Expert Systems With Applications 191 (2022) 116234

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Multi-view Graph Attention Network for Travel Recommendation

Lei Chen [a], Jie Cao [b,*], Youquan Wang [a], Weichao Liang [b], Guixiang Zhu [a]

[a] College of Information Engineering, Nanjing University of Finance and Economics, Nanjing, China
[b] College of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China

ARTICLE INFO

Keywords:
Recommender systems
Travel recommendation
Neural network
Graph neural network
Personalized attention

ABSTRACT

As an e-commerce feature, the recommender system can enhance the consumer shopping experience and create huge benefits for businesses. The e-tourism has become one of the largest service industries with the application and popularity of recommender systems. Many studies have confirmed that the travel product recommendation is widely different from traditional recommendations. Due to the financial and time costs, travel products are usually browsed and purchased relatively infrequently compared with other traditional products (*e.g.,* books, movies and food). In addition, choosing the appropriate travel product will be influenced by many factors, such as departure, destination and price. To tackle this challenging problem, we propose a MV-GAN (short for **M**ulti-**V**iew **G**raph **A**ttention **N**etwork for travel recommendation) model. It enriches user and product semantics through both metapath-guided neighbors aggregation and multi-view fusion in heterogeneous travel product recommendation graph. In particular, we design node-level and path-level attention networks for learning user and product representations from every single view. To collaboratively integrate multiple types of relationships in different views, a view-level attention mechanism is proposed to aggregate the node representations and obtain global user and product representations. We evaluate the proposed method on a public dataset and a dataset constructed from a large tourism e-commerce website in China. Extensive experiments not only validate the effectiveness of MV-GAN, but also show its potentially good interpretability.

1. Introduction

Tourism is one of the world's largest service industries (Kolahkaj et al., 2020). According to the UNWTO report,[1] the contribution of tourism to global GDP is more than 1.34 trillion U.S. dollars. Indeed, with improved living standards, tourism has become a popular leisure activity undertaken by billions of tourists per annum. As a trend, more and more online travel agencies (OTA), such as Expedia, Booking.com, TripAdvisor, Trip.com, and Tuniu keep popping up to provide online services. However, a significant problem that tourists must face is how to choose travel products that exactly match their preferences from massive online travel information. On the other hand, to improve the economic benefits, OTAs have to understand tourists' personalized preferences to attract and retain them, and finally improve conversions from browsers to purchasers.

In order to enhance tourist satisfaction and corporate revenue, travel recommendations have emerged. However, compared with traditional products, such as books, foods and movies, travel recommendations have unique characteristics. These characteristics pose a significant challenge to design a systematic and flexible recommender framework for personalized travel product recommendation.

Firstly, travel data are extremely sparser than traditional products. Since travel is not a necessity of life and requires more time and financial cost, users usually do not browse the travel website for a long time. Taking Figs. 1(a) and 1(b) as an example, it compares the data sparseness between Tuniu dataset Tuniu_D_1 (Chen, Wu et al., 2020) in our experiment and two classical datasets (MovieLens-100K Kapetanakis et al., 2020 and Tmall Zhu et al., 2018) for recommender system. We select the cumulative distribution function (CDF) as the statistical index. CDF is the probability that takes a value less or equal to the corresponding x-value. The results are shown in Fig. 1(a). We can find that over 99.9% of users in the Tuniu dataset have clicked less than 50 times, while nearly 40% of both MovieLens and Tmall datasets have clicked and rated less than 50 items. A similar observation can be seen from Fig. 1(b). The number of clicks on each item is much smaller in the Tuniu dataset than in other ones. The extreme sparseness of the travel data makes it difficult to directly apply traditional recommendation techniques, such as collaborative filtering and matrix factorization, to realize personalized travel recommendations.

* Corresponding author at: College of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China.
E-mail addresses: chenleinjust@gmail.com (L. Chen), caojie690929@163.com (J. Cao), youq.wang@gmail.com (Y. Wang), weicliang@foxmail.com (W. Liang), zgx881205@gmail.com (G. Zhu).

[1] http://www.unwto.org/.

https://doi.org/10.1016/j.eswa.2021.116234
Received 22 August 2021; Received in revised form 24 October 2021; Accepted 12 November 2021
Available online 25 November 2021

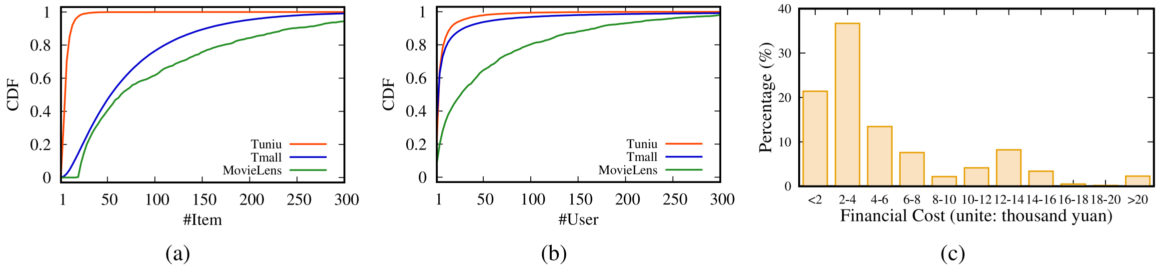


Fig. 1. Characteristics of Tuniu data. Figs. 1(a) and 1(b) compare data sparseness among three real-world datasets. Fig. 1(c) plots the distribution of financial costs of travel products.

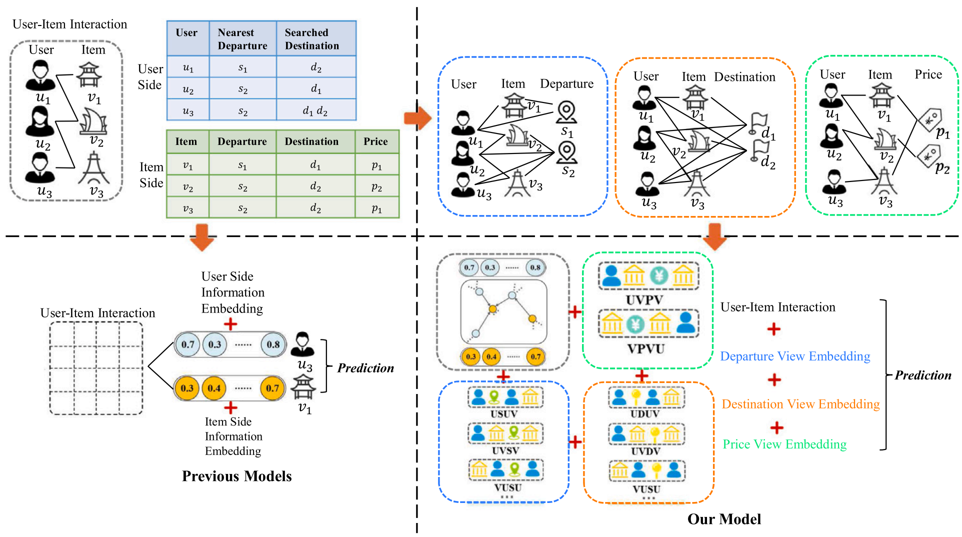


Fig. 2. The difference between our proposed MV-GAN model and the previous travel product recommendation models.

