between programs and users.

Our work is motivated by several observations. First, the auxiliary information is important for program and user representation, which can be divided into program and user side, as shown in Fig. 1. The auxiliary information for the program is abundant, mainly including program names, labels, synopses, broadcast channels, etc. The auxiliary information of user mainly contains user viewing time, user labels and watched programs. The information can better enrich the semantic information for the program and user, which can assist in obtaining more accurate program and user representation, and alleviate the problems of data sparsity and cold start. In addition, graph neural networks can learn the high-level interaction behaviors between users and programs through multi-layer link associations, which are able to capture more accurate user preferences for better program recommendations. As shown in Fig. 1, from the program perspective, program P_1 is watched by user U_1 , and then user U_1 watches program P_2 , hence there are path associations between program P_1 and P_2 . From the user perspective, program P_1 is watched by user U_1 , and program P_1 is also watched by user U_2 , there are also higher-level path relations between U_1 and U_2 , and these relations are transmitted over the link through graph neural network, which can enrich the semantic representation of users and programs, therefore the path association information is important in user personalized program recommendation.

Therefore, it is of great significance to solve the problem of inadequate semantic representation in personalized program recommendation and capture the abundant knowledge-level associations of the user and programs by combining the

auxiliary information of TV programs and graph neural networks. We propose an enhanced graph recommendation with heterogeneous auxiliary information, focusing on auxiliary information aggregation representation, node feature update based on graph neural network and user viewing prediction based on deep network. It can deep mine representation of program and user association through interaction data, and obtain semantic representation of users and programs based on knowledge and content fusion, which can solve the sparseness and cold start problem. Finally, the effectiveness of our model is verified by comparing with existing methods.

The main contributions of this paper are:

- We propose an enhanced graph recommendation with heterogeneous auxiliary information (EGR-HA), which utilize rich auxiliary information to capture the program features and learn the accurate semantic representations of programs and users.
- We combine program and user representations with graph neural networks. The semantic representation of programs and users is used as the initial input features of the graph neural networks to learn the high-level interaction characteristics between users and programs. Through the multi-layer interaction graph structure, the high-level feature association modeling between users and programs and information dissemination between nodes are realized for better personalized recommendation.
- We create a small sample dataset for the TV program recommendation task, which comes from a cable TV platform in capital of China for a certain month. The dataset contains program attributes, user's viewing records and other information, which provides convenience for fully tapping users' interests, modeling users' preferences and recommending programs.
- We extensively conduct experiments and performance evaluation. The experiment results show that our proposed EGR-HA method outperforms state-of-the-art methods on recommendation fields.

The remainder of this paper is organized as follows. We review prior related literature in Sect. “Related works”. Section “Our EGR-HA model” gives a detailed description of enhanced graph recommendation model with heterogeneous auxiliary information. Detailed experiment results and analysis are shown in Sect. “Experiments and result analysis”. Finally, Sect. “Conclusions” states our conclusions and ideas for future work.

Related works

Recommendation algorithms originated from the information overload problem which was proposed in 1964 [4,34], and then a series of traditional recommendation algorithms were proposed successively, including collaborative filtering (CF) algorithms [19,39,40], content-based (CB) recommendations [2,23,29], and matrix factorization-singular value decomposition (MF-SVD) [5]. These methods have improved the performance of recommendation algorithms to different degrees. However, there is a bottleneck in the accuracy rate. Then deep learning techniques took off in the field of recommendation systems from 2016 [10,11]. Meanwhile, neural networks can fully unveil the deep relationship underlying the auxiliary information, programs and users, avoiding the lack of semantics brought by only relying on user-program interactive data. In addition, neural networks can provide better basis and robustness for accurate program and user representations, which have gradually become a key technology for the TV program recommendation research. In the following, we mainly introduce the research on the application of neural network technology in the field of recommendations.

Development of neural network recommendation algorithm

Recommendation algorithms are used to extract user and item characteristics from user history data to make recommendations for users. In 2016, Google applied deep neural networks to video recommendation on YouTube [10], dividing the recommendation process into two stages: recall and ranking. In the same year, Cheng et al. [7] proposed the Wide&Deep model, which combined the memory capability of the wide linear model and the generalization capability of the deep neural network. Wang et al. [47] proposed the Deep&Cross model, which used a cross network instead of wide to increase the interaction between features through the cross layer. Guo et al. [14] proposed DeepFM to replace the wide section of Wide&Deep with FM to strengthen the ability of shallow network combination features. In the same period, some improved models for FM were also proposed, such as NFM [16], AFM [52], etc. AFM [52] was not only the contin-

uation and evolution of FM, but also introduced the attention mechanism into the recommendation system. Zhang et al. [42] considered the dynamic preferences of users, proposing a deep sequential model for live streaming recommendation. Zhou et al. [57] combined DNN with attention mechanism and proposed DIN, taking the influence of different user behaviors into account. The team then proposed an improved DIEN model [56], which used a sequence model to simulate the evolution of user interests. Zhu et al. [58] proposed a deep attentional neural network DAN for news recommendation, which considered the sequential information of users’ clicks, but it ignored the different importance of different words in each name or profile.

With the development of technology, graph neural network has gradually become popular research in various fields [53]. In recommendation systems, most information has a graph structure so that related research based on graph neural network recommendation has attracted more and more attention from scholars. Berg et al. [3] proposed a graph auto-encoder framework to solve the problem of rating prediction in recommendation systems from the perspective of link prediction. Subsequently, Ying et al. [54] proposed PinSage, a random walk GCN capable of learning node embeddings, which learned aggregation functions instead of fixed nodes and could be more adaptable to the practical situation of constantly changing graph nodes. Guo et al. [15] proposed a new hybrid normalized deep graph convolutional network, which solved the problem that the CF model based on GCN could not model high-order cooperative signals. Wang et al. [49] proposed neural graph collaborative filtering (NGCF), which encoded user-item interaction information into the embeddings. Wu et al. [51] proposed a session-based graph neural network model which applied session data as graph-structured data to capture the complex connections between items. Recently, Chang et al. [6] re-constructed loose item sequences into tight item-item interest graphs based on metric learning to integrate different types of user preferences. Wu et al. [50] proposed the User-as-Graph which models the user as a heterogeneous graph composed of behaviors to represent users more accurately.

