Knowledge-Based Systems 277 (2023) 110822

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

A heterogeneous graph neural network model for list recommendation

Wenchuan Yang, Jichao Li, Suoyi Tan, Yuejin Tan, Xin Lu *

College of Systems Engineering, National University of Defense Technology, Changsha, Hunan, 410073, PR China

ARTICLE INFO

Article history:
Received 3 October 2022
Received in revised form 17 July 2023
Accepted 20 July 2023
Available online 25 July 2023

Keywords:
List recommendation
Bundle recommendation
Graph neural network
Heterogeneous information network

ABSTRACT

List recommendation, an approach for recommending a set of items closely related to user preferences, plays a significant role in improving the user experience and has received growing attention in recent years. However, most existing models ignore list-level attributes and the heterogeneity of the interactions between the users and lists when predicting user preferences. In this paper, we propose a heterogeneous graph neural network list recommendation (HGLR) model to bridge these gaps. First, a representation learning module based on pre-training techniques is developed to learn the embeddings of the lists, by encoding textual information and aggregating the features of the constituent items. We then transform the user–list interaction into a bipartite graph and incorporate the list attributes into a heterogeneous information network (HIN). Propagation-based embedding learning is subsequently performed to generate user and list representation based on the message-passing mechanism of the graph neural network. Finally, an attentive fusion layer is designed to obtain the global representations for the recommendation. We also develop a new large-scale dataset containing multiple types of side information to evaluate the list recommendation model. Extensive experiments demonstrate that our model significantly outperforms state-of-the-art list recommendation methods and achieves performance gains over 8.96% and 14.59% in Recall and NDCG, respectively, against the strongest baselines.

1. Introduction

In recent years, list recommendation (or bundle recommendation) has received growing attention in online services [1–3]. Specifically, compared with conventional recommendation [4–6], which is designed for recommending a single item, list or bundle recommendation aims to present users with a collection of items that coheres around a specific semantic concept (Fig. 1a). By presenting a variety of carefully designed lists, the search scope is largely narrowed down, and users can find their desired items more efficiently. Thus, list recommendation is an effective way for businesses to meet the users' complex interests and improve their experience.

Recently, efforts have been made to build list recommender systems. Most existing works [7–9] follow a multi-task learning framework, in which item and list recommendation models are trained sequentially or simultaneously. Subsequently, the learned parameters can be shared to enhance the performance of the list recommender system. Lately, it has been proposed to model the user–list/item interaction and list-item affiliation relations as graphs. Graph neural network (GNN)-based methods then are naturally used to perform representation learning for providing the recommendations [10–12]. Although these techniques achieve promising results to some extent, they still suffer from several limitations:

(1) **Drawback of multi-task learning framework**: User–item interactions are commonly much denser than user–list interactions; thus, it is difficult to balance two tasks in a loss function towards an optimal performance [13].

(2) **Lack of consideration for relation heterogeneity:** Existing recommendations approaches predominantly focus on characterizing user–list direct interactions, while neglecting the valuable side information that can reveal the heterogeneous interactive relations between the users and lists [14]. For instance, a user may be attracted to a list that shares the same tag as the lists they have interacted with. Such preferences cannot be identified by the traditional interaction modeling method.

(3) **Insufficient modeling of users' limited attention span**: Items in lists usually have varying impacts on decision-making [9] and should not be treated equally. Moreover, users have a browsing threshold beyond which users will stop browsing when a sufficient number of items are explored [15].

(4) **Overlook of list attributes**: A list has multimodal characteristics, such as titles and main images, which serve as prominent illustrations of the semantics or themes of the list [16]; however, these important attributes are often neglected in the existing studies.

To address these limitations, we propose an end-to-end **h**eterogeneous **g**raph neural network-based **l**ist **r**ecommendation

* Corresponding author.
E-mail address: xin.lu.lab@outlook.com (X. Lu).

https://doi.org/10.1016/j.knosys.2023.110822
0950-7051/

Fig. 1. (a) Lists in various forms (left) and (b) interaction process in playlist browsing (right).

(HGLR) model. The proposed model recognizes that both the intrinsic characteristics and constituting items of the list, are two key factors for learning the list representations. We generate list-level features from textual attributes and proposed a truncated aggregation mechanism to learn the item-level features that mimic a user's limited attention span. Then, we build a heterogeneous information network (HIN), to capture the complex collaborative signals between the users and lists, except for the user–list interaction graph. In the HIN the list attributes are introduced as a distinct node type to reveal latent user–item interactions. We utilize the message-passing mechanism inherited from the GNN methods to fully encode the collaborative signals and yield satisfactory embeddings. Furthermore, an attention network is designed to fuse the embeddings learned from different interaction patterns, to obtain global representations for the users and lists. Finally, we utilize a neural-network-based module to make predictions. The model is evaluated on a newly built dataset that is much larger than the existing benchmarks, and achieves significant improvement over the state-of-the-art baselines.

The main contributions of this work are as follows:

- We emphasize the importance of including the list-level properties in list recommendation and built a new large-scale benchmark dataset, the *Netease Playlist* for performance evaluation. To the best of our knowledge, *Netease Playlist* represents the largest publicly available collection of curated, ready-to-use benchmark datasets for list recommendation.
- We propose a general framework that can effectively incorporate auxiliary information into preference modeling for real-world list recommendation. We design a comprehensive learning module to obtain list representations from list and item views. We also leverage auxiliary information to enrich diverse connections between users and lists. This is the first study to introduce an attribute-based HIN for representation learning in a list-recommender system.
- Extensive experiments are conducted on the collected dataset *Netease Playlist*, and the results show the superiority of the proposed framework over the current state-of-the-art baselines for all metrics.

The remainder of this paper is organized as follows. Section 2 reviews the works related to list recommendations. Section 3 defines the problem and briefly describes the dataset. Section 4 provides a detailed description of the proposed framework. Section 5 details the experimental setup. Section 6 presents the results of the performance comparison, ablation study, and analysis of the effects of the hyperparameters. Finally, we conclude the paper in Section 7.

2. Related works

As lists are widely adopted by businesses and have become a popular and an efficient form of recommendation, a line of research has explored list recommendations. Early attempts [7,9,15,17] applied a multi-task learning framework in which the user and user–list recommendation models are trained simultaneously with shared parameters. He et al. [13] notice the difficulty in balancing weights among tasks and proposed a single hierarchical model for list recommendation. Some existing approaches [18,19] treat the user–item and -list interaction data as sequences and propose corresponding sequence-aware learning modules to learn the representations of the users and lists.

Recently, GNN methods [10,12] achieve significant improvements by capturing diverse connections through propagation. Interaction data are transformed into multiple bipartite [10] or unified tripartite graphs [12]. Wang et al. [20] add multiplication-based aggregation to the GNN messaging-passing mechanism to capture the interactions among neighbors. Vijaikumar et al. [11] propose adopting a graph attention network (GAT) to distinguish the importance of different neighbors. Zhang et al. [21] design a DT-CDBR framework that introduces dual-target cross-domain learning to improve the recommendation performance. Zhao et al. [22] consider user- and list-item interactions as the global and local views of the user's intention and proposed a GNN-based contrastive learning framework, MIDGN, that disentangles and compares different views to obtain fine-grained representations. Zhang et al. [23] generate multiple heterogeneous subgraphs around user–list pairs and utilized graph-level GNNs to extract the user preferences.

Although user- and list–item interactions are utilized to form a user–list heterogeneous graph [10–12,21], it is a simple combination of three bipartite graphs that are flawed and could fail to extract certain semantic relations. For instance, one can infer potential interests by finding similar users through item co-purchase relations in conventional recommender systems. However, in the list recommendation scenario, this inference fails because a single item cannot represent the properties of a list. Existing studies based on available interactions cannot cover diverse collaborative signals in the absence of list-level attributes.