Secondly, the price of travel products varies very much. As shown in Fig. 1(c), the travel products are sold for dozens of dollars to thousands of dollars. Unlike other attributes, such as product category and product title that affect user preferences, the price determines whether a user is willing to pay for it. Without considering financial cost, the travel recommendation is likely to recommend a travel product to a user who cannot afford it because of the price. Many existing studies (Wan et al., 2017; Zheng et al., 2020) have confirmed that price is a principal element influencing user behaviors and product sales in marketing research. Nevertheless, it is rarely considered in the recommender system.

Thirdly, travel product includes departure and destination. To reduce the financial and time cost, tourists usually select a city closer to their residence (inferred by IP address) as the departure city. For example, if a tourist who lives in Nanjing plans to purchase outbound travel products, he/she will generally choose Nanjing or Shanghai as the departure city instead of Beijing. In addition, a tourist often searches a specific query, such as a destination city, to find travel products that match his/her travel intention. Queries can explicitly reveal users' intentions, we believe that queries (*i.e.,* destination cities) could help users make informed decisions and improve the likelihood of purchase.

Many travel recommendation methods (Chen, Cao, Zhu et al., 2021; Chen, Wu et al., 2020; He et al., 2016) have been proposed to leverage auxiliary information of users and products to address the data sparsity problem. For example, Ge et al. (2014) try to integrate the financial and time cost information with the latent factor model to improve recommendation accuracy. Liu et al. (2014) consider temporal–spatial correlations of travel data and propose a Bayesian network. Recently, Zhu et al. (2021) utilize complex descriptive information of the travel product to learn product representation by an attentive mechanism. However, these research works commonly regard departure, destination and price as product attributes, and fuse them into latent factor models, matrix factorization or deep learning models. The complex relations among users, travel products and attributes are paid less attention. For instance, a user may click a travel product that other users have clicked with the same searched destination.

Inspired by the recent developments of Graph Neural Network (GNN) (Cao et al., 2021; Wu et al., 2020; Zheng et al., 2020), which can fuse rich semantic information among different nodes, we consider using GNN to learn the complex relations among users, travel products and attributes. This paper proposes a new solution named **M**ulti-**V**iew **G**raph **A**ttention **N**etwork for travel recommendation (called MV-GAN), which employs the recently emerged metapath and GNN to learn the attribute-aware user representations and product representations.

Specifically, to learn the user embeddings and product embeddings under different views, we build four single-view graphs, *i.e.,* the user–product graph, the user–product–departure graph, the user–product–destination graph, and the user–product–price graph. These four graphs have different structures and contain different semantic information. We first apply GNN on a user–product graph to capture user and product's free embeddings through the graph structure after transformation and aggregation operations. Then, for the other three views, we utilize the metapath-guided neighbors to aggregate rich neighbors information, where different metapaths are assigned different attention weights according to the semantic information, and obtain the attribute embeddings of user and product. After combining the free embeddings and attribute embeddings via view-level attention networks, we can get the global user representations and product representations. Fig. 2 shows the difference between our proposed MV-GAN model and the previous travel product recommendation models. To sum up, our main contributions can be summarized as follows:

- To our best knowledge, this is the first attempt to leverage GNN to learn user and product representations in the travel recommender system. MV-GAN leverages metapath-guided neighbors to build other connections between user and product to alleviate the problem of data sparseness.
- MV-GAN learns user and product representations in multi-view graph embedding networks and explores view-level attention networks for integrating the user and product information from multiple views.
- MV-GAN is evaluated on real-world datasets. The results demonstrate that MV-GAN significantly outperforms the state-of-the-art approaches in recommender systems.

The remainder of this paper is organized as follows. Section 2 reviews the work related to tourism-oriented recommendations and graph neural network-based recommendations. In Section 3, the background and problem definition for travel recommendation is present. The proposed model MV-GAN for travel recommendation is presented in Section 4. We exhibit the experimental results in Section 5. Finally, we conclude the paper with future research directions in Section 6.

2. Related work

2.1. Tourism-oriented recommendations

Generally, the tourism-oriented recommendations can roughly fall into the following two categories. One school is POI recommendation, which predicts the next POI according to user preferences, or further connects POIs as an itinerary that satisfies temporal and spatial constraints. Plenty of side information, such as POI category (Sojahrood & Taleai, 2021), POI textual information (Chen, Cao, Chen et al., 2021; Chen, Zhang et al., 2020), social information (Liu et al., 2018; Zhao et al., 2018) and geographical information (Yang et al., 2017) have been exploited to improve the recommendation quality. For instance, JRLM++ (Zhao et al., 2018) captures unified user and POI representations by modeling check-in sequences together with social connections. PACE (Yang et al., 2017) considers social and geographical information, and proposes a context embedding framework combined with collaborative filtering for POI recommendation. However, these approaches consider only user-POI interactions and side information, and ignore the complex relations among user, POI and side information.

Although many studies fall within the aforementioned field, this work is highly related to the second school: travel product or travel package recommendation. Related literature (Ge et al., 2014; He et al., 2016; Liu et al., 2014) has demonstrated that travel recommendation has unique characteristics, such as being extremely sparse, involving multi-auxiliary information and unavailable ratings. To address this challenge, Chen, Wu et al. (2020) propose a matrix factorization model with auxiliary content information. Consider product category, product title and destination, Zhu et al. (2021) propose a neural attentive travel product recommendation method. Nevertheless, most related studies focus on several specific types of factors to improve recommendation accuracy and little attention is paid to design a systematic and flexible framework to incorporate all-round knowledge for travel recommendation. Furthermore, they primarily regard attributes as hidden feature vectors and fused them into the matrix factorization or deep learning models. The complicated relations among users, travel products and attributes are paid less attention.

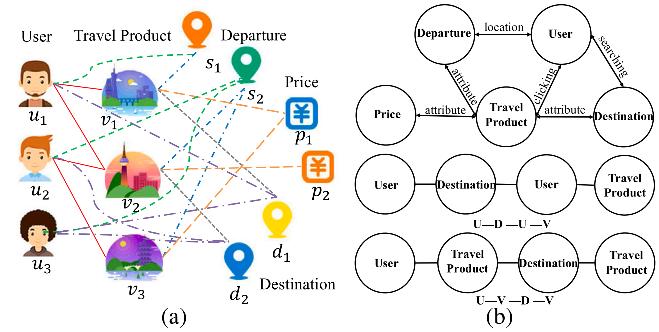


Fig. 3. Heterogeneous information network based travel recommendation. Fig. 3(a) is network instance. Fig. 3(b) is network schema and examples of metapaths.

2.2. Graph neural network-based recommendations

GNNs have received increasing attention due to their ability to model complex relations between objects, and have widely applied in many domains (Liu et al., 2020; Ma et al., 2021; Su et al., 2021), such as node classification, node clustering, link prediction, and graph classification. Since user–item interactions can be quite naturally regarded as a graph structure, researchers have attempted to adopt GNN on recommendation tasks (Wan et al., 2020). For instance, Wu et al. (2019) present SR-GNN to use gated graph networks to capture complex transitions of items so as to obtain accurate item embedding. On this basis, Xu et al. (2019) further extend SR-GNN and propose GC-SAN, which can get contextualized non-local representations via a node-level attention mechanism. Despite their effectiveness, these methods are built for homogeneous networks, which is not suitable for dealing with graphs including many types and rich relations in the recommender system.