For film and television program recommendation, there are many studies on different application scenarios. Covington et al. [10] used a deep neural network to recommend YouTube videos, focusing on candidate generation and ranking. Kim et al. [25] used channels and genres of TV programs to construct multiple context preference matrices as recommendation features. Zhao and Pan [13] designed an offline recommendation module based on the information retrieval generative adversarial network (IRGAN) algorithm model, and an online recommendation module based on online user behavior data collection and processing, and implemented a movie recommendation system based on the IRGAN model. Liu [28] used a targeted preview approach for live cable

TV programs based on labels, and it could not consider the impact of different tags and the relationship between users and tags. Seo et al. [41] proposed a new approach that integrates explicit information from IPTV and OTT services and used probability matrix decomposition to achieve program recommendations. Wang et al. [48] used CNN to extract local relevant features of movie titles, and then fused each feature to calculate predicted ratings and recommend movies.

Development of neural network recommendation based on auxiliary information

Since the information that users interact with items is very scarce or even absent, recommendation systems often face the problems of data sparsity and cold start. To solve these two problems, recommendation systems in various fields have used different types of auxiliary information.

The most common auxiliary information is based on user or item attribute features, such as user's gender, age, hobbies, and item's category and content, etc. Gantner et al. [11] introduced many different attributes such as user's gender, age, geographic location, occupation, or item's genre, category, and keywords as auxiliary information. Hwang et al. [22] used the ratings of category experts as auxiliary information. Ji et al. [24] optimized Bayesian personalized ranking by introducing personal tags. Subsequently, Ma et al. [31] proposed the use of cross-topic and cross-media information, etc., and they believed that unrelative information to the topic on other platforms could be conducive to recommendation. Recently, Ni et al. [33] introduced a novel two-stage embedding model (TSEM), which adequately leverage item multimodal auxiliary information to improve recommendation performance. Hui et al. [21] regarded knowledge graphs as heterogeneous networks to add auxiliary information, and proposed a recommendation system with unified embeddings of behavior and knowledge features.

With the development of social networks, social data has become an increasingly important source of information. Massa et al. [32] first proposed to integrate trust relationships in social into recommendation algorithms, using trust between users instead of traditional similarity to predict the vacancy value of the user. This method has greatly improved accuracy compared to traditional collaborative filtering recommendation algorithms. Ma et al. [30] mapped the high-dimensional user rating matrix to the low-dimensional

feature matrix, and fused the social information of users and their respective implicit data sources to make recommendations. The experimental results prove that this method can improve the accuracy of recommendation, but it could cause some information lost. Wang et al. [46] combined users' social trust and rating similarity, and proposed a new matrix-filled recommendation method, which significantly improved the accuracy of predictive ratings and improved the sparsity problem in the recommendation process. In 2020, Sankar et al. [38] used social networks as an auxiliary data source to model user behavior in social platforms. Recently, Liu et al. [37] expressed the friend recommendation problem as a multi-faceted friend ranking on the friendship graph, and exploited heterogeneous information in the platform to overcome the structural and interactive sparsity.

Different from the existing methods, our approach exploits the heterogeneous auxiliary information of programs and users, combining with graph neural network to better learn more informative program and user representations for personalized TV programs recommendation.

Our EGR-HA model

To solve the problems of data sparsity and cold start causing by semantics missing and ambiguity in personalized program recommendation in the field of movies and TV programs, and to capture the rich knowledge-level association between users and programs, in this paper we propose an enhanced graph representation recommendation with heterogeneous auxiliary information (EGR-HA), which integrates multi-source auxiliary information. It can complement and extend the program and user semantics through the knowledge embedded in the auxiliary information to solve the problem of data sparsity, with updating and aggregating the program and user semantic representation based on iterations of graph neural network at a high level. Finally, the effect is verified through the users' viewing interest prediction task. The model consists of three parts, which are semantic knowledge aggregation representation based on auxiliary information, node feature update based on graph neural network, and user viewing prediction based on deep network. The model architecture is shown in Fig. 2.

To represent our model better, the pseudo code of the model is listed:

Algorithm: The pseudo code of EGR-HA algorithm

Input: Each user's viewing data and program data

Output: The probability that each candidate program may be watched by users

1. **For** each program p **in** program list:
 2. **For** each lexical-level feature **in** name, labels and synopsis **of** program p :
 3. Conduct word embedding after segmenting;
 4. Obtain the contextual word representations;
 5. Obtain the corresponding lexical-level feature representation (program name, labels and synopsis) by adding and averaging;
 6. **For** each identity-level feature **in** channel, directors and actors **of** program p :
 7. Conduct word embedding by random;
 8. Obtain the corresponding identity-level feature representation (program channel, directors and actors);
 9. Obtain initial program representation by aggregating the semantic representations of auxiliary information, which are obtained by step 2 to 8;
 10. **For** each user u **in** user list:
 11. **For** each program **in** programs watched by user u :
 12. Calculate viewing interest matrix W by watching duration;
 13. Obtain program representation by step 9;
 14. Obtain initial user representation by integrating W and programs representations watched by the user u ;
 15. **For** each user-program pair (u, p) **in** user-program list:
 16. Obtain information embedding representation of (u, p) by step 9 and 14;
 17. Obtain nodes (program and user) representation in layers by aggregating neighbors and own previous layer using graph neural network;
 18. **For** program **in** program list to be predicted:
 19. Concatenate user and program representations by step 17;
 20. Calculate the probability that the program may be watched through deep neural network.
-