During the examination of existing studies on list recommendation, we find another, perhaps more prominent, limitation: the evaluation of existing methods is excessively performed on only a few public benchmarked datasets. However, the limited scope of these datasets leads to several issues. (1) Because the largest public benchmark contains only 18,528 users, 22,864 lists, and 123,628 items, it is questionable whether the existing models can be generalized to untested large-scale datasets. (2) New models are proposed only when they exceed the baselines on these datasets, which might result in architectural overfitting [24,25].

Motivated by these caveats, in this study, we propose a novel architecture that can efficiently leverage auxiliary information for real-world list recommendation and construct a new large-scale evaluation dataset, containing various list-level information, for performance evaluation (e.g., list descriptions and genres).

3. Problem definition and dataset

User–list interaction description: The interaction behavior between users and lists can be interpreted as a sequential process. User's attention is first attracted by the overall theme (attributes) of a list; then, the user interacts with the items within the list, and finally, a decision is made based on the degree of satisfaction. Hence, it is necessary to include both the list-related attributes and item information in the recommendation. An illustration of this process regarding a playlist is shown in Fig. 1b.

Problem definition: We use $U = \{u_1, u_2, \ldots, u_{n1}\}, L = \{l_1, l_2, \ldots, l_{n2}\}$, *and* $V = \{v_1, v_2, \ldots, v_{n3}\}$ to denote the set of users, lists and items, respectively. The user–list and user–item interaction data are expressed as $\boldsymbol{X}_{n1\times n2} = \{x_{ul} | u \in \boldsymbol{U}, l \in \boldsymbol{L}\}$ and $\boldsymbol{Y}_{n1\times n3} = \{y_{uv} | u \in \boldsymbol{U}, v \in \boldsymbol{V}\}$, respectively. Each list can be described by a set of items following the display order $l_* = \{v_1^{l_*}, v_2^{l_*}, \ldots, v_{|l_*|}^{l_*}\}$, where $|l_*|$ is the length of l_*. Thus, we can obtain the list-item affiliation matrix $\boldsymbol{Z}_{n2\times n3}$. The attributes of the lists are denoted by $\boldsymbol{A} = \{A_1, A_2, \ldots, A_{n2}\}$, where A_* represents the attribute information of each list. Our goal in this study is to build a model F to predict the list that the user will most likely interact with based on the given information:

$$\forall_{u\in\boldsymbol{U}}, l'_u = \arg\max_{l\in\boldsymbol{L}} F(u, l|\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{A}) \tag{1}$$

Dataset description: To overcome the limitations of the existing benchmarks, such as the availability of large-scale datasets and list attributes, we construct a new list recommendation dataset by collecting data from Netease Cloud Music, a popular music-streaming platform based in China that has attracted over 800 million registered users since its launch [26]. Our dataset comprises information extracted from this platform, covering data up to December 2021. Ultimately, we obtain 14,833 users, 195,283 lists, 3,693,730 songs, 1,469,835 user–list interactions, and 7,695,867 user–song interactions. Notably, existing datasets do not contain all the data required for modeling, such as textual information. Thus, we also retrieve the important and common characteristics of the lists, including the title, description, and genres (or tags).

To ensure the completeness of the experiments, we follow the processing procedure in [9] and obtain a denser dataset in which we filter out the users and lists that appeared less than 25 times. The detailed statistics for the two datasets are presented in Table 1.

4. Methodology

In this section, we introduce the proposed HGLR model, which falls into four consecutive stages (Fig. 2): First, a list representation is generated by combining the list- and item-level embeddings, which are learned directly from textual information and by aggregating the features of the constituent items. Subsequently, based on the user–list interaction graph and constructed HIN, we develop an embedding learning mechanism inspired by graph convolutional networks to update user and list representations, by capturing diverse collaborative signals. In the following section, the user and list representations are fused into global representations using the designed attentive layers. Finally, a neural prediction module is adopted to provide recommendations using global user and item representations.

4.1. List representation learning module (Stage 1)

For the first stage of HGLR, a general learning module is carefully designed to consider the intrinsic properties of the lists and users' limited attention span to better model the list characteristics. Specifically, we propose a textual feature extraction method based on pre-training models to learn list-level characteristics and a truncated attention layer to aggregate the item-level features from the constituent items.

4.1.1. List-level features: leveraging text embeddings

Textual attributes contain intuitive information that can reveal the main topic of a list [16,27]. We collect the titles and descriptions of each list and apply the BERT [28] model to learn the concept embeddings of the lists.

We first remove stop words from the raw text and truncate the list descriptions to 510 words (title information remained unchanged). We then classify the list titles based on language and adopted pre-trained BERT-based models [29] with a mean pooling strategy to encode titles and descriptions. Subsequently, the title embeddings $\boldsymbol{t}$ and description embeddings $\boldsymbol{d}$ for the lists are obtained. To learn an aggregated list-level feature from the textual embeddings $\boldsymbol{l}_t$, we adopt a neural fusion layer as follows:

$$\boldsymbol{l}_t = \boldsymbol{W}_t(\alpha\boldsymbol{t} + \beta\boldsymbol{d}) + \boldsymbol{b}_t, \tag{2}$$

where α and β are learnable parameters deciding the weights for titles and descriptions, respectively, and $\boldsymbol{W}_t$ and $\boldsymbol{b}_t$ represent the learnable weight matrix and bias for the projection.

4.1.2. Item-level features: leveraging user–item interaction

Items within a list can affect user preferences; therefore, it is important to consider the characteristics of the list items when representing a list. We propose using pre-training techniques, such as MFBPR [30], to learn the initial representations of the items from user–item interactions. Inspired by [15], we hypothesize that only the top N items in the list play an important role in the users' decision-making processes because the attention span is limited. Besides, a list may contain numerous items that may affect the distinguishing ability of the attention mechanism. Hence, we design a truncated "attention" layer to generate item-level features, $\boldsymbol{l}_p$ as follows:

$$\boldsymbol{l}_p = \text{LeakyReLU}\left(\frac{1}{|\mathcal{N}_l|}\sum_{j\in\mathcal{N}_l}\boldsymbol{W}_p\boldsymbol{v}_j + \boldsymbol{b}_p\right), \tag{3}$$

where $\mathcal{N}_l$ represents the top N items in list l, $\boldsymbol{v_j}$ is the pre-trained item representation of the item j, $\boldsymbol{W}_p$ is a trainable transformation matrix, and LeakyReLU$(\cdot)$ is an activation function. In the proposed module, the attention mechanism is implemented by manually setting the attention span. A sensitivity analysis of $\mathcal{N}_l$ is presented in Section 6.4.

Table 1
Description and statistics of the *Netease Playlist* dataset (Netease for short).