Recently, several works (Fan et al., 2019; Zheng et al., 2020) have taken notice of the benefits of heterogeneous information networks (HIN) for the recommendation. For instance, Wang, He et al. (2019) extend GNNs for HIN and generalize them for the recommendation. Wang et al. (2020) propose a multi-component graph convolutional collaborative filtering method to explore the purchasing motivations underneath the edges in a user–item bipartite graph. Jin et al. (2020) consider multiple types of user–item interaction behaviors, such as clicks and purchases, and propose a multi-behavior graph convolutional network. Our proposed MV-GAN belongs to a HIN-based recommendation model. However, it distinguishes itself from the work mentioned above based on the following facts: (1) Unlike most of HIN-based recommendations that focus on binary user–product click interaction, MV-GAN uses the relation among user, attribute and travel product to design metapaths and build other connections between user and product to alleviate the problem of data sparseness. (2) Our proposed method is more interpretable, since each view learns user representations and product representations from a different perspective that carries semantics.

3. Preliminaries

This section introduces several preliminaries and formalizes the problem of travel recommendations.

Definition 1 (*Heterogeneous Information Network* Shi et al., 2016**).** Given a directed graph $\mathcal{G} = \{\mathcal{O}, \mathcal{E}, \varphi, \psi\}$ with nodes set $\mathcal{O}$ and edges set $\mathcal{E}$. If there are a node type mapping function $\varphi(\cdot)$ and an edge type mapping function $\psi(\cdot)$, the nodes or edges are mapped to different types, *i.e.,* $\varphi(o) \rightarrow \mathcal{T}(o \in \mathcal{O})$, $\psi(e) \rightarrow \mathcal{R}(e \in \mathcal{E})$, then $\mathcal{G}$ is the information network. If the types of nodes $|\mathcal{T}| > 1$ or the types of edges $|\mathcal{R}| > 1$, then $\mathcal{G}$ is called a heterogeneous information network. In addition, the network schema is denoted as $\mathbf{S}_G = \{\mathcal{T}, \mathcal{R}\}$.

Example 1. Fig. 3(a) shows a HIN example on travel data. The network consists of five types of nodes (*i.e.,* User (U), Travel Product (V), Departure (S), Destination (D) and Price (P)) and their interaction relationships (*e.g.,* the edge between the user and the travel product represents the relationship of clicking or being clicked).

Definition 2 (*Metapath* Fan et al., 2019)**.** A meta path P is a path defined on a network schema $\mathbf{S}_G = \{\mathcal{T}, \mathcal{R}\}$, and is denoted in the form of $T_1 \xrightarrow{R_1} T_2 \xrightarrow{R_2} \cdots \xrightarrow{R_k} T_{k+1}$, which defines the composite relation $R_1 \circ R_2 \circ \cdots \circ R_k$ between nodes T_1 and T_{k+1}, where $\circ$ represents the composition operator on relations. For simplicity, the metapath can be abbreviated as $P = (T_1 T_2 \cdots T_{k+1})$, and the concrete path $p = (t_1 t_2 \cdots t_{k+1})$ between nodes t_1 and t_{k+1} in the network $\mathcal{G}$ is called the path instance of the relevance path P.

Example 2. Take Fig. 3(b) as an example, users and travel products can be connected via different metapaths, and these paths represent different semantics. The path "User–Destination-User–Travel Product (UDUV)" indicates a user may click a travel product that other users have clicked with the same searched destination. And path "User–Travel Product–Destination-Travel Product (UVDV)" means a user may click a travel product that has the same destination as the travel product he has clicked.

Definition 3 (*Metapath-guided Neighbors* Wang, Ji et al., 2019)**.** Given a node o and a metapath P starting from node o, the metapath-guide neighbors is defined as the set of nodes that connect with node o via metapath P.

Example 3. Taking Fig. 3(a) as an example, given the metapath "UDUV" and a user u_3, we can obtain the metapath-guided neighbors $\mathcal{N}_{UDUV}(u_3) = \{v_1, v_2, v_3\}$. We can get metapath-based neighbors by the multiplication of a sequence of adjacency matrices.

Definition 4 (*Heterogeneous Information Network based Travel Recommendation*)**.** Given a heterogeneous information network $\mathcal{G} = \{\mathcal{O}, \mathcal{E}\}$, where the node set $\mathcal{O}$ contains five type nodes (*i.e.,* User (U), Travel Product (V), Departure (S), Destination (D) and Price (P)) and their corresponding relations (*i.e.,* User–Travel Product (u, v), Travel Product–Departure (v, s), Travel Product–Destination (v, d), Travel Product–Price (v, p), User–Departure (u, s), and User–Destination (u, d)). The goal is to predict the click probability r_{uv} of the user $u \in \mathcal{U}$ on the unclicked travel product $v \in \mathcal{V}$.

Similar to Zheng et al. (2020), we discretize price values into separate levels to facilitate modeling. Less than 2000 and greater than 20 000 are divided into two price ranges, and the remaining prices are divided into 9 price ranges on average.

4. The MV-GAN model

This section proposes a recommendation method based on a multi-view graph attention network, and the framework is presented in Fig. 4. Following many embedding models (Tong et al., 2021; Wu et al., 2020), we hypothesize that node embedding is composed of free embedding and attribute embeddings, which are learned from the user–product view and three user–product-attribute views. After learning view-specific user and product representations for each single view, we develop view-level attention networks to learn view weights by enabling users (products) to concentrate on the most informative view. Then, user and product representations for each individual view are fused into the global user and product representations via the learned view-level attention weights. Finally, the global user and product representations will be used for the recommendation task.

4.1. User–product view embedding

To capture the free embeddings of user and product, we build upon an embedded propagation layer between user and product. The propagation-based and pooling-based embedding updating rules for user u and product v can be formulated as below:

$$\mathbf{u}_i^{0,l+1} = \sigma(\mathbf{W}^{l+1} \times (\mathbf{u}_i^{0,l} + AGG(\mathbf{v}_j^{0,l} | j \in \mathcal{N}_{ui}) + \mathbf{b}^{l+1})), \tag{1}$$

$$\mathbf{v}_j^{0,l+1} = \sigma(\mathbf{W}^{l+1} \times (\mathbf{v}_j^{0,l} + AGG(\mathbf{u}_i^{0,l} | j \in \mathcal{N}_{vj}) + \mathbf{b}^{l+1})), \tag{2}$$

where $\mathbf{u}_i^{0,l} \in \mathbb{R}^{D\times 1}$ is free embedding of user u_i on lth layer and $\mathbf{v}_j^{0,l} \in \mathbb{R}^{D\times 1}$ is free embedding of product v_j on lth layer. D is the embedding size. $\mathbf{W}^{l+1} \in \mathbb{R}^{D\times D}$ and $\mathbf{b}^{l+1} \in \mathbb{R}^{D\times D}$ are learned weight and learned bias in $l+1$ step, σ is non-linear activation function $LeakReLU$. $\mathcal{N}_{ui}$ means the travel products clicked by user u_i, $\mathcal{N}_{vj}$ denotes the users that click item v_j. $AGG(\cdot)$ is the aggregate function, such as averaging and max-pooling operation. In this article, we use the mean aggregator. $\mathbf{u}_i^{0,0} \in \mathbb{R}^{D\times 1}$ and $\mathbf{v}_j^{0,0} \in \mathbb{R}^{D\times 1}$ are the ID embeddings of user u_i and product v_j learned by the trainable ID embedding matrices.