Unlike the basic neural network method that only considers the influence of programs on user representation, our model incorporates more multi-source heterogeneous auxiliary information feature data, such as program names, labels, synopses, directors, actors, and broadcast channels to explore the influence of different auxiliary information for recommendation. First, in the stage of multi-source heterogeneous knowledge aggregation, for each program, the initial semantic representation of the program is obtained by pre-trained embedding representations with auxiliary information. And for each user, the initial user interest representation is obtained by embedding representations with factors such as programs watched by the user, user labels, and viewing duration. Second, in the node information updating of the graph neural network, the initial semantic representations of programs and users are input to build a user-program relationship graph, then the semantic transfer and aggregation are performed in the graph structure through the connection relationship among neighbor nodes to achieve the update of the program and user representations. Third, user viewing

interest prediction based on deep network is conducted to achieve program recommendation.

Program and user initial representation with auxiliary information

Recommendation algorithms in the TV programs and movies basically use very simple forms of data, such as users' clicking behaviors on programs, ignoring the influence of many additional information, like some attribute features of programs and users. To enhance the richness of semantic capture, we utilize multi-source heterogeneous auxiliary information to extend semantic features for the acquisition of initial semantic representations of programs and users, including names, labels, synopses, directors, actors, and broadcast channels of each program; viewing time and watched programs of each user. According to the different features of program auxiliary information, they are classified into 2 categories: the first category is lexical-level features, which specifically include program names, labels, and synopses; the second category is identity-level features, which specifically include program channels, directors, and actors.

Convergence of auxiliary information representations

(1) Lexical-level features

The words contained in program names, program labels and program synopses are rich in semantic information, and there may be information overlap between them. For example, the word "happiness" appears in the synopsis of program A, and the word "contentment" exists in the name of program B. Thus, even though the names and the synopsis of programs A and B are different, there is still semantic similarity between them. Therefore, we first build the vocabulary table V with the words contained in the program names, labels and synopses, and then semantically characterize the words in the vocabulary table to obtain the word embedding matrix $\mathbf{M}_o$. For the i -th word w_i of the vocabulary table V , its corresponding pre-trained word embedding $\mathbf{e}_i$ is defined as:

$$\mathbf{e}_i = \mathbf{M}_o(w_i), \quad (1)$$

All auxiliary information vocabularies are then queried by $\mathbf{M}_o$ to assign embedding representations to the different attributes of the programs. The program name is denoted as pn , and after splitting it into words, there are s_1 words in total. The words are queried in $\mathbf{M}_o$, and finally the program name representation $\mathbf{e}_{pn}$ is defined

Fig. 2 EGR-HA model architecture diagram

as follows:

$$\mathbf{e}_{pn} = \frac{1}{s_1} \sum_{i=1}^{s_1} \mathbf{M}_o(w_i), \quad (2)$$

The program label is denoted as pl , and the label set is $\{w_1, w_2, \dots, w_i, \dots, w_{s_2}\}$, which contains s_2 words in total. The words are queried in $\mathbf{M}_o$, and finally the program label representation $\mathbf{e}_{pl}$ is defined as follows:

$$\mathbf{e}_{pl} = \frac{1}{s_2} \sum_{i=1}^{s_2} \mathbf{M}_o(w_i), \quad (3)$$

The program synopsis is noted as ps , and after splitting it into words, there are s_3 words in total. The words are queried in $\mathbf{M}_o$, and finally the program synopsis representation $\mathbf{e}_{ps}$ is defined as follows:

$$\mathbf{e}_{ps} = \frac{1}{s_3} \sum_{i=1}^{s_3} \mathbf{M}_o(w_i), \quad (4)$$

(2) Identity-level feature extraction

For factors such as channels, directors and actors, they are more used as identifiers to distinguish or obtain the similarity of different programs. For example, if program A and program B are broadcasted on the same channel or have the same actor or director, it can be considered that

in comparison, programs A and B have a higher similarity. Therefore, for program channels, all programs broadcast on s_4 platforms are expressed as a sequence $\{c_1, c_2, \dots, c_i, \dots, c_{s_4}\}$ and then we initialize these channels randomly, so that for the program p broadcast on the i -th channel c_i , the channel embedding $\mathbf{e}_{pc}$ is denoted as:

$$\mathbf{e}_{pc} = \text{random}(c_i, D), \quad (5)$$

where $\text{random}(\cdot)$ denotes a random initialization of the word c_i , and the dimension of the initialization vector is D .

Similar process is done for directors and actors here, because both are personal names, we aggregate the two together for processing. The directors and actors contained in all programs have a total of s_5 , which are represented as a sequence $\{a_1, a_2, \dots, a_i, \dots, a_{s_5}\}$. These program names are then initialized randomly, so that the embedding $\mathbf{e}_a^i$ of the i -th director or actor a_i of the program p is expressed as:

$$\mathbf{e}_a^i = \text{random}(a_i, D), \quad (6)$$

Programs p contains s_6 directors and actors, so the overall embedding $\mathbf{e}_{pa}$ of the directors and actors of the program p is represented as:

$$\mathbf{e}_{pa} = \frac{1}{s_6} \sum_{i=1}^{s_6} \mathbf{e}_a^i, \quad (7)$$

The above is a knowledge convergence representation of all kinds of auxiliary information at the program layer, including program labels, program names, program synopses, program channels, and the directors and actors of the programs.