	Statistics	
	Netease-Sparse	Netease-Dense
# User	14,833	8,346
# List	195,283	10,871
# Song	3,693,730	1,005,884
# User-List	1,469,835	838,640
# Average list interaction	99.09	100.48
# User-List sparsity	0.051%	0.92%
# User-Song	7,584,962	6,319,668
# Average song interaction	511.38	757.21
# User-Song sparsity	0.014%	0.08%
# Max list size	16,151	10,000
# Min list size	1	1
# Average bundle size	178.4	196.72
# Text information (title, description)	195,283 instances	10,871 instances
# Genre type	78	

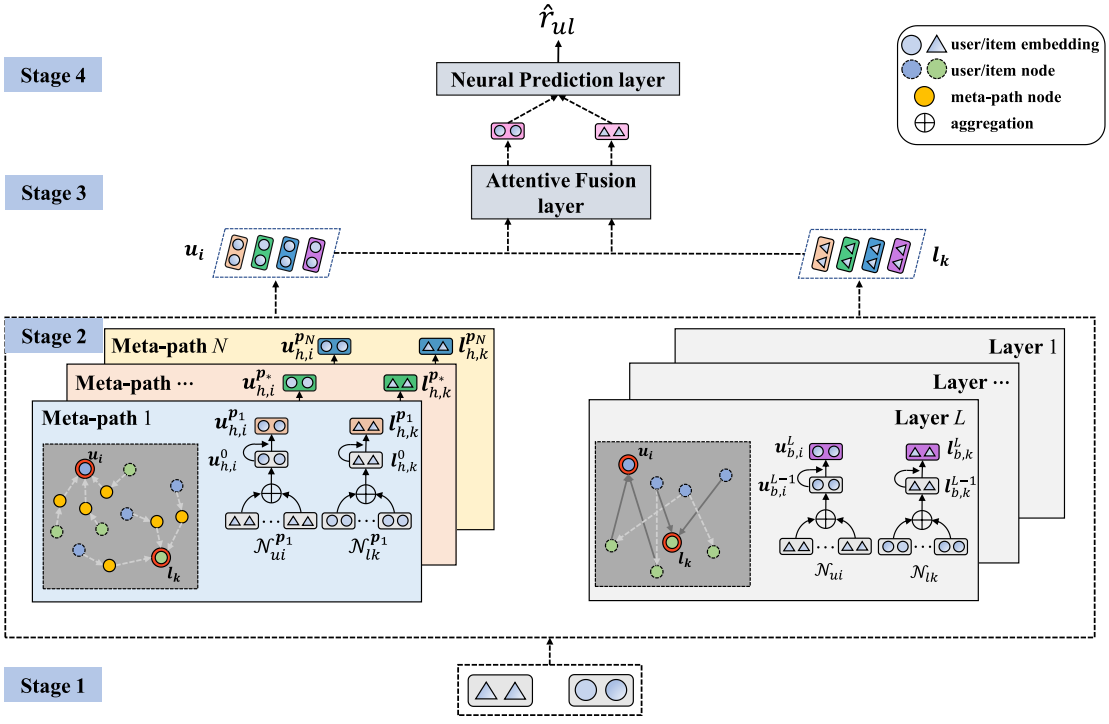

Fig. 2. Overall framework of the HGLR.

4.1.3. Gated fusion module

The learned list- and item-level embeddings can be added up using various techniques such as concatenation [31,32], summation [33], and element-wise product [34]. However, these techniques are not suitable in our scenario since items within the list can also reflect the overall theme of the list, which may overlap with the list-level features. The aggregation should be able to remove the redundant information.

To address this challenge, we adopt a gated fusion method to eliminate overlapping features, which can be described as follows:

$$\begin{cases} \boldsymbol{f} = tanh\left(\boldsymbol{W}_{\boldsymbol{f}}\left[\boldsymbol{l}_t \oplus \boldsymbol{l}_p\right] + \boldsymbol{b}_{\boldsymbol{f}}\right) \\ \boldsymbol{g} = sigmoid\left(\boldsymbol{W}_{\boldsymbol{g}}\left[\boldsymbol{l}_t \oplus \boldsymbol{l}_p\right] + \boldsymbol{b}_{\boldsymbol{g}}\right) \\ \boldsymbol{l}_c^{(0)} = \boldsymbol{g} \odot \boldsymbol{f} + (1 - \boldsymbol{g}) \odot \boldsymbol{l}_t \end{cases} \tag{4}$$

where $\boldsymbol{W}_*$ and $\boldsymbol{b}_*$ denote trainable weight matrices and vectors, respectively, $\odot$ denotes the element-wise production, $\boldsymbol{f}$ stands for the learned feature that contains the overall characteristics of the list, $\boldsymbol{g}$ is the element-wise weight vector serving as the gate, and $\boldsymbol{l}_c^{(0)}$ is the fused representation prepared for a propagation-based embedding learning module.

4.2. Propagation-based embedding learning (Stage 2 & 3)

The goal of the propagation-based embedding learning module is to learn the latent representations for the users and lists from different interaction patterns. Specifically, we propose bipartite- and HIN-view information propagation layers to embed direct interaction information and reveal implicit diverse interacting relations.

4.2.1. user–list bipartite graph learning

The user–list interaction data are expressed by a bipartite graph, $G_1 = (V_1, E_1)$, where V_1 refers to the set of users and lists, and E_1 denotes the set of connections between the users and lists. To capture the user preferences and interactive characteristics of the lists for the recommendation tasks, we build a propagation-based embedding propagation layer between the users and lists.

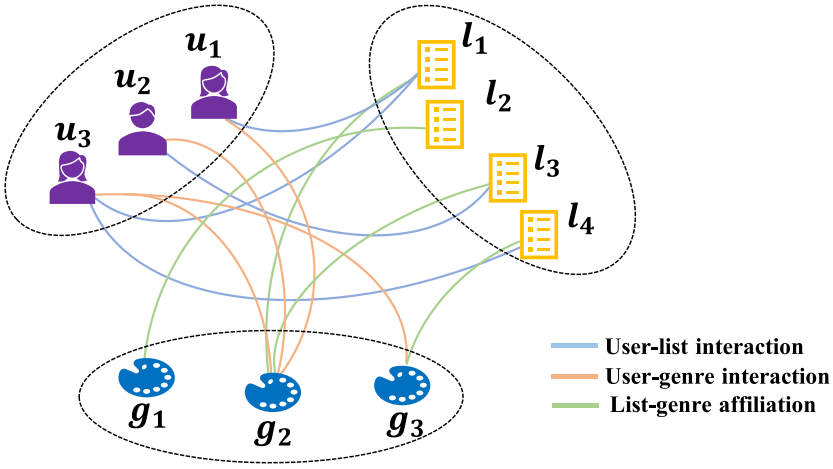

Fig. 3. Illustration of the HIN.

The embedding update rules for the user u_i and list l_k can be formulated as follows:

$$\begin{aligned}\mathbf{u}_{b,i}^{l+1} &= \sigma\left(\mathbf{W}_1^{l+1} \times \left(\mathbf{u}_i^l + \frac{1}{|\mathcal{N}_{ui}|}\sum_{k\in\mathcal{N}_{ui}} \boldsymbol{l}_k^l + \mathbf{b}_1^{l+1}\right)\right),\\ \boldsymbol{l}_{b,k}^{l+1} &= \sigma\left(\mathbf{W}_2^{l+1} \times \left(\boldsymbol{l}_k^l + \frac{1}{|\mathcal{N}_{lk}|}\sum_{k\in\mathcal{N}_{lk}} \boldsymbol{u}_i^l + \mathbf{b}_2^{l+1}\right)\right),\end{aligned} \tag{5}$$

where $\mathbf{u}_i^{l+1}$ and $\boldsymbol{l}_k^{l+1} \in$ denote the updated embedding of the user u_i and list l_k, respectively, on the $(l+1)$th layer. $\mathbf{W}_1^{l+1}$ and $\mathbf{W}_2^{l+1}$ are the learned weights in the $l+1$ step. $\mathbf{b}_1^{l+1}$ and $\mathbf{b}_2^{l+1}$ are the biases in the $l+1$ step. $\mathcal{N}_{ui}$ represents the lists subscribed by the user u_i, $\mathcal{N}_{lk}$ stands for all the users who subscribed l_k. LeakyReLU$(\cdot)$ is used as the activation function. Specifically, $u_{b,i}^0 = u_c^{(0)}$ *and* $l_{b,k}^0 = l_c^{(0)}$, where $\boldsymbol{u}_c^{(0)}$ is a randomly initialized embedding for the user u_i, and $\boldsymbol{l}_{b,k}^0$ is obtained from the learned $\boldsymbol{l}_c^{(0)}$.