4.2. User–product-attribute view embedding

To learn the attribute embeddings of user and travel product under the influence of the departure, destination and price, we aggregate the metapath-guided neighbors in the attribute view (departure view, destination view, and price view). Attribute embeddings of user and travel product in different view are the aggregation of their neighbors under different metapaths. For instance, the embedding of user u_3 is calculated based on several metapaths, such as UDUV, UVDV and UDV. According to the network structure in Fig. 3(a), we obtain the first step neighbor set of u_3, $\mathcal{N}_{UDUV}(u_3) = \{v_1, v_2, v_3\}$. the embeddings of metapath-neighbors are aggregated to obtain the embeddings of the $\mathbf{u}_3^{UDUV}$. Along this line, we can learn different metapath-guided embeddings of u_3, *e.g.,* $\mathbf{u}_3^{UVDV}$ and $\mathbf{u}_3^{UDV}$. Finally, we aggregate all metapath-guided embeddings to obtain the u_3's final embedding $\mathbf{u}_3$.

The effect of different metapath-guided neighbors on learning embeddings is different. Thus, we use a node-level attention mechanism to learn the importance of different nodes and fuse them for the node embedding. The calculating process is given:

$$\mathbf{h}_i^P = \sum_{j\in\mathcal{N}_i^P} \alpha_{ij}^P \cdot \mathbf{o}_j + \mathbf{o}_i, \tag{3}$$

where $\mathbf{h}_i^P$ is the learned embedding of node o_i under the metapath P, and α_{ij}^P represents the node-level attention weight of node o_j to node o_i in the metapath P. $\mathbf{o}_i$ and $\mathbf{o}_j$ are ID embeddings of nodes $o_i \in \mathbb{R}^{D\times 1}$ and $o_j \in \mathbb{R}^{D\times 1}$.

$$\alpha_{ij}^P = \mathbf{X}_P^{\top} tanh(\mathbf{V}_P \mathbf{o}_i + \mathbf{W}_P \mathbf{o}_j + \mathbf{b}_P),$$
$$\alpha_{ij}^P = \frac{exp(\alpha_{ij}^P)}{\sum_{k\in\mathcal{N}_i^P} exp(\alpha_{ik}^P)}, \tag{4}$$

where $\mathbf{X}_P \in \mathbb{R}^{D_p\times 1}$, $\mathbf{V}_P \in \mathbb{R}^{D_p\times D}$, $\mathbf{W}_P \in \mathbb{R}^{D_p\times D}$ and $\mathbf{b}_P \in \mathbb{R}^{D_p\times 1}$ are learnable parameters.

It should be noted that in practice, we do not aggregate the information from all the metapath-based neighbors in $\mathcal{N}_i^P$, but we set a threshold N. Specifically, when the number of metapath-based neighbors exceeds the predefined threshold, we unrepeatably select N metapath-based neighbors as $\mathcal{N}_i^P$.

Given a metapath set $\{P_1, P_2, \dots, P_t\}$, following the above steps, we can obtain the node embeddings $\{\mathbf{H}^{P_1}, \mathbf{H}^{P_2}, \dots, \mathbf{H}^{P_t}\}$ under different metapaths. Since different metapaths may reveal diverse semantics, we propose a path-level attention network to fuse these semantic information to learn final node embeddings.

$$\mathbf{z}_i = \sum_{p=1}^{P} \beta_p \cdot \mathbf{h}_i^p, \tag{5}$$

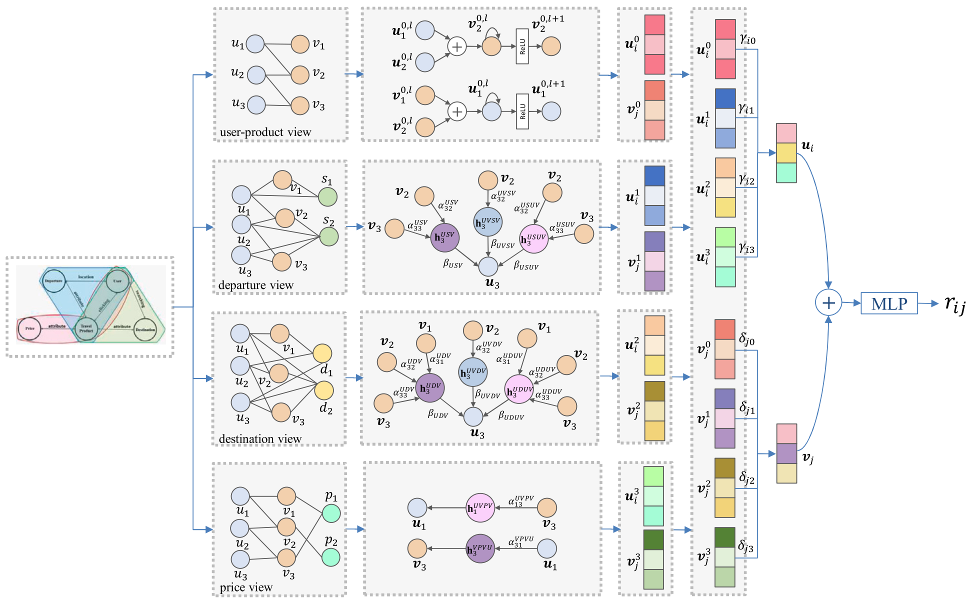


Fig. 4. Illustration of our proposed MV-GAN model in the travel recommendation.

where $\mathbf{z}_i$ is the final embedding of node o_i, $\mathbf{h}_i^p \in \mathbb{R}^{D\times 1}$ is the embedding of node o_i for metapath p, β_p represents the path-level attention weight.

$$\beta_p = \frac{1}{|\mathcal{O}|}\sum_{i\in\mathcal{O}} \mathbf{q}^\top tanh(\mathbf{W}_q\mathbf{h}_i^p + \mathbf{b}_q),$$
$$\beta_p = \frac{exp(\beta_p)}{\sum_{k=1}^{P} exp(\beta_k)}, \tag{6}$$

where $\mathbf{q} \in \mathbb{R}^{D_q\times 1}$, $\mathbf{W}_q \in \mathbb{R}^{D_q\times D}$ and $\mathbf{b}_q \in \mathbb{R}^{D_q\times 1}$ are learnable parameters.

Following the above steps, we can obtain the attribute embeddings of users (travel products) $\mathbf{u}_i^1(\mathbf{v}_j^1)$, $\mathbf{u}_i^2(\mathbf{v}_j^2)$ and $\mathbf{u}_i^3(\mathbf{v}_j^3)$ learned from departure view, destination view, and price view, respectively.

4.3. Embedding fusion layer

Since the importance of different views of each node differs dramatically, we design view-level attention networks to integrate the node embeddings learned from four views to get the global representations of user u_i and product v_j. The higher weights indicate that the corresponding views have a significant impact on the node representations.