Initial embedding semantic representation for programs and users

(1) Program representation

Firstly, the representation of multi-source heterogeneous program auxiliary information is realized with the auxiliary information embedding representation, and then the initial semantic embedding vector of the program is jointly built with the aggregation of semantic representations of auxiliary information.

$$\mathbf{e}_p^s = w_1 \mathbf{e}_{pn} + w_2 \mathbf{e}_{pl} + w_3 \mathbf{e}_{ps} + w_4 \mathbf{e}_{pc} + w_5 \mathbf{e}_{pa}, \quad (8)$$

where $\{w_1, w_2, w_3, w_4, w_5\}$ are the corresponding combination weights of program name, program label, program synopsis, program channel, and director and actor of the program, which are set to 0 or 1.

(2) User representation

We obtain user semantic embeddings from the programs watched by users, which mainly consider the interest level of different users for different programs. For example, although users u watched programs A and B, but program A was broadcasted for 1 hour and watched for 30 min, while program B was broadcasted for 30 minutes and watched for 30 min, both of them obviously have different representations for users' interests, so it is necessary to measure the weights of users' interest in watching programs. Here, we define the viewing interest matrix of user u for the program p , defined as follows:

$$W_{p,u} = \sum_{i=1}^N \frac{V_i}{B}, \quad (9)$$

where $W_{p,u}$ denotes the user's interest level for the program p , and V_i denotes the i -th time the user watched the program p , B denotes the duration of the program p , and N denotes the number of times the user watched the program p over a period of time. Then, by integrating the viewing completeness of the programs watched by the user u , we obtain the user representation:

$$\mathbf{e}_u^s = \sum_{p \in P_u} \mathbf{e}_p^s W_{p,u}, \quad (10)$$

where P_u denotes the programs the user u watched.

In summary, this section aggregates the semantic information contained in these auxiliary information contents to characterize them, and finally aggregates them upwards to obtain the initial semantic representations of programs and users for subsequent graph neural network inputs.

Graph neural network information transfer

Information dissemination embedding layer

The method in the previous subsection obtains the initial semantic representation of the program and user, $\mathbf{e}_p^s$ and $\mathbf{e}_u^s$, which are used as the initial input features of the graph neural network model. Traditional methods may directly obtain these embedding results through the interaction layer and then predict the user's probability of viewing programs [18]. In contrast, in graph neural networks, the influence of neighboring nodes and different numbers of networks can be taken into account, and the information is propagated through the interaction graph of users and programs to achieve aggregated updates of the semantic representations of them, and to obtain embedding representations of user and program nodes containing richer semantic information for personalized program recommendations. The specific implementation process mainly includes the construction and aggregation of information.

(1) Construction of information

For a user-program pair (u, p) , we first define the information dissemination process from the program to the user as follows:

$$\mathbf{I}_{p \rightarrow u} = f(\mathbf{e}_p, \mathbf{e}_u, W_{p,u}), \quad (11)$$

where $\mathbf{I}_{p \rightarrow u}$ is the information embedding representation that needs to be integrated, the $\mathbf{e}_p$ and $\mathbf{e}_u$ are respectively the embedding representations of users and programs. The study focuses on the user representation that needs to be integrated through the input $\mathbf{e}_u$ and $\mathbf{e}_p$ and the user's interest in the program $W_{p,u}$ to design the information dissemination function $f(\cdot)$. The specific information dissemination function is defined as follows:

$$\mathbf{I}_{p \rightarrow u} = \frac{1}{\sqrt{|N_u|} \sqrt{|N_i|}} (W_1 \mathbf{e}_p + W_2 (\mathbf{e}_p \odot \mathbf{e}_u)), \quad (12)$$

where W_1 and W_2 denote the trainable parameters, N_u and N_i denote the number of neighbors directly connected to the user u and program p , and " $\odot$ " denotes the point-by-point multiplication operation. Based on the traditional consideration of only single synthesis of

user representation by program representation, we incorporate the interaction behavior of user and program, which refer to reference [48] to better perform user representation and program representation to improve the effectiveness of recommendation.

(2) Program information integration

In the graph neural network, we represent both users and programs as nodes, and the embedded representation of nodes is used as features of nodes, and then the representation of a new layer is obtained by aggregating neighbors and their own previous layer, and here no distinction is made between users and programs as different nodes. At this time, the current node set as n is connected to the neighboring nodes, hence the embedding representation $\mathbf{e}_p^{(1)}$ of the first layer of programs:

$$\mathbf{e}_p^{(1)} = A \left(\mathbf{e}_p^s + \sum_{u \in N_p} \mathbf{I}_{u \rightarrow p} \right), \quad (13)$$

where $A(\cdot)$ represents the activation function, which this paper uses ReLU, $\mathbf{e}_p^s$ is the initial program embedding obtained from Sect. “Program and user initial representation with auxiliary information”, $\mathbf{I}_{u \rightarrow p}$ indicates the information transmitted by user u to program p .

The multi-layer embedding uses the similar principle, and the high-level embedding is defined as follows:

$$\mathbf{e}_p^{(l)} = A \left(\mathbf{e}_p^{(l-1)} + \sum_{u \in N_p} \mathbf{I}_{u \rightarrow p}^{(l)} \right), \quad (14)$$

where $\mathbf{I}_{u \rightarrow p}^{(l)}$ denotes the information transfer of the neighboring users surrounding the l -th layer of the program p , and $\mathbf{e}_p^{(l-1)}$ denotes the representation vector of the $l-1$ -th layer of the program p .

(3) User information integration

In each layer of a graph neural network, the embedding representations of user u are obtained by aggregating the user representations of the previous layer and the embedding representations of neighboring programs connected to the user. In the first layer, the initial embedding representation of the user features is:

$$\mathbf{e}_u^{(1)} = A \left(\mathbf{e}_u^s + \sum_{p \in N_u} \mathbf{I}_{p \rightarrow u} \right), \quad (15)$$

The subsequent high-level embedding is defined as follows:

$$\mathbf{e}_u^{(l)} = A \left(\mathbf{e}_u^{(l-1)} + \sum_{p \in N_u} \mathbf{I}_{p \rightarrow u}^{(l)} \right), \quad (16)$$

where $\mathbf{I}_{p \rightarrow u}^{(l)}$ denotes the information transfer of surrounding neighboring programs to the l -th layer user u , and $\mathbf{e}_u^{(l-1)}$ denotes the $l-1$ -th layer of the representation vector of the user u , l denotes the l -th layer of the graph neural network.