4.2.2. user–list-attributeHIN learning

To capture the implicit connections between the users and lists, we first generate a user–genre interaction graph using matrix multiplication based on the user–list interaction and list-genre affiliation information. Next, we incorporate the user–genre relation into user–list interactions to build a complete HIN [14]. Subsequently, diverse interactions guided by various meta-paths can be identified [35]. For instance, according to Fig. 3, we can obtain the neighbor set of the user u_1 through the meta-path UGUL ($\mathcal{N}_{UGUL}(u_1) = \{l_3\}$), which is inaccessible in a bipartite graph (the bipartite neighbors are l_1 and l_4).

To incorporate the newly obtained neighborhood information into the representation learning, we adopt a restricted message-passing layer to aggregate the neighbor information as follows:

$$\boldsymbol{u}_{h,i}^{p_*} = \text{LeakyReLU}\left(\boldsymbol{W}_3(\boldsymbol{u}_{h,i}^0 + \frac{1}{|\mathcal{N}_u^{p_*}|}\sum_{k\in\mathcal{N}_u^{p_*}} \boldsymbol{l}_k) + \boldsymbol{b}_3\right), \tag{6}$$

$$\boldsymbol{l}_{h,k}^{p_*} = \text{LeakyReLU}\left(\boldsymbol{W}_4(\boldsymbol{l}_{h,k}^0 + \frac{1}{|\mathcal{N}_l^{p_*}|}\sum_{i\in\mathcal{N}_l^{p_*}} \boldsymbol{u}_i) + \boldsymbol{b}_4\right), \tag{7}$$

where $\boldsymbol{u}_{h,i}^{p_*}$ and $\boldsymbol{l}_{h,i}^{p_*}$ denote the aggregated embeddings based on the neighbors generated by the meta-path p_*, with $\boldsymbol{l}_{h,k}^0 = \boldsymbol{u}_c^{(0)}$ and $\boldsymbol{u}_{h,i}^0 = \boldsymbol{l}_c^{(0)}$, respectively. $\mathcal{N}_u^{p_*}$ and $\mathcal{N}_l^{p_*}$ are the neighbor sets of the obtained users and lists, respectively, guided by the meta-paths. $\boldsymbol{W}_*$ and $\boldsymbol{b}_*$ represent the learnable weight matrix and bias, respectively. Notably, the size of the neighbor set is limited; hence, we set a threshold b and sample the neighbors based on the path-count value [36].

4.2.3. Attentive fusion layer

Given the bipartite graph and a set of meta-paths $\{p_1, p_2, \ldots, p_t\}$, we can obtain the corresponding latent representations $\{\boldsymbol{u}_b, \boldsymbol{u}_h^{p_1}, \boldsymbol{u}_h^{p_2}, \ldots, \boldsymbol{u}_h^{p_t}\}$ and $\{\boldsymbol{l}_b, \boldsymbol{l}_h^{p_1}, \boldsymbol{l}_h^{p_2}, \ldots, \boldsymbol{l}_h^{p_t}\}$ for the users and lists, respectively. It should be noted that different interactive patterns are not equally important when reflecting the users' global preferences and overall characteristics of the lists. We apply an attention network to fuse the embeddings through dynamic weight learning.

$$\boldsymbol{u}_g = \alpha_1 \cdot \boldsymbol{u}_b + \sum_{i=2}^{t+1} \alpha_i \cdot \boldsymbol{u}_h^{p_{(i-1)}}, \tag{8}$$

$$\boldsymbol{l}_g = \beta_1 \cdot \boldsymbol{l}_b + \sum_{i=2}^{t+1} \beta_i \cdot \boldsymbol{l}_h^{p_{(i-1)}}, \tag{9}$$

where $\boldsymbol{u}_g$ and $\boldsymbol{l}_g$ are the global representations of the users and lists, respectively. α_i and β_i are the weights learned by the attention network.

$$\alpha_1 = \boldsymbol{g}_u^T \tanh\left(\mathbf{W}_u\boldsymbol{u}_b + \mathbf{b}_u\right), \alpha_i = \boldsymbol{g}_u^T \tanh\left(\mathbf{W}_u\boldsymbol{u}_h^{p_{(i-1)}} + \mathbf{b}_u\right), \text{ and}$$

$$\alpha_i = \frac{\exp(\alpha_i)}{\sum_{n=1}^{t+1}\exp(\alpha_n)}, \tag{10}$$

$$\beta_1 = \boldsymbol{g}_l^T \tanh\left(\mathbf{W}_l\boldsymbol{l}_b + \mathbf{b}_l\right), \beta_i = \boldsymbol{g}_l^T \tanh\left(\mathbf{W}_l\boldsymbol{l}_h^{p_{(i-1)}} + \mathbf{b}_l\right), \text{ and}$$

$$\beta_i = \frac{\exp(\beta_i)}{\sum_{n=1}^{t+1}\exp(\beta_n)}, \tag{11}$$

where $\boldsymbol{W}_*$ is the weight matrix, $\boldsymbol{b}_*$ is the bias vector, and $\boldsymbol{g}_*$ is the learnable projection vector. The *Softmax* function is utilized for normalization, and we employ *tanh* as the activation function.

4.3. Prediction and training (Stage 4)

The estimated preference score $\hat{r}_{ul}$ is calculated using the user global representation $\boldsymbol{u}_g$ and list global representation $\boldsymbol{l}_g$. In line with [8,37], we adopt both concatenation and element-wise product to make the prediction. The element-wise product captures the collaborative signals embedded in the interaction, whereas a concatenation operation is used to avoid information loss.

Concatenation: A neural transformation module is applied to learn the corresponding predictive factors from the concatenated representation.

$$\begin{aligned}
&\boldsymbol{h}_1^{(0)} = \boldsymbol{u}_g \oplus \boldsymbol{l}_g, \\
&\boldsymbol{h}_1^{(1)} = \boldsymbol{W}_1^{(0)} \boldsymbol{h}_1^{(0)} + \boldsymbol{b}_1^{(0)}, \\
&\cdots \\
&\boldsymbol{r}_1 = \boldsymbol{W}_1^{(l)} \boldsymbol{h}_1^{(l)} + \boldsymbol{b}_1^{(l)},
\end{aligned} \tag{12}$$

where $\boldsymbol{r}_1$ denotes the predictive factor generated from the concatenation $\oplus$, and $\boldsymbol{W}_1^{(*)}$ and $\boldsymbol{b}_1^{(*)}$ are the learned weight matrix and bias, respectively.

Element-wise product: Initially we adopt two MLP (Multilayer Perceptron) layers to separately transform the representations of the users and lists, and perform the element-wise product as follows:

$$\begin{aligned}
&\boldsymbol{h}_{2,u}^{(0)} = \boldsymbol{u}_g, \boldsymbol{h}_{2,l}^{(0)} = \boldsymbol{l}_g, \\
&\boldsymbol{h}_{2,u}^{(1)} = \boldsymbol{W}_{2,u}^{(0)} \boldsymbol{h}_{2,u}^{(0)} + \boldsymbol{b}_{2,u}^{(0)}, \boldsymbol{h}_{2,l}^{(1)} = \boldsymbol{W}_{2,l}^{(0)} \boldsymbol{h}_{2,l}^{(0)} + \boldsymbol{b}_{2,l}^{(0)}, \\
&\cdots \\
&\boldsymbol{r}_{2,u} = \boldsymbol{W}_{2,u}^{(l)} \boldsymbol{h}_{2,u}^{(l)} + \boldsymbol{b}_{2,u}^{(l)}, \boldsymbol{r}_{2,l} = \boldsymbol{W}_{2,l}^{(l)} \boldsymbol{h}_{2,l}^{(l)} + \boldsymbol{b}_{2,l}^{(l)}, \\
&\boldsymbol{r}_2 = \boldsymbol{r}_{2,u} \odot \boldsymbol{r}_{2,l}
\end{aligned} \tag{13}$$

where $\odot$ denotes the element-wise product operation, $\boldsymbol{r}_2$ stands for the subsequently obtained predictive factor, $\boldsymbol{W}_{2,*}^{(*)}$ is the learned matrix, and $\boldsymbol{b}_{2,*}^{(*)}$ is the bias vector. Specifically, the activation functions in the prediction module are removed due to their poor performance validated by sufficient experiments.