The same travel product may in varying degree attract different users to click the travel product. For example, if User-1 wants to take an island trip but User-2 pays more attention on financial cost. User-1 may be attracted by destination Maldives that contains many islands while discount travel products may attract User-2. The view-level attention network is designed to model the personalized travel product representations as follows.

$$\mathbf{v}_j = \sum_{k=0}^{3} \delta_{jk}\mathbf{v}_j^k, \tag{7}$$

where $\mathbf{v}_j \in \mathbb{R}^{D\times 1}$ is the global product representation, δ_{jk} is the weight learned by the view-level attention network.

$$\delta_{jk} = \mathbf{X}_D^\top tanh(\mathbf{V}_D\mathbf{v}_j^k + \mathbf{W}_D\mathbf{u'}_i + \mathbf{b}_D),$$
$$\delta_{jk} = \frac{exp(\delta_{jk})}{\sum_{l=1}^{4} exp(\delta_{jl})}, \tag{8}$$

where $\mathbf{X}_D \in \mathbb{R}^{D_d\times 1}$, $\mathbf{V}_D \in \mathbb{R}^{D_d\times D}$, $\mathbf{W}_D \in \mathbb{R}^{D_d\times D}$ and $\mathbf{b}_D \in \mathbb{R}^{D_d\times 1}$ are learnable parameter. $\mathbf{u'}_i \in \mathbb{R}^{D\times 1}$ is user ID embedding.

Similarly, the same user has different concerns for different travel products. For instance, for domestic tourism products, users pay more attention to the departure city, while for the outbound tourism product, due to travel time constraints, users may pay less attention to the departure city. We also utilize the similar view-level attention mechanism to learn global user representation for different travel products as follows.

$$\mathbf{u}_i = \sum_{k=0}^{3} \gamma_{ik}\mathbf{u}_i^k, \tag{9}$$

where $\mathbf{u}_i \in \mathbb{R}^{D\times 1}$ is global user representation for user u_i. γ_{ik} is the weight learned by the view-level attention network.

$$\gamma_{ik} = \mathbf{X}_G^\top tanh(\mathbf{V}_G\mathbf{u}_i^k + \mathbf{W}_G\mathbf{v'}_j + \mathbf{b}_G),$$
$$\gamma_{ik} = \frac{exp(\gamma_{ik})}{\sum_{l=1}^{4} exp(\gamma_{il})}, \tag{10}$$

where $\mathbf{X}_G \in \mathbb{R}^{D_g\times 1}$, $\mathbf{V}_G \in \mathbb{R}^{D_g\times D}$, $\mathbf{W}_G \in \mathbb{R}^{D_g\times D}$ and $\mathbf{b}_G \in \mathbb{R}^{D_g\times 1}$ are learnable parameters. $\mathbf{v'}_j \in \mathbb{R}^{D\times 1}$ is travel product ID embedding.

4.4. Predictor

After obtaining the global represents $\mathbf{u}_i$ and $\mathbf{v}_j$ for user u_i and product v_j, we apply the element-wise product on their representations and concatenate them with the original representation:

$$\mathbf{e} = \begin{bmatrix} \mathbf{u}_i \odot \mathbf{v}_j \\ \mathbf{u}_i \\ \mathbf{v}_j \end{bmatrix} \tag{11}$$

where $\mathbf{u}_i \odot \mathbf{v}_j$ models the interactive relation between user and product. The concatenation of the original representations $\mathbf{u}_i$ and $\mathbf{v}_j$ is to prevent information loss.

Finally, the output $\mathbf{e}$ is transformed to a preference score via:

$$r_{ij} = \sigma(MLP(\mathbf{e})). \tag{12}$$

Following Fan et al. (2019), we treat the preference score as a binary classification problem and apply the binary cross-entropy loss to optimize the model parameters.

$$\mathcal{L} = -\sum_{u_i\in\mathcal{U}}\sum_{\mathcal{R}_i^+\cup\mathcal{R}_i^-} r_{ij}log\hat{r}_{ij} + (1-r_{ij})log(1-\hat{r}_{ij}), \tag{13}$$

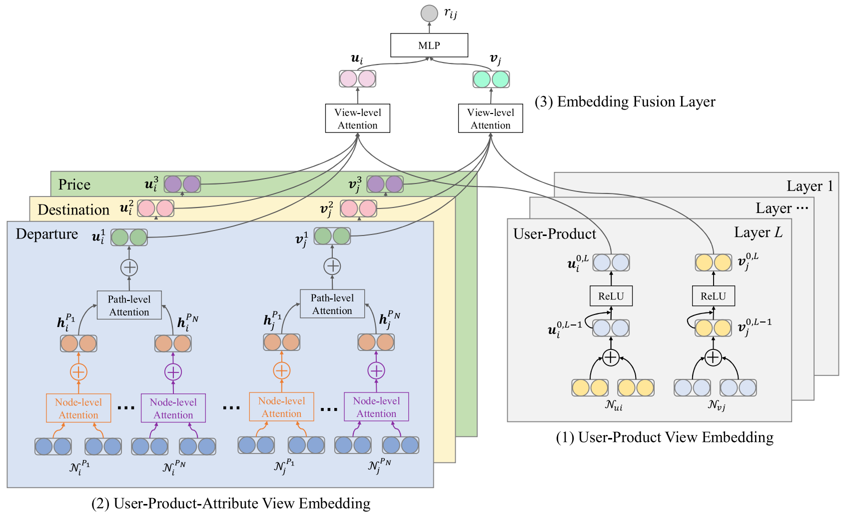


Fig. 5. The overall framework of the proposed model in the travel recommendation.

Algorithm 1: The overall process of MV-GAN

Input : User–product graph $\mathcal{G}_0$,
Departure-view graph $\mathcal{G}_1$,
Destination-view graph $\mathcal{G}_2$,
Price-view graph $\mathcal{G}_3$,
Metapath set $\mathcal{P}_k = \{P_1^k, P_2^k, \cdots, P_{|\mathcal{P}_k|}^k\}$,
Number of layers L.
Output: Preference score r_{ij}.

Random initialize model parameters;
for $l \leftarrow 0$ ***to*** L **do**
  $\mathbf{u}_i^{0,l+1} \leftarrow \mathbf{u}_i^{0,l} \cup \mathcal{N}_{ui}$ (Eq. (1));
  $\mathbf{v}_i^{0,l+1} \leftarrow \mathbf{v}_j^{0,l} \cup \mathcal{N}_{vj}$ (Eq. (2));
end
for $k \leftarrow 1$ ***to*** 3 **do**
  for $\mathcal{P}_k \in \{P_1^k, P_2^k, \cdots, P_{|\mathcal{P}_k|}^k\}$ **do**
    for $i \in \mathcal{O}$ **do**
      Sample the metapath-guided neighbors $\mathcal{N}_i^P$;
      for $j \in \mathcal{N}_i^P$ **do**
        Calculate the weighting coefficient α_{ij}^P (Eq. (4));
      end
      Calculate the semantic-specific node embedding $\mathbf{h}_i^P$ (Eq. (3));
    end
    Calculate the weight of metapath β_p (Eq. (6));
    Obtain the attribute embeddings $\mathbf{u}_i^1$, $\mathbf{u}_i^2$, $\mathbf{u}_i^3$, $\mathbf{v}_j^1$, $\mathbf{v}_j^2$ and $\mathbf{v}_j^3$ (Eq. (5));
  end
end
Calculate the view-level weights δ_j and γ_i (Eqs. (8) and (10));
Obtain the global representations of product $\mathbf{v}_j$ and user $\mathbf{u}_i$ (Eqs. (7) and (9));
Predict preference score r_{ij} via a neural network (Eqs. (11) and (12)).
return r_{ij}.

where R_i^+ denotes the set of products clicked by u_i and R_i^- is negative samples sampled from unclicked products. The overall process of MV-GAN is shown in Algorithm 1.