User and program node update strategy

Through graph neural networks, in this paper we obtain a multi-layer representation of each node, each layer of which represents the different importance of the node in disseminating information in different layers. The model integrates them to obtain the final node representation, including both program and user representations, defined as follows:

$$\mathbf{e}_u^f = \text{cat} [\mathbf{e}_u^s, \dots, \mathbf{e}_u^{(l)}, \dots, \mathbf{e}_u^{(L)}], \quad (17)$$

$$\mathbf{e}_p^f = \text{cat} [\mathbf{e}_p^s, \dots, \mathbf{e}_p^{(l)}, \dots, \mathbf{e}_p^{(L)}], \quad (18)$$

where $\mathbf{e}_u^f$ denotes the final semantic representation of the user, and $\mathbf{e}_p^f$ denotes the final representation of the program, and $\text{cat}[\cdot]$ denotes the vector concatenation processing operation.

In summary, the high-level interaction characteristics between users and programs can be captured by this information transfer integration and update method, which can enrich the semantic representations of users and programs, hence the path association information is important in user personalized program recommendation.

Deep network-based user viewing prediction

The similarity between users and candidate programs is calculated by program and user representation. In this paper, a deep neural network is used to train the user's interest level in watching a program. The user and program representations are first spliced to obtain a vector of user interest features for the program. Then the results are transferred to predict interest level $\hat{y}_{(u,p)}$ of the user u to the program p , which is defined as shown in the following equation.

$$\hat{y}_{(u,p)} = D(\text{cat}[\mathbf{e}_p^f, \mathbf{e}_u^f]), \quad (19)$$

where $D(\cdot)$ denotes the deep neural network architecture, and $\text{cat}[\cdot]$ denotes the splicing processing operation.

Eventually, the model is trained to optimize the program and user representation to obtain more accurate results.

$$\text{Loss} = - \sum_{i \in Z} (\hat{y}_{(u,p)} - y_{(u,p)})^2 + \lambda \|\Theta\|_2^2, \quad (20)$$

where Z denotes the training data set, and $y_{(u,p)}$ denotes the interest level of the real user u to the program p , and $\lambda \|\Theta\|_2^2$ denotes the regularization item.

Experiments and result analysis

Experimental setup

The data in this paper is the viewing data of users in Beijing, the capital of China, for a month in a year, totaling 17 million records, including 37,459 households and 2422 programs. Each viewing record includes the user ID, the programs the user watched, the channel the program was on, the program duration, the program label, and the user viewing time. To further extend the program and user attributes, in this paper, we collect auxiliary information data, including the label type and content data of programs on the Internet to extend the program labels, as well as collecting program attribute data from Baidu Baike, Wikipedia, etc., and supplement the directors, actors and synopses information of some programs. The data of 1000 of these users are randomly selected for analysis in the experiment, and then randomly split into a training set, a validation set, and a test set according to the ratio of 6:2:2.

In the experiment, the word embedding is 80-dimensional, the graph neural network has 2 layers, the final deep neural network has 3 layers, Adam [26] is used for model optimization, the learning rate is 0.01, the batch size is 2048, all metrics are repeated 10 times for each experiment to take the average value, and the training epochs are 90 in the following experiments.

Evaluation indicators

The model proposed in this paper calculates the probability of users watching the program and the level of interest in the program. To evaluate the effectiveness of the model, the following three evaluation indicators are used: Hit Rate (HR), Normalized Discounted Cumulative Gain (NDCG) and Root Mean Square Error (RMSE), where NDCG is even subdivided into two indicators NDCG@5 and NDCG@10. The larger NDCG and HR means the better result, while the smaller RMSE means the better result.

HR is a recall-based metric that can be expressed as:

$$HR = \frac{N_k}{GT}, \quad (21)$$

where GT denotes the number of test sets, and N_k denotes the total number of test sets in each user's first K recommended results for each user.

The NDCG metric is sensitive to the position ranking of the recommendation results, and it will give a higher score

to the top ranked programs of the recommendation results. The formulas of NDCG can be expressed as follows.

$$rel_i = \max(act_i) - act_i + 1, \quad (22)$$

$$DCG@N = \sum_{i=1}^N \frac{2^{rel_i} - 1}{\log_2(1 + pre_i)}, \quad (23)$$

$$IDCG@N = \max(DCG@N), \quad (24)$$

$$NDCG@N = \frac{DCG@N}{IDCG@N}, \quad (25)$$

where act_i denotes the actual ranking value, and rel_i denotes the degree of correlation, and pre_i denotes the predicted ranking value, and N denotes the length of the recommendation list.

RMSE is mainly to calculate the deviation of two columns of numbers (observed and true values). Supposing that the true value sequence is y_T and the predicted sequence of values is y_P , the RMSE of the two columns is calculated as:

$$RMSE(y_T, y_P) = \sqrt{\frac{1}{I} \sum_{i=1}^I (y_{T_i} - y_{P_i})^2}, \quad (26)$$

where I denotes the total number of each column, the y_{T_i} and y_{P_i} denote the true and predicted value of the i -th in sequence.