The preference score $\hat{r}_{ul}$ is then calculated as follows:

$$\hat{r}_{ul} = \mathbf{W}_p \left(\boldsymbol{r}_1 \oplus \boldsymbol{r}_2\right) + \mathbf{b}_p, \tag{14}$$

where $\mathbf{W}_p$ and $\mathbf{b}_p$ denote the weight matrix and bias of the prediction layer, respectively.

To optimize the proposed framework, we utilize a pairwise learning framework that is extensively used in recommender systems [15,30]. We sample a set of triplets $\mathcal{R} = \left\{\left(u, l, l'\right) \mid (u, l) \in G_1, \left(u, l'\right) \notin G_1\right\}$ for model training. Thus, the overall objective function can be expressed as:

$$L = \sum_{(u,l,l') \in \mathcal{R}} - \ln \sigma \left(\hat{r}_{ul}\left(\Theta\right) - \hat{r}_{ul'}\left(\Theta\right)\right), \tag{15}$$

where $\sigma(\cdot)$ is the sigmoid function, and Θ represents the overall model parameters.

5. Experimental setup

In this section, we provide a detailed introduction to the evaluation criteria, baselines, and parameter settings used in our implementation.

Evaluation criteria: We adopt the widely used *leave-one-out* evaluation protocol [38] to accelerate the evaluation time in the experiments. Specifically, for each user, we randomly sample one of their list interactions as a test instance (the remaining interactions are used for the training dataset). We then pair each test instance with 99 randomly sampled unobserved lists with which the user had not interacted, to construct the test dataset. In addition, we use two metrics: normalized discounted cumulative gain *(NDCG)* [39] and *Recall* at 5 and 10 to evaluate the performance of the top-N recommendations.

Baselines: We compare the proposed HGLR model with seven state-of-the-art methods for list recommendation.

- **MFBPR** [30] is a matrix factorization method optimized using Bayesian personalized ranking pairwise learning. These are widely used in conventional recommendation systems.
- **NGCF** [39] is a neural network-based model that captures the collaborative signals between users and items, based on a user–item bipartite graph. In this study, we apply this to the user–list bipartite graph.
- **GCN** [40] is a recently proposed GNN model based on graphs for providing recommendations. Similarly, we apply a GCN to the user–list bipartite graph.
- **BR** [7] is a two-stage approach for listing recommendations. In the first stage, it learns the user and item representations using the matrix factorization under a pairwise learning framework. In the second stage, the learned parameters are utilized to represent lists in the MF-based user list recommendation model.
- **DAM** [9] adopts a multi-task learning method for list recommendation. User–item and user–list interactions are modeled simultaneously to exploit the knowledge from the user–item interactions, to enhance list recommendation. In addition, a deep attention network is designed to aggregate the item information for list representation.
- **BGCN** [10] treats the various interactions as three bipartite graphs and employs the graph convolutional neural network to learn the user and list representations.
- **MIDGN** [22] extracts global and local intents from the user and list-item interactions, respectively, using the GNN model and subsequently adopts a contrastive learning framework to learn the final representations for the users and lists, and achieves state-of-the-art performance recently.

Parameters: In list representation learning, the default output size of text embedding is 768 [29], and we utilize a single-layer MLP with an output layer of 64 units for projection. The dimensions of the pre-trained latent vector, obtained from the MFBPR, are set to 64. The embedding size is fixed at 64 in the subsequent sections. Herein, we provide a general module for learning the properties of lists that can be obtained using different methods. For example, textual embeddings, $\boldsymbol{t}$ and $\boldsymbol{d}$, can be computed using ERNIE [41] or fine-tuning techniques [42,43].

In bipartite graph learning, the depth of the propagation layer and the neural prediction layer are both set to 2, and the hidden dimensions are 64. There are 16 hidden units in the attentive fusion layer. The batch size and learning rate are determined to be [128, 256, 512, 1024] and [1e−4, 1e−3, 1e−2], respectively, and the batch size 1024 and learning rate 1e−3 are selected for the experiments. The meta-paths used to discover the heterogeneous neighbors in the HIN are listed in Table 2. We implement our HGLR model based on Pytorch, and the experiments are conducted in a Windows server with Intel(R) i7-10870H CPU @ 2.2 GHz (8 cores), 32 G RAM, and GeForce RTX 3070 Ti. The code is available at https://github.com/Chuan1997/HGLR.

Time complexity: To update the representations of the users and lists, the time complexity of HGLR is given by $O\left(\left(L\left\|A\right\|_0 F + LNF^2\right) \times |R|\right)$. Here, L stands for the number of propagation layers, F is the dimension of embeddings, $\|A\|_0$ represents the number of non-zeros in the adjacency matrix A, N denotes the number of users and lists, and $|R|$ is the number of meta-paths used. Notably, the time complexity is the same as that when employing multiple GCNs [40] because we consider multiple meta-paths in our framework.

6. Results and analysis

6.1. Overall performance comparison

Table 3 summarizes the results of our experiments on the two Netease datasets in terms of the metrics containing Recall @5, NDCG@5, Recall@10, and NDCG@10.

Table 2
Semantic meaning of the meta-paths designed for datasets.

Meta-paths	Semantic meaning
UGL	A user is interested in a list that is compatible with the user's genre preference.
UGUL	A user is interested in the list that is subscribed by users who have similar tastes.
ULGL	A user is interested in the list which shares the same genre as the observed lists.
LGU	A list may be subscribed by a user who is in favor of its genre.
LGLU	A list may be subscribed by a user who has subscribed to lists with the same genre.
LUGU	A list may be subscribed by a user who is similar to the interacted users.

Table 3
Performance comparison on two real-world datasets.

Method	Netease-Sparse				Netease-Dense			
	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@5	NDCG@5	Recall@10	NDCG@10
BPR	0.241	0.141	0.429	0.200	0.332	0.226	0.473	0.271
NGCF	0.198	0.136	0.305	0.170	0.101	0.063	0.165	0.083
GCN	0.467	0.299	0.664	0.364	0.226	0.150	0.332	0.184
BR	0.574	0.372	0.720	0.420	0.302	0.201	0.457	0.251
DAM	0.586	0.458	0.681	0.486	0.353	0.246	0.473	0.286
BGCN	/	/	/	/	0.417	0.283	0.558	0.329
MIDGN	/	/	/	/	0.403	0.278	0.549	0.325
HGLR	**0.690**	**0.540**	**0.791**	**0.572**	**0.457**	**0.328**	**0.608**	**0.377**
Improved %	**17.75**	**17.90**	**16.15**	**17.70**	**9.59**	**15.90**	**8.96**	**14.59**

The proposed HGLR method consistently outperforms all the baselines by a large margin. Specifically, HGLR exceeds the best baseline DAM by 17.86%, 17.81%, 16.22%, and 17.71% for Recall @5, NDCG@5, Recall@10, and NDCG@10, respectively, on Netease-Sparse. For Netease-Dense, we observe that our model achieves Recall-based improvements of 6.52% to 6.98% and NDCG-based improvements of 10.98% to 11.26%.