4.5. Complexity analysis

As shown in Fig. 5, MV-GAN contains three major parts: (1) User–product view embedding. For the lth propagation layer, the computational complexity is $O(|\mathrm{R}^+|D^2)$, where $|\mathrm{R}^+|$ denotes the number of interactions between users and products and D is the embedding size. (2) User–product-attribute view embedding. The complexity of the node-level attention and the path-level attention are $O((|U| + |V|)D_pDN)$ and $O((|U| + |V|)D_qD|P|)$, where $|U|$ and $|V|$ are the number of users and products, D_p and D_q are the row of parameters, N is the maximum number of metapath-based neighbors and $|P|$ is the number of metapaths. (3) Embedding fusion layer. The complexity of the view attention networks for users and products are $O(|U|D_gD)$ and $O(|V|D_dD)$, where D_g and D_d are the row of parameters. Hence, the total computational complexity of MV-GAN is $O(|\mathrm{R}^+|D^2 + (|U| + |V|)D_pDN + (|U|+|V|)D_qD|P| + |U|D_gD + |V|D_dD)$. Since D, D_p, D_q, D_g, D_d and N are all relatively small numbers, $|U|$ and $|V|$ are very small to the number of the user–product interactions $|\mathrm{R}^+|$, the overall complexity of MV-GAN is linear with the number of the user–product interactions.

5. Experiments

In this section, we first describe the datasets, evaluation metrics, baseline methods, followed by the experimental results answering the following three research questions:

- **RQ1:** How does the proposed MV-GAN method compare with the state-of-the-art recommendation approaches?
- **RQ2:** How do different components in MV-GAN contribute to the recommendation performance?
- **RQ3:** What is the impact of different hyper-parameters tuning on the performance of MV-GAN?

5.1. Experimental settings

Datasets. Since our proposed method MV-GAN is not only suitable for travel product recommendations, it can also be extended or simplified to other recommendation scenarios, we experimented with two types of real-world datasets: Tuniu (Chen, Wu et al., 2020) dataset for the travel product recommendation and MovieLens[2] dataset for the movie

[2] https://grouplens.org/datasets/hetrec-2011/.

Table 1
Description of datasets.

Datasets	# Users	# Items	# User–item interactions	# Departures	# Destinations	# Prices	# Actors	# Directors	# Genres
Tuniu_D_1	25,005	22,802	258,566	30	1036	11	–	–	–
Tuniu_D_2	27,947	25,948	298,454	30	1433	11	–	–	–
MovieLens	2113	10,197	855,598	–	–	–	95,321	4060	20

Note: "#" indicates the number of objects.

Table 2
Metapaths designed for datasets.

Datasets	Views	Metapaths	Semantic meaning
Tuniu	Departure (S)	USV	A user may click a travel product whose departure city is near the place of residence
		USUV	A user may click a travel product that other users have clicked with the same departure
		UVSV	A user may click a travel product with the same departure as the travel product he has clicked
		VSU	A travel product may be clicked by a user whose place of residence is near the departure city
		VUSU	A travel product may be clicked by other users with the same departure
		VSVU	A travel product may be clicked by a user that clicked travel products with the same departure
	Destination (D)	UDV	A user may click a travel product whose destination city he has searched
		UDUV	A user may click a travel product that other users have clicked with the same searched destination
		UVDV	A user may click a travel product with the same destination as the travel product he has clicked
		VDU	A travel product may be clicked by a user who has searched its destination
		VUDU	A travel product may be clicked by other users with the same searched destination
		VDVU	A travel product may be clicked by a user that clicked travel products with the same destination
	Price (P)	UVPV	A user may click a travel product with the same price range as the travel product he has clicked
		VPVU	A travel product may be clicked by a user that clicked travel products with the same price range
MovieLens	Actor (A)	UMAM	A user may like a movie with the same actor as the movie he has rated
		MAMU	A movie may be watched by a user that rated movie with the same actor
	Director (R)	UMRM	A user may like a movie with the same director as the movie he has rated
		MRMU	A movie may be watched by a user that rated movie with the same director
	Genre (G)	UMGM	A user may like a movie with the same genre as the movie he has rated
		MGMU	A movie may be watched by a user that rated movie with the same genre

recommendation. Detailed statistics of all datasets can be found in Table 1. Meanwhile, we employ the metapaths that start from users (items) and end to items (users) to perform experiments. Table 2 plots the selected metapaths in the datasets. In order to simulate real-world recommendations, for both datasets, we use the first 70% of user–item interactions as the training set and split the remaining data equally into a validation and test set.

Tuniu Dataset. The dataset is provided by Tuniu,[3] one of the most popular tourism e-commerce companies in China. In particular, we collect user click logs from two periods of time, the first period is the range from July 1st to July 31st in 2013 (*i.e.,* Tuniu_D_1). The second time period is from August 1st to August 31st in 2013 (*i.e.,* Tuniu_D_2). Then we randomly select 0.2% of users and their clicked travel products from each dataset. For each travel product, they have some side information, such as departure, destination and price. For each user, they have IP address and searched keywords. The IP address can be used to infer the nearest departure city and the searched keywords often involve the intended destination. Finally, we can obtain the preprocessed dataset, which contains five types of nodes, *i.e.,* user (U), travel product (V), departure (S), destination (D), price (P) and six types of edges, *i.e.,* user–product (click), User–Departure (near), User–Destination (search for), Product–Departure (belong to), Product–Destination (belong to), Product–Price (belong to).

MovieLens Dataset. MovieLens is a publicly available dataset, which is widely used to evaluate numerous relevant methods. Though there exists only one metapath for user (item) node in different views and the path-level attention mechanism does not work, we have chosen this dataset, because to the best of our knowledge, there is no dataset except Tuniu that fits the MV-GAN model perfectly. Thus, we simplify the MV-GAN model by removing the path-level attention network to apply it to the MovieLens dataset, and extract a subset from this dataset, which contains five types of nodes, *i.e.,* user (U), movie (M), actor(A), director (D), genre (G) and four types of edges, *i.e.,* user–movie (rate), actor–movie (act in), director–movie (direct), movie-genre (belong to). Following the prior study (Wu et al., 2020), we transform it into implicit data, where each user–movie pair has an edge or not indicating whether the user has rated the movie or not.

Evaluation Scheme. To evaluate the effectiveness of the proposed MV-GAN, we adopt two types of metrics, Recall@k and Normalized Discounted Cumulative Gain (NDCG@k). We denote the top-k list of items recommended for user u as R_u, and the list of items that user u actually clicked as G_u. The specific calculation process is as follows.

Recall@k measures the proportion of the cases that have correctly recommended items among the top-k items in all test cases. It is defined as follows:

$$\text{Recall@}k = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \mathbb{I}(R_{u,g_u} \le k), \tag{14}$$

where g_u is the clicked item in test set for user u, R_{u,g_u} is the generated rank for item g_u and user u, and $\mathbb{I}$ is an indicator function.

NDCG is the normalized position-discounted precision score:

$$\text{NDCG@}k = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\sum_{j=1}^{k} \frac{2^{\mathbb{I}(R_u(j)\in G_u)}-1}{\log_2^{1+j}}, \tag{15}$$

where $R_u(j)$ represents the item recommended in the jth position, the indicator function $\mathbb{I}(\cdot) = 1$ if $R_u(j) \in G_u$, otherwise for 0. In this experiment, we set $k = 5, 10, 15, 20$, and summarize the experimental results of Recall@k and NDCG@k.