Comparative experiments

To explore the effectiveness of the model proposed in this paper, the performance of the model is compared with the following models widely used in recommendation systems to evaluate the performance of our model. The specific methods include:

- (1) User-based CF algorithm, UserCF [40], it is to search the nearest neighbors of the target user based on the similarity between the user's rating vectors of items, and then generate a recommendation list for the target user based on the ratings of the nearest neighbors.
- (2) Item-based CF algorithm, ItemCF [19], it is to first calculate the similarity of all programs, and then find the set of programs that are most similar to the programs in the user's viewing history for the target user.
- (3) Singular value decomposition, SVD [5], a typical matrix decomposition method, recommends through the interaction matrix between users and programs.
- (4) Neural collaborative filtering, NCF [18], a generic framework for collaborative filtering based on neural networks that expresses and generalizes matrix decomposition and models the latent features of users and programs.
- (5) Graph convolutional networks, GCN [27], a variant of traditional convolution algorithm on graph structure

Table 1 Comparative effects of different methods

Serial number	Method	NDCG@5	NDCG@10	HR	RMSE
1	UserCF [40]	0.7305	0.8054	0.5784	1.31
2	ItemCF [19]	0.7403	0.8134	0.5802	0.6015
3	SVD [5]	0.7422	0.8133	0.6050	1.9529
4	NCF [18]	0.8182	0.8698	<u>0.6618</u>	0.5817
5	GCN [27]	0.8151	0.8689	0.6564	0.5587
6	LightGCN [17]	0.8173	0.8698	0.6613	0.6339
7	BiNE-NGCF [12]	0.8120	0.8662	0.6551	<u>0.5467</u>
8	NGCF [49]	<u>0.8186</u>	<u>0.8706</u>	0.6589	0.5528
9	EGR-HA	0.8454	0.8903	0.6722	0.5226
10	Boost %	3.27%	2.26%	1.57%	4.41%

The bold numbers indicate the best performance of each metric, and the underscore numbers indicate the secondbest result of each metric

data, whose key component is neighborhood aggregation mechanism to extract user and item representation.

- (6) LightGCN [17], simplifies the original GCN to include only the most basic neighborhood aggregation components.
- (7) BiNE-NGCF [12], BiNE is a two-part network representation learning method for initial embedding in NGCF.
- (8) NGCF [49], a graph-based collaborative filtering recommendation method which aggregated the interaction behaviors of users and programs through graph results.
- (9) Our EGR-HA proposed in this paper, it obtains richer semantic representations of programs and users through auxiliary information aggregation knowledge, and then uses them as input embeddings of graph neural network models for node updating, which is of great significance for mining knowledge-program-user relationships.

As shown in Table 1, “Boost %” represents the improvement of our model compared to the best result in the comparison models, which is calculated by subtracting the best result in the comparison models from our result and dividing it by the result of the best comparison model. Specifically, NGCF [49], BiNE-NGCF [12] and NCF [18] have the best result in the comparison models, which are underlined.

As observed in Table 1, first, the traditional recommendation methods such as UserCF, ItemCF, and matrix decomposition method SVD are significantly less effective than the neural network NCF, and the graph neural network methods such as GCN, LightGCN, BiNE-NGCF, NGCF and EGR-HA. This is because the neural network based methods can capture more accurate representations of user and program through the rich auxiliary information, and alleviate the data sparsity and cold start problem, thus improve the recommendation performance. In addition, our EGR-HA method can better learn program and user representations by expanding knowledge through multi-source heterogeneous auxiliary information, making the results significantly bet-

Fig. 3 Performance of NDCG@5 index under single factor

ter than those using only graph neural networks, for better personalized program recommendations.

Auxiliary experiments

For personalized program recommendation, we utilize multiple sources of heterogeneous auxiliary information to explore performance of model from a fine-grained perspective. We explore the effect of different combinations of auxiliary information on the results of the recommendation model. The auxiliary information includes program name, program label, program synopsis, program channel, program director and actor. These are a few combinations of these factors, which are analyzed separately below. Firstly, for the case of single-factor combination, the results are shown in Fig. 3.

As can be seen from Fig. 3, the factor of channel has the best effect, followed by the program label, then the program name and director-actor factors have similar effects, while the program synopsis has the worst effect. The reason for this may be that the program channel, program label, and program name exist for each program in the original rating data, but the director, actor and synopsis information is captured on the Internet and is not complete, especially the synopsis

Fig. 4 Performance of NDCG@5 index under two factors based on channel

information, which shows that the more complete the information is, the better the effect is. The channel, program label and program name have the best effect, probably because the number of channels is small, and the programs broadcasted on the same channel have commonality and are more likely to be seen by the same user, which also reflects users watch programs with strong channel regularity and further reflects the particularity of TV users' interest in programs. This is the same reason as the director and actor information which has less data but also performs better.

From the above analyses, the channel has the best effect in the case of single factor. To further explore the performance of multiple factors, we combine the channel with other four factors, and the experimental results are shown in Fig. 4.

It can be seen from Fig. 4 that the overall effect of the two-factor combination is better than using only a single factor, and “Channel + Program Name” has the best effect, followed by “Channel + Director and Actor” and “Channel + Program Synopsis”, the worst is the effect of “Channel + Program Label”. The possible reason for the above results may be that although the synopsis factor is not effective when using single factor, the combination of channel and synopsis information makes the two complementary, leading to better results, which is also the reason for the better

performance of “Channel + Director and Actor”. The combination of “Channel + Program Name” has the best effect, which means that the complementation of the two can better explore users' interests, while “Channel + Program Label” has the worst effect, probably because there is more information about program label, which causes the program label and channel information aliasing in the combination process, which caused mutual interference in semantic learning.

Based on the best two-factor combination, we conduct a multi-factor experiment, combining “Channel + Program Name” with other factors. There are three cases of three factors, and we determine the best three-factor combination and then carry out four-factor and all-factor combinations, so the whole multi-factor combination has six cases, the specific results are shown in Fig. 5. The overall fluctuation of indicators is not obvious. The combination of “Channel + Program Name + Program Synopsis” has the best effect. The more auxiliary information is not the better. Once too much auxiliary information is added, it will overload the model training load. In addition, the associated interference between different auxiliary information may lead to poor results. The effect of “Channel + Name + Label” is obviously better than the other results.