Among the item-based recommendation models, the GCN largely outperforms BPR and NGCF. One possible reason for this is that the GCN can effectively exploit the rich collaborative signals from high-order connections by stacking layers in a bipartite interaction graph. However, list recommendation models such as BR and DAM achieve higher Recall and NDCG than item-based recommendation models on both datasets, suggesting the necessity of leveraging user–item interactions to enhance the list recommendation performance. In particular, DAM performs better than BR, indicating that the multi-task learning framework with a deep neural network is more effective in transferring the information learned from user–item interactions than a multi-stage approach.

The newly proposed graph-based list recommendation models, such as BGCN and MIDGN, perform the best among the baselines because they explore diverse relations or intents among users, lists, and items simultaneously. However, neither BGCN nor MIDGN could be reproduced on the Netease-Sparse. A possible reason is that BGCN utilizes matrix multiplication to calculate the overlap intensity between lists, whereas MIDGN needs to create multiple extra intent-aware graphs for the user–list interaction modeling, which is not feasible owing to memory constraints when dealing with large-scale datasets such as Netease-Sparse. Although MIDGN obtains the highest score in the two existing datasets [22], its performance on the Netease dataset is relatively poor when compared with BGCN, suggesting that attention should be paid to the architectural overfitting in the list recommendation scenario. Thus, generalization ability should be emphasized in the further research.

Additionally, we empirically compare the proposed HGLR with competitive baselines in terms of training speed on Netease-Dense, as presented in Table 4. HGLR is the fastest with only an average of 3.51 s spent for each training epoch.

In conclusion, HGLR can generate a more comprehensive list representation than the existing solutions, by capturing multi-level characteristics, and is capable of exploring heterogeneous connections between users and lists to uncover the users' complex preferences. In particular, HGLR shows excellent scalability on large-scale datasets in terms of performance and effectiveness, whereas the existing methods experience generalization failure or increased training costs.

6.2. Ablation study

Since HGLR consists of several important design decisions, such as the inclusion of list-level attributes and HIN-based embedding learning, we conduct an ablation study to investigate the effects of these modules on the recommendation performance. We present the results of HGLR versus multiple variants in Tables 5 and 6.

Without heterogeneous information network (W/O hete for short): The HIN is removed in propagation-based embedding learning in Stage 2, which means we only perform representation learning on a bipartite graph. The results show that the performance worsened in almost all metrics, with a significant drop. With an increase in data sparsity, the HIN plays an increasingly important role, with the maximum performance drop reaching 20.09% in NDCG@5 for Netease-Sparse. This demonstrates that the introduction of genre information can reveal diverse connections between the users and lists and capture comprehensive user preferences.

Without item-level features (W/O item for short): The item-based embedding learning module with truncated attention is removed when learning the representations of the lists. As shown in Tables 5 and 6, a sharp drop is observed in all criteria. In particular, the removal results in average drops of 15.67% and 13.29% in terms of NDCG on Netease-Sparse and Netease-Dense, respectively. These findings prove that user–item interactions provide useful information in modeling interactions and that our proposed truncated attention mechanism is effective in summarizing the characteristics of the list.

Without text embeddings (W/O text for short): We remove textual information encoding in the list representation learning in Stage 1. The effect of removing list-level attributes is significant. On the Netease-Sparse dataset, both Recall and NDCG dropped sharply by 9.95% and 11.8%, respectively, on average. The contribution of textual information varies between the two datasets. One reason for this is that we use a simpler text feature extraction technique such that some useful information might

Table 4
Empirical time comparison on Netease-Dense.

	DAM	BGCN	MIDGN	HGLR
Avg. time per epoch	60.94 s	22.32 s	339.38 s	**3.51 s**

Table 5
Ablation analysis on Netease-Sparse dataset.

	Recall@5		NDCG@5		Recall@10		NDCG@10	
HGLR	**0.6904**		**0.5397**		**0.7910**		**0.5723**	
W/O hete	0.6045	−12.44%	0.4313	−20.09%	0.7599	−3.93%	0.4729	−17.37%
W/O item	0.6006	−13.01%	0.4551	−15.68%	0.7366	−6.88%	0.4948	−15.66%
W/O text	0.6314	−8.55%	0.4707	−12.78%	0.7613	−3.79%	0.5104	−10.82%
W/O des	0.6750	−2.23%	0.5181	−4.00%	0.7833	−0.97%	0.5528	−3.41%
W/O title	0.6803	−1.46%	0.5266	−2.43%	0.7838	−0.91%	0.5603	−2.1%
R/W add	0.6414	−7.09%	0.4771	−11.59%	0.7742	−2.12%	0.5191	−9.29%
R/W product	0.6173	−10.59%	0.4584	−15.06%	0.7510	−5.06%	0.5003	−12.58%

Table 6
Ablation analysis on Netease-Dense dataset.

	Recall@5		NDCG@5		Recall@10		NDCG@10	
HGLR	**0.4572**		**0.3280**		**0.6083**		**0.3773**	
W/O hete	0.4329	−4.02%	0.3051	−5.24%	0.5835	−4.08%	0.3538	−6.23%
W/O item	0.3956	−13.47%	0.2831	−13.69%	0.5376	−11.62%	0.3287	−12.88%
W/O text	0.4456	−2.54%	0.3127	−4.66%	0.5939	−2.42%	0.3604	−4.48%
W/O des	0.4493	−1.75%	0.3214	−2.01%	0.5980	−1.69%	0.3687	−2.33%
W/O title	0.4402	−3.72%	0.3148	−4.02%	0.5893	−3.12%	0.3622	−4.00%
R/W add	0.4491	−1.77%	0.3212	−2.07%	0.6017	−1.08%	0.3709	−1.70%
R/W product	0.4497	−1.64%	0.3211	−2.10%	0.6023	−0.98%	0.3690	−2.25%

be unexploited. Future studies should explore more accurate and advanced methods for encoding text in the recommendation.

Without title or descriptions (**W/O title**, **W/O des** for short): We further investigate the isolated effect of titles and descriptions in textual information when encoding the list representations. We observe a relatively small impact on the performance of our proposed framework, and the decrease is similar for the two types of information. A possible reason for this is that the titles and descriptions may have overlapping information; hence, the removal effects could be similar.

Replace with simply add or element-wise product (**R/W add**, **R/W product** for short): In this case, we replace the gated fusion module in the list representation learning by simply adding an element-wise product strategy. The experimental results show that these two strategies are inferior to the gated fusion method. On the Netease-Sparse dataset, we observe a significant performance drop compared with HGLR. This proves that gated fusion is effective in preserving representative characteristics and removing noisy information.

6.3. Effect of meta-paths

In this subsection, we examine the use of the selected meta-paths on the two datasets. The empirical results are shown in Fig. 4. For clarity, *UGL-LGU* means that we use only the interaction information provided by this meta-path to make the recommendations.

It can be seen that different meta-paths make different contributions to the datasets. For Netease-Dense, as shown in Fig. 4(b), the meta-path *UGL-LGU* obtains the best performance, which is also superior to the existing state-of-the-art bundle recommendation methods. Even the "worst" meta-path UGUL-LGLU has a better performance than the conventional recommendation models. For the Netease-Sparse dataset, the meta-path *UGUL-LGLU* becomes the most informative choice and is beneficial for enhancing the recommendation performance (Fig. 4(a)).

The experiment demonstrates that the chosen meta-paths contain significant interaction patterns for making a recommendation, proving the effectiveness of the meta-paths used in HGLR. In addition, the superiority of the single metapath-based HGLR indicates the rationality of our framework.

6.4. Impact of hyper-parameters

In this section, we focus on two representative hyperparameters of HGLR to discuss their impact on the performance.