Baselines. To evaluate the performance of travel recommendation, we compare MV-GAN with four traditional recommendation methods (*i.e.,* POP, Item-KNN, BPR-MF, and FM), and five widely-applied neural-based recommendation methods (*i.e.,* DeepFM, SR-GNN, NGCF, LightGCN, and IGMC).

[3] http://www.tuniu.com.

Table 3
Performance comparisons of travel recommendation on dataset Tuniu_D_1.

Methods	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@15	NDCG@15	Recall@20	NDCG@20
POP	0.0143	0.0066	0.0256	0.0081	0.0336	0.0087	0.0421	0.0092
Item-KNN	0.0933	0.0857	0.1405	0.1010	0.1731	0.1076	0.1969	0.1111
BPR-MF	0.0724	0.0912	0.0958	0.0931	0.1110	0.0927	0.1231	0.0917
FM	0.1581	0.1140	0.1766	0.1253	0.2104	0.1473	0.2405	0.1661
DeepFM	0.1664	0.1263	0.1859	0.1370	0.2214	0.1595	0.2512	0.1743
SR-GNN	0.1731	0.1375	0.2048	0.1488	0.2449	0.1676	0.2692	0.1771
NGCF	0.1615	0.1177	0.1863	0.1297	0.2139	0.1488	0.2352	0.1663
LightGCN	0.1696	0.1238	0.1928	0.1358	0.2246	0.1562	0.2484	0.1709
IGMC	0.1608	0.1148	0.1900	0.1303	0.2086	0.1518	0.2386	0.1688
MV-GAN	**0.1892**	**0.1515**	**0.2250**	**0.1611**	**0.2685**	**0.1890**	**0.2967**	**0.1960**

Table 4
Performance comparisons of travel recommendation on dataset Tuniu_D_2.

Methods	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@15	NDCG@15	Recall@20	NDCG@20
POP	0.0156	0.0079	0.0263	0.0092	0.0363	0.0100	0.0456	0.0105
Item-KNN	0.0928	0.0862	0.1376	0.1011	0.1687	0.1075	0.1910	0.1109
BPR-MF	0.0750	0.0946	0.0992	0.0963	0.1141	0.0958	0.1261	0.0950
FM	0.1606	0.1160	0.1774	0.1281	0.2146	0.1519	0.2385	0.1709
DeepFM	0.1668	0.1297	0.1868	0.1409	0.2277	0.1643	0.2501	0.1789
SR-GNN	0.1782	0.1408	0.2119	0.1527	0.2512	0.1718	0.2755	0.1814
NGCF	0.1654	0.1240	0.1893	0.1324	0.2259	0.1568	0.2557	0.1736
LightGCN	0.1715	0.1326	0.1963	0.1418	0.2355	0.1646	0.2628	0.1789
IGMC	0.1647	0.1254	0.1879	0.1310	0.2302	0.1594	0.2657	0.1789
MV-GAN	**0.1919**	**0.1562**	**0.2304**	**0.1705**	**0.2738**	**0.1902**	**0.3003**	**0.2040**

- **POP**. POP simply recommends the popular items based on the user–item interaction (click or rate) frequency in the training data.
- **Item-KNN** (Sarwar et al., 2001). Item-KNN selects the similar items according to the cosine similarity.
- **BPR-MF** (Rendle et al., 2012). BPR-MF models the order of candidate items by a pairwise ranking loss. It adopts a stochastic gradient descent framework that contrasts pairs of positive and negative samples.
- **FM** (Rendle, 2010). FM applies a sum of pairwise inner products of the user or item features to get the prediction score. In Tuniu datasets, we regard the departure, destination, and price as item-specific features, and nearest departure and intended destination as user-specific features. In the MovieLens dataset, we integrate actor, director and genre into FM by regarding them as item-specific features.
- **DeepFM** (Guo et al., 2017). DeepMF combines FM and deep neural networks to learn the low- and high-order feature interactions. Similar to FM, we use the same user-specific features and item-specific features.
- **SR-GNN** (Wu et al., 2019). SR-GNN is a GNN-based recommendation model that models the session data in the graph structure, and obtains the session representation via an attention mechanism.
- **NGCF** (Wang, He et al., 2019). NGCF is a GNN-based recommendation approach that learns the collaborative signal by performing embedding propagation on a bipartite user–item graph.
- **LightGCN** (He et al., 2020). LightGCN simplifies NGCF by removing the feature transformation and nonlinear activation module since these two components contribute little to the recommendation performance.
- **IGMC** (Zhang & Chen, 2019). IGMC is an inductive graph neural network model that extracts a subgraph for each user–item pair, and leverages GNN to train a regression model that maps the subgraph structure to its corresponding ratings.

Parameters Settings. For all comparison methods, we carry out experiments under the optimal parameter setting. For the proposed MV-GAN, we use the Xavier initializer to initialize parameters and optimize the model with Adam. The embedding size is fixed to 64. The batch size is searched in [64, 128, 256, 512], and 256 is selected for Tuniu_D_1 and Tuniu_D_2, while 128 is better for the MovieLens dataset. The learning rate is tuned amongst [0.0001, 0.0005, 0.001, 0.005], and the validation results show that 0.0005 is better for Tuniu_D_1 and Tuniu_D_2, and 0.001 is better for MovieLens. We apply the dropout strategy to avoid overfitting and the dropout rate is set to 0.2 for all datasets. In the user–product bipartite graph, we set the number of layers L to 2. The proposed MV-GAN and all the compared neural-based models are defined and trained on a Windows server with 3.60 GHz Intel I9-9900k CPU and 11 GB Nvidia GeForce RTX 2080 Ti GPU, and implemented in Pytorch.[4]

5.2. Overall performance comparison (RQ1)

This section analyzes the results of MV-GAN and the state-of-the-art recommendation approaches. The results on two datasets concerning Recall and NDCG are shown in Tables 3–5, where the best performer on each column is highlighted in bold type. The observations are outlined as follows:

(1) Our proposed method MV-GAN achieves the best performance in terms of Recall and NDCG. Specifically, MV-GAN outperforms the best baseline by 9.3%–12.7%, 7.7%–12.4% and 1.0%–7.4% on the datasets Tuniu_D_1, Tuniu_D_2 and MovieLens.
(2) Regarding the traditional methods, such as POP and Item-KNN, their performance is relatively poor. Such simple approaches make recommendations solely according to repetitive cooccurred items or item similarity, which is not suitable for solving recommendation problems with sparse data, such as travel recommendation scenarios.
(3) DeepFM achieves higher Recall and NDCG than FM, suggesting that the necessity of using sophisticated methods (*e.g.,* neural network) to model feature interactions for better predictions.

[4] https://pytorch.org/.

Table 5
Performance comparisons of travel recommendation on dataset MovieLens.