In the end, it can be seen from the combination of multiple factors that the results are not getting better as more information is integrated, which also indicates that the increase of information may confound the model training and not necessarily produce optimal results. The effects do not differ significantly when multiple factors are used overall, suggesting that the mixture of various factors allows the semantic factors to complement each other.

Conclusions

The scientific problem of this paper is the enhanced knowledge-level representation of auxiliary information for personalized program recommendation. To fully show the impact of dif-

Fig. 5 Multi-factor NDCG@5 index performance based on “channel + program name”

ferent auxiliary information on semantic representation, in this paper we propose an enhanced graph recommendation with heterogeneous auxiliary information, which is used to better explore the user-program relationship and investigate the outstanding effect of auxiliary information on semantic representation. The experimental results show that the performance of our models is significantly better than that of existing methods, indicating that auxiliary information, as supplementary knowledge, can better capture the semantic meaning of programs and has outstanding performance for semantic representation. However, the theoretical and technical research on semantic representation for personalized program recommendation is still in the initial stage. Although this paper has made some useful conclusions and achieved some research results, there are still problems such as fine-grained semantic representation and lack of multi-view data that have not been solved well. We will continue to work hard in subsequent research.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No. 61801440), the High-quality and Cutting-edge Disciplines Construction Project for Universities in Beijing (Internet Information, Communication University of China), State Key Laboratory of Media Convergence and Communication (Communication University of China), and the Fundamental Research Funds for the Central Universities.

Author Contributions FLY contributed conceptualization, methodology and supervision; MQJ contributed formal analysis, methodology, software and writing; STL contributed investigation, validation and writing; YYW contributed validation, visualization, supervision and writing; ZBY contributed visualization; XLF contributed investigation.

Funding This work was supported by the National Natural Science Foundation of China (No. 61801440), the High-quality and Cutting-edge Disciplines Construction Project for Universities in Beijing (Internet Information, Communication University of China), State Key Laboratory of Media Convergence and Communication (Communication University of China), and the Fundamental Research Funds for the Central Universities.

Availability of data and material We can provide these if needed.

Declarations

Conflict of interest The authors declare no competing interests.

Code availability We can provide these if needed.

Ethics approval All the authors have approved.

Consent to participate All the authors have approved this submission.

Consent for publication All the authors have approved publication.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the

source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Ao X, Derong L, Hongkang T, Zhengyuan L, Peng Y, Michel K (2021) News keyword extraction algorithm based on semantic clustering and word graph model. *Tsinghua Sci Technol* 26:886–893
2. Balabanovic M, Shoham Y (1997) Fab: content-based, collaborative recommendation. *Commun ACM* 40:66–72
3. Berg R, Kipf TN, Welling M (2017) Graph convolutional matrix completion. <https://arxiv.org/abs/1706.02263>
4. Bertram MG (1965) The managing organizations: the administrative struggle. *Ann Am Acad Pol Soc Sci* 360:197–198
5. Billsus D, Pazzani MJ (1998) Learning collaborative information filters. In: *Proceedings of the Fifteenth International Conference on Machine Learning*. pp 46–54
6. Chang J, Gao C, Zheng Y, et al. (2021) Sequential recommendation with graph neural networks. In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp 378–387
7. Cheng HT, Koc L, Harmsen J, et al. (2016) Wide & deep learning for recommender systems. In: *Proceedings of the 1st workshop on deep learning for recommender systems*. pp 7–10
8. Choudhury SS, Mohanty SN, Jagadev AK (2021) Multimodal trust based recommender system with machine learning approaches for movie recommendation. *Int J Inf Technol* 13: 475–482
9. Chuhan W, Fangzhao W, Tao Q, Suyu G, Yongfeng H, Xing X (2019) Reviews meet graphs: enhancing user and item representations for recommendation with hierarchical attentive graph neural network. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. pp 4884–4893
10. Covington P, Adams J, Sargin E (2016) Deep neural networks for youtube recommendations. In: *Proceedings of the 10th ACM conference on recommender systems*. pp 191–198
11. Gantner Z, Drumond L, Freudenthaler C, et al. (2010) Learning attribute-to-feature mappings for cold-start recommendations. In: *2010 IEEE International Conference on Data Mining*. pp 176–185
12. Gao M, Chen L, He X, et al. (2018) Bine: bipartite network embedding. In: *Proceedings of the 41st International ACM SIGIR conference on research and development in Information Retrieval*. pp 715–724
13. Guisheng Z, Pan S (2019) Design of film recommendation system based on IRGAN model and Hadoop. *Comput Appl Softw* 36:43–50
14. Guo H, Tang R, Ye Y, et al. (2017) DeepFM: a factorization-machine based neural network for CTR prediction. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. pp 1725–1731
15. Guo W, Yang Y, Hu Y, et al. (2021) Deep graph convolutional networks with hybrid normalization for accurate and diverse recommendation. In: *Proceedings of 3rd Workshop on Deep Learning Practice for High-Dimensional Sparse Data with KDD*