6.4.1. Truncated list length for learning item-level features

Fig. 4(a) shows that, the performance of our model increases, peaks at $\mathcal{N}_l = 20$, and immediately experiences performance degradation when the truncated list length keeps increasing. The upward tendency verifies that constituent items reflect the characteristics of the lists, whereas the downward curve proves the existence of the limited attention span.

6.4.2. Neighborhood size of HIN

As shown in Fig. 5(b), as the neighborhood size $\mathcal{N}_*^{p_*}$ increases from 50 to 200, the performance of HGLR increases accordingly. The best performance is achieved at $\mathcal{N}_*^{p_*} = 200$. This pattern indicates that, with a larger neighborhood size, more collaborative signals can be utilized for interaction modeling, resulting in better performance.

6.5. Case study

We present a case study based on the Netease-Dense dataset to validate the superiority of the proposed HGLR, as shown in Fig. 6. Specifically, we randomly select a user ($uid = 3866$) and deliver recommendations from the test dataset. Additionally, we employ a state-of-the-art BGCN model for comparison. As illustrated in Fig. 6, HGLR can correctly predict the target item ($lid = 1841$) as the user's next interaction, while BGCN suggests that

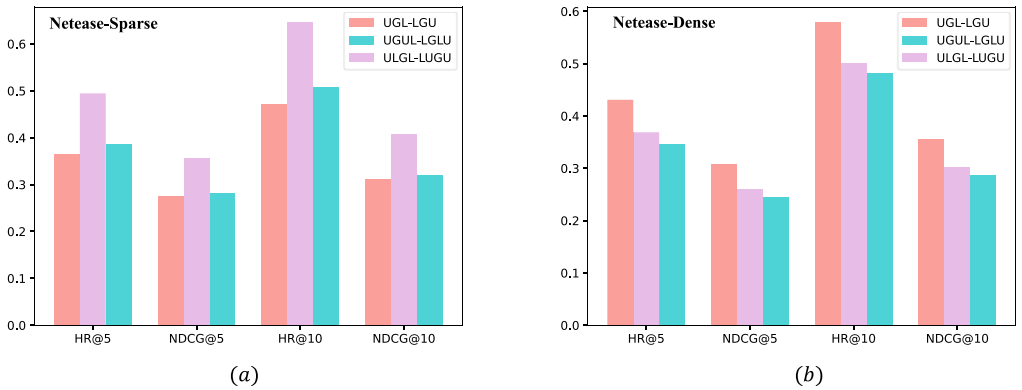

Fig. 4. Effects of different meta-paths.

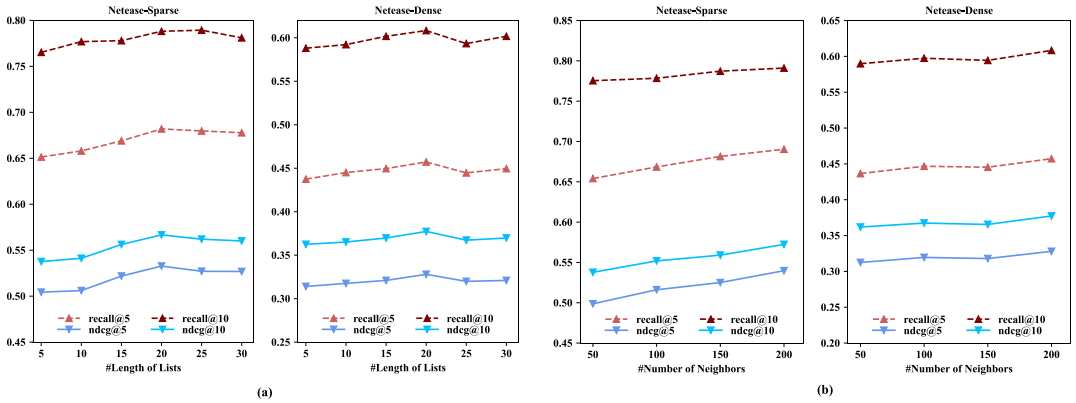

Fig. 5. Hyper-parameters analysis. (a) Optimal number of items when learning item-level representation for the lists. (b) Optimal size of the neighbor sets in the HIN.

Fig. 6. Case study.

the item ($iid = 1841$) should be out of the top-10 list. Specifically, BGCN is confused by similar items like $item_{1372}$ and $item_{2432}$. However, HGLR is capable of identifying the difference between the candidate item and target item, thus ranks $item_{1841}$ in the first place.

We further present a visualization example of how the attentive fusion layer works in HGLR based on the Netease-Dense dataset in Fig. 7. We randomly sample one user ($uid = 4005$) and calculate the attentive weights for $user_{3866}$ and $user_{4005}$. This shows that the attentive weights on meta-paths vary depending on the users and that the U-G-L-based interactions are crucial for user modeling, proving that the genre attribute is effective in finding the hidden interaction patterns. Additionally, we visualize the weights for a randomly selected list, revealing that the original U-L interactions are useful for mining the properties of the list.

7. Conclusions

In this study, we focus on a new recommendation paradigm called list recommendation, whereby a user's sophisticated interests can be expressed through various lists. We propose an HGLR that incorporates heterogeneous information into an end-to-end GNN recommendation framework. Specifically, we design a pre-training technique-based list representation learning module that efficiently fused the textual information and item-level features. To mine heterogeneous relations, we construct a HIN based on the attribute information to enrich the pairwise interaction between the users and lists. On the bipartite interaction graph and constructed HIN, the embeddings propagated among the interactions following the message-passing mechanism. An attentive layer is proposed for fusion and for obtaining the global representations of the users and lists. The experimental results verify

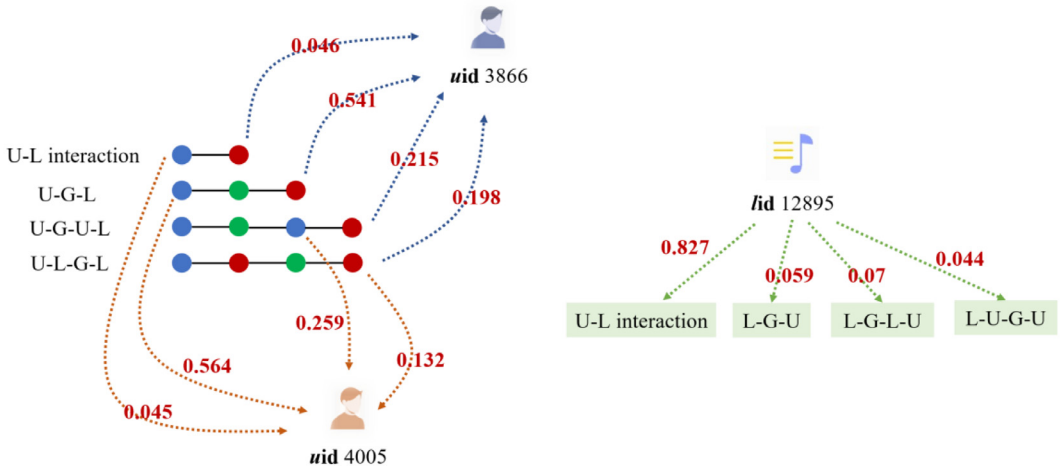

Fig. 7. Example of the designed attentive fusion.

the superiority of the proposed HGLR in terms of both recommendation accuracy and scalability. Moreover, ablation analysis provides solid support for each carefully designed module.

Based on the experimental results, architectural overfitting is observed. Although the newly proposed model can obtain accurate predictions, significant improvements in the ability to generalize to various scenarios have been neglected because of the limited diversity of the testing dataset. Therefore, we create a high-quality *Netease Playlist* dataset, which is the largest to date in comparison to the existing benchmarks, to facilitate the research community for further exploration and evaluation of the list recommendations.

In our future work, we plan to design an adaptive function to learn the personalized attention span of each user and study an item-bundling strategy for high-quality lists.

CRediT authorship contribution statement

Wenchuan Yang: Writing – original draft, Validation, Methodology, Data curation. **Jichao Li:** Data curation, Conceptualization. **Suoyi Tan:** Writing – review & editing, Validation. **Yuejin Tan:** Supervision, Funding acquisition, Conceptualization. **Xin Lu:** Writing – review & editing, Visualization, Resources, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (72025405, 72088101, 72001211), the National Social Science Foundation of China (22ZDA102), the Hunan Science and Technology Plan Project (2020TP1013, 2020JJ4673, 2022JJ20047, 2023JJ40685), the Shenzhen Basic Research Project for Development of Science and Technology (JCYJ20200109141218676, 20200829172650001), and the Innovation Team Project of Colleges in Guangdong Province (2020KCXTD040).

References

[1] P. Kouki, I. Fountalis, N. Vasiloglou, N. Yan, U. Ahsan, K.A. Jadda, H. Qu, Product collection recommendation in online retail, in: Proceedings of the 13th ACM Conference on Recommender Systems, 2019, pp. 486–490.
[2] P. Papreja, H. Venkateswara, S. Panchanathan, Representation, exploration and recommendation of playlists, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2019, pp. 543–550.
[3] D.D. Le, H.W. Lauw, Collaborative curating for discovery and expansion of visual clusters, in: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, pp. 544–552.
[4] J. Liu, Y. Chen, X. Huang, J. Li, G. Min, Gnn-based long-and-short term preference modeling for next-location prediction, Inform. Sci. 629 (2023) 1–14.
[5] G. Liu, L. Zhang, J. Wu, Beyond similarity: Relation-based collaborative filtering, IEEE Trans. Knowl. Data Eng. 35 (1) (2021) 128–140.
[6] W. Yang, J. Li, S. Tan, Y. Tan, X. Lu, Feature-enhanced embedding learning for heterogeneous collaborative filtering, Neural Comput. Appl. 34 (21) (2022) 18741–18756.
[7] A. Pathak, K. Gupta, J. McAuley, Generating and personalizing bundle recommendations on steam, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 1073–1076.
[8] C. Yang, L. Miao, B. Jiang, D. Li, D. Cao, Gated and attentive neural collaborative filtering for user generated list recommendation, Knowl.-Based Syst. 187 (2020) 104839.
[9] L. Chen, Y. Liu, X. He, L. Gao, Z. Zheng, Matching user with item set: Collaborative bundle recommendation with deep attention network, in: IJCAI, 2019, pp. 2095–2101.
[10] J. Chang, C. Gao, X. He, D. Jin, Y. Li, Bundle recommendation and generation with graph neural networks, IEEE Trans. Knowl. Data Eng.
[11] M. Vijaikumar, S. Shevade, M.N. Murty, Gram-smot: Top-n personalized bundle recommendation via graph attention mechanism and submodular optimization, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2020, pp. 297–313.
[12] Q. Deng, K. Wang, M. Zhao, Z. Zou, R. Wu, J. Tao, C. Fan, L. Chen, Personalized bundle recommendation in online games, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2381–2388.
[13] Y. He, J. Wang, W. Niu, J. Caverlee, A hierarchical self-attentive model for recommending user-generated item lists, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1481–1490.
[14] C. Shi, B. Hu, W.X. Zhao, S.Y. Philip, Heterogeneous information network embedding for recommendation, IEEE Trans. Knowl. Data Eng. 31 (2) (2018) 357–370.
[15] Y. Liu, M. Xie, L.V. Lakshmanan, Recommending user generated item lists, in: Proceedings of the 8th ACM Conference on Recommender Systems, 2014, pp. 185–192.
[16] H. Tzaban, I. Guy, A. Greenstein-Messica, A. Dagan, L. Rokach, B. Shapira, Product bundle identification using semi-supervised learning, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 791–800.

[17] D. Cao, L. Nie, X. He, X. Wei, S. Zhu, T.-S. Chua, Embedding factorization models for jointly recommending items and user generated lists, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 585–594.
[18] B. Li, B. Jin, X. Dong, W. Zhuo, List recommendation via co-attentive user preference fine-tuning, in: International Conference on Neural Information Processing, Springer, 2021, pp. 554–562.
[19] B. Li, B. Jin, X. Dong, W. Zhuo, Multiple: Multi-level user preference learning for list recommendation, in: International Conference on Web Information Systems Engineering, Springer, 2021, pp. 221–236.
[20] X. Wang, X. Liu, J. Liu, H. Wu, Relational graph neural network with neighbor interactions for bundle recommendation service, in: 2021 IEEE International Conference on Web Services, ICWS, IEEE, 2021, pp. 167–172.
[21] T. Zhang, Y. Han, X. Dong, Y. Xu, Y. Shen, Dual-target cross-domain bundle recommendation, in: 2021 IEEE International Conference on Services Computing, SCC, IEEE, 2021, pp. 183–192.
[22] S. Zhao, W. Wei, D. Zou, X. Mao, Multi-view intent disentangle graph networks for bundle recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 4379–4387.
[23] Z. Zhang, B. Du, H. Tong, Suger: A subgraph-based graph convolutional network method for bundle recommendation, arXiv preprint arXiv:2205.11231.
[24] Y. Nie, A. Williams, E. Dinan, M. Bansal, J. Weston, D. Kiela, Adversarial NLI: A new benchmark for natural language understanding, arXiv preprint arXiv:1910.14599.
[25] B. Recht, R. Roelofs, L. Schmidt, V. Shankar, Do cifar-10 classifiers generalize to cifar-10? arXiv preprint arXiv:1806.00451.
[26] NetEase, Netease cloud music, 1997, https://music.163.com/.
[27] P. Kouki, I. Fountalis, N. Vasiloglou, N. Yan, U. Ahsan, K.A. Jadda, H. Qu, Product collection recommendation in online retail, in: Proceedings of the 13th ACM Conference on Recommender Systems, 2019, pp. 486–490.
[28] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805.
[29] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al., Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2020, pp. 38–45.
[30] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, arXiv preprint arXiv:1205.2618.
[31] T. Bai, J.-R. Wen, J. Zhang, W.X. Zhao, A neural collaborative filtering model with interaction-based neighborhood, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 1979–1982.
[32] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.
[33] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, T.-S. Chua, Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 335–344.
[34] Y. Zhang, Q. Ai, X. Chen, W.B. Croft, Joint representation learning for top-n recommendation with heterogeneous information sources, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 1449–1458.
[35] Y. Sun, J. Han, X. Yan, P.S. Yu, T. Wu, Pathsim: Meta path-based top-k similarity search in heterogeneous information networks, Proc. VLDB Endow. 4 (11) (2011) 992–1003.
[36] L. Chen, Y. Liu, Z. Zheng, P. Yu, Heterogeneous neural attentive factorization machine for rating prediction, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 833–842.
[37] C. Wang, Z. Guo, G. Li, J. Li, P. Pan, K. Liu, A light heterogeneous graph collaborative filtering model using textual information, Knowl.-Based Syst. 234 (2021) 107602.
[38] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 426–434.
[39] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165–174.
[40] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907.
[41] Y. Sun, S. Wang, Y. Li, S. Feng, H. Tian, H. Wu, H. Wang, Ernie 2.0: A continual pre-training framework for language understanding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 8968–8975.
[42] C. Sun, X. Qiu, Y. Xu, X. Huang, How to fine-tune bert for text classification? in: China National Conference on Chinese Computational Linguistics, Springer, 2019, pp. 194–206.
[43] E.B. Zaken, S. Ravfogel, Y. Goldberg, Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models, arXiv preprint arXiv:2106.10199.