16. He X, Chua TS (2017) Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp 355–364
17. He X, Deng K, Wang X, et al. (2020) Lightgcn: simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. pp 639–648
18. He X, Liao L, Zhang H, Nie L, Hu X, Chua TS (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web. pp 173–182
19. Herlocker JL, Konstan JA, Borchers A et al (2017) An algorithmic framework for performing collaborative filtering. ACM SIGIR Forum 51:227–234
20. Hu L, Li C, Shi C et al (2020) Graph neural news recommendation with long-term and short-term interest modeling. Inf Process Manag 57:102142
21. Hui B, Zhang L, Zhou X, et al. (2021) Personalized recommendation system based on knowledge embedding and historical behavior. Appl Intell 52:954–966
22. Hwang WS, Lee HJ, Kim SW, et al. (2012) On using category experts for improving the performance and accuracy in recommender systems. In: Proceedings of the 21st ACM international conference on Information and knowledge management. pp 2355–2358
23. IJntema W, Goossen F, Frasincar F, et al. (2010) Ontology-based news recommendation. In: Proceedings of the 2010 EDBT/ICDT Workshops. pp 1–6
24. Ji K, Shen H, Tian H, et al. (2014) Two-phase layered learning recommendation via category structure. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp 13–24
25. Kim N, Oh S, Lee JH (2018) A television recommender system learning a user's time-aware watching patterns using quadratic programming. Appl Sci 8:1323
26. Kingma D, Ba JL (2014) Adam: A Method for Stochastic Optimization. <https://arxiv.org/abs/1412.6980>
27. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907)
28. Liu J, Wang Y, Wang G, et al. (2016) Personalized recommendation of live programs in cable television. In: 2016 5th International Conference on Computer Science and Network Technology. pp 268–272
29. Lops P, De Gemmis M, Semeraro G (2011) Content-based recommender systems: state of the art and trends. In: Ricci F, Rokach L, Shapira B, Kantor P (eds) Recommender systems handbook. Springer, Boston
30. Ma H, Lyu MR, King I (2009) Learning to recommend with trust and distrust relationships. In: Proceedings of the third ACM conference on Recommender systems. pp 189–196
31. Ma W, Zhang M, Wang C, et al. (2018) Your tweets reveal what you like: introducing cross-media content information into multi-domain recommendation. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. pp 3484–3490
32. Massa P, Bhattacharjee B (2004) Using trust in recommender systems: an experimental analysis. In: International conference on trust management. pp 221–235
33. Ni J, Huang Z, Hu Y et al (2021) A two-stage embedding model for recommendation with multimodal auxiliary information. Inf Sci (NY) 582:22–37
34. Paromita N, Joseph C, Praveen M (2021) Improvising personalized travel recommendation system with recency effects. Big Data Min Anal 4:139–154
35. Rathore SS, Kumar S (2017) A decision tree logic based recommendation system to select software fault prediction techniques. Computing 99:255–285
36. Rendle S, Freudenthaler C, Gantner Z, Thieme LS (2009) BPR: bayesian personalized ranking from implicit feedback. In: Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence. pp 452–461
37. Sankar A, Liu Y, Yu J et al (2021) Graph neural networks for friend ranking in large-scale social platforms. Proc Web Conf 2021:2535–2546
38. Sankar A, Zhang X, Krishnan A, et al. (2020) Inf-vae: a variational autoencoder framework to integrate homophily and influence in diffusion prediction. In: Proceedings of the 13th International Conference on Web Search and Data Mining. pp 510–518
39. Sarwar B, Karypis G, Konstan J, et al. (2001) Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web. pp 285–295
40. Schafer JB, Frankowski D, Herlocker J, et al. (2007) Collaborative filtering recommender Systems. In: The adaptive web. pp 291–324
41. Seo YD, Lee E, Kim YG (2020) Video on demand recommender system for internet proto-col television service based on explicit information fusion. Expert Syst Appl 143:113045
42. Shuai Z, Hongyan L, Jun H, Sanpu H, Xiaoyong D (2021) Deep sequential model for anchor recommendation on live streaming platforms. Big Data Min Anal 4:173–182
43. Sun Z, Yang J, Zhang J, et al. (2018) Recurrent knowledge graph embedding for effective recommendation. In: Proceedings of the 12th ACM Conference on Recommender Systems. pp 297–305
44. Tripathi AK, Mittal H, Saxena P et al (2021) A new recommendation system using map-reduce-based tournament empowered Whale optimization algorithm. Complex Intell Syst 7:297–309
45. Velickovic P, Cucurull G, Casanova A, et al. (2018) Graph attention networks. In: International Conference on Learning Representations
46. Wang M, Ma J (2016) A novel recommendation approach based on users' weighted trust relations and the rating similarities. Soft Comput 20:3981–3990
47. Wang R, Fu B, Fu G, et al. (2017) Deep & cross network for ad click predictions. In: Proceedings of the ADKDD'17. pp 1–7
48. Wang W, Ye C, Yang P, et al. (2020) Research on movie recommendation model based on LSTM and CNN. In: 2020 5th International Conference on Computational Intelligence and Applications. pp 28–32
49. Wang X, He X, Wang M, et al. (2019) Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. pp 165–174
50. Wu C, Wu F, Huang Y, et al. (2021) User-as-graph: user modeling with heterogeneous graph pooling for news recommendation. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence. pp 1624–1630
51. Wu S, Tang Y, Zhu Y, et al. (2019) Session-based recommendation with graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp 346–353
52. Xiao J, Ye H, He X, et al. (2017) Attentional factorization machines: learning the weight of feature interactions via attention networks. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp 3119–3125
53. Xuan Z, Zhongdao W, Lei G, Yali L, Shengjin W (2021) Incremental face clustering with optimal summary learning via graph convolutional network. Tsinghua Sci Technol 26:536–547
54. Ying R, He R, Chen K, et al. (2018) Graph convolutional neural networks for web-scale recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp 974–983
55. Yu T, Ruiqing Z, Zhenlan L, Suning L, Fangxiang W, Min L (2021) A data-driven clustering recommendation method for single-cell RNA-sequencing data. Tsinghua Sci Technol 26:772–789

56. Zhou G, Mou N, Fan Y et al (2019) Deep interest evolution network for click-through rate prediction. *Proc AAAI Conf Artif Intell* 33:5941–5948
57. Zhou G, Zhu X, Song C, et al. (2018) Deep interest network for click-through rate prediction. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. pp 1059–1068
58. Zhu Q, Zhou X, Song Z et al (2019) Dan: Deep attention neural network for news recommendation. *Proc AAAI Conf Artif Intell* 33:5973–5980

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.