Methods	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@15	NDCG@15	Recall@20	NDCG@20
POP	0.0492	0.1718	0.0875	0.2120	0.1148	0.2382	0.1367	0.2566
Item-KNN	0.0745	0.2331	0.1304	0.2799	0.1706	0.2974	0.1937	0.3185
BPR-MF	0.0981	0.2893	0.1518	0.3291	0.1977	0.3767	0.2263	0.3906
FM	0.1228	0.3080	0.1689	0.3623	0.2305	0.4076	0.2597	0.4352
DeepFM	0.1342	0.3161	0.1782	0.3675	0.2410	0.4155	0.2672	0.4461
SR-GNN	0.1562	0.3286	0.2093	0.3948	0.2603	0.4356	0.3228	0.4717
NGCF	0.1424	0.3044	0.1857	0.3679	0.2454	0.3914	0.2856	0.4445
LightGCN	0.1474	0.3077	0.1975	0.3723	0.2535	0.3968	0.3086	0.4475
IGMC	0.1302	0.2755	0.1673	0.3345	0.2248	0.3511	0.2609	0.4165
MV-GAN	**0.1592**	**0.3397**	**0.2248**	**0.4063**	**0.2630**	**0.4496**	**0.3307**	**0.4783**

Table 6
Ablation study of MV-GAN on dataset Tuniu_D_1.

Methods	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@15	NDCG@15	Recall@20	NDCG@20
MV-GAN-w/ N	0.1853	0.1484	0.2141	0.1539	0.2543	0.1824	0.2840	0.1900
MV-GAN-w/ T	0.1811	0.1442	0.2189	0.1576	0.2580	0.1845	0.2888	0.1910
MV-GAN-w/ V	0.1833	0.1452	0.2157	0.1554	0.2561	0.1839	0.2804	0.1877
MV-GAN-w/ N,T,V	0.1776	0.1438	0.2086	0.1525	0.2525	0.1754	0.2763	0.1802
MV-GAN-w/o S	0.1849	0.1471	0.2175	0.1575	0.2581	0.1770	0.2808	0.1888
MV-GAN-w/o D	0.1799	0.1421	0.2091	0.1530	0.2523	0.1731	0.2735	0.1823
MV-GAN-w/o P	0.1814	0.1434	0.2122	0.1544	0.2542	0.1740	0.2768	0.1858
MV-GAN	**0.1892**	**0.1515**	**0.2250**	**0.1611**	**0.2685**	**0.1890**	**0.2967**	**0.1960**

Note: (1) "w/o" indicates removing corresponding components from MV-GAN;
(2) "w/" denotes replacing some components of MV-GAN with alternative ways.

(4) Although NGCF, LightGCN, and IGMC can learn the high-order interactions between users and items, they ignore the side information of users and items. Thus, the performances of NGCF, LightGCN, and IGMC are still worse than MV-GAN.

Case Studies on Travel Recommendation. To better understand what can be recommended by the MV-GAN model, we show two examples in Figs. 6 and 7, one of which contains searched destination while the other does not. From these figures, we can make the following observations: (1) MV-GAN can capture users' intended destinations. For instance, in Example 1, although the user does not search for any destination, MV-GAN can learn the user's intention of the destination from his/her historical record, and recommend four travel products about Maldives to the user in the top 5 recommendation list. (2) MV-GAN can capture users' intended departure cities. In Example 1, the user's IP address is Beijing, and due to the price reason, he/she also chooses the nearest city Tianjin as the departure. MV-GAN can learn that the user's main intended departure city is Beijing from the user's IP address and the clicked travel products in the train set. Similarly, in Example 2, MV-GAN can judge that the user's intended departure city is Hangzhou. (3) MV-GAN considers the price factor of travel products. In the recommended list, it recommends travel products with the same or adjacent price range as the travel products that he/she clicked in the train set. (4) Accidental clicks in the train set can be filtered by the MV-GAN model. For instance, in Example 2, TRKG only recommends travel products relative to Xiamen in the top 5 of the recommendation list and ignores accidental clicks relative to Yunnan and Mount Wuyi.

Fig. 6. Example 1 of input and output of MV-GAN. We mark the ground truth with a red rectangular box.

5.3. Ablation study (RQ2)

To further validate the effectiveness of each component of the proposed MV-GAN, we conduct some ablation studies. Since the simplified version of the MV-GAN method is used on the MovieLens dataset, we only perform ablation analysis on the datasets Tuniu_D_1 and Tuniu_D_2. First, we investigate the effect of three type attention mechanisms by replacing node-level attention as the average pooling method, termed MV-GAN-w/ N, replacing path-level attention as the average pooling method, termed MV-GAN-w/ T, replacing view-level attention as the average pooling method, termed MV-GAN-w /V, and removing node-level, path-level and view-level attention, termed MV-GAN-w/ N,T,V. The results are shown in Tables 6 and 7, from which we have the

Table 7
Ablation study of MV-GAN on dataset Tuniu_D_2.

Methods	Recall@5	NDCG@5	Recall@10	NDCG@10	Recall@15	NDCG@15	Recall@20	NDCG@20
MV-GAN-w/ N	0.1867	0.1520	0.2249	0.1657	0.2680	0.1868	0.2864	0.1926
MV-GAN-w/ T	0.1856	0.1497	0. 2271	0.1683	0.2645	0.1836	0.2920	0.1956
MV-GAN-w/ V	0.1836	0.1482	0.2211	0.1638	0.2612	0.1825	0.2902	0.1923
MV-GAN-w/ N,T,V	0.1827	0.1466	0.2169	0.1630	0.2572	0.1772	0.2808	0.1845
MV-GAN-w/o S	0.1885	0.1497	0.2213	0.1631	0.2646	0.1813	0.2872	0.1934
MV-GAN-w/o D	0.1846	0.1452	0.2165	0.1582	0.2583	0.1762	0.2818	0.1876
MV-GAN-w/o P	0.1867	0.1467	0.2187	0.1602	0.2619	0.1888	0.2846	0.1908
MV-GAN	**0.1919**	**0.1562**	**0.2304**	**0.1705**	**0.2738**	**0.1902**	**0.3003**	**0.2040**

Note: (1) "w/o" indicates removing corresponding components from MV-GAN;
(2) "w/" denotes replacing some components of MV-GAN with alternative ways.

Fig. 7. Example 2 of input and output of MV-GAN. We mark the ground truth with a red rectangular box.

following observations: (1) Removing node-level, path-level or view-level attention networks yields a drop of performance on both datasets. It indicates that these three attention mechanisms are useful for enhancing the representations of users and items, and improving the recommendation performance. (2) We can observe significant drops for both metrics across both datasets when removing all attention mechanisms.

Then, we present another ablation analysis to study which graph contributes more to the overall travel recommendation performance. In this work, we exploit three types of graphs (*i.e.,* the user–product–departure graph, the user–product–destination graph, and the user–product–price graph) to construct graph structured information among users, products, and attributes. We remove these graphs and term them as MV-GAN-w/o S, MV-GAN-w/o D, and MV-GAN-w/o P, respectively.

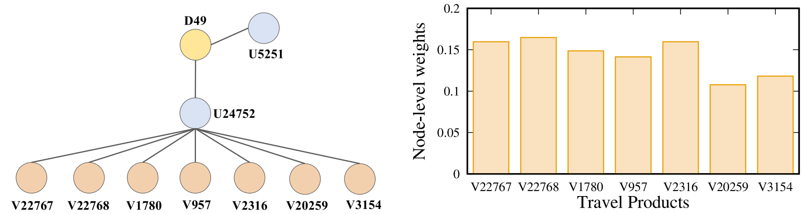
Table 8
A example of a sampled travel product V606 on the effect of view-level attention.

User	Model	Interaction	Departure	Destination	Price	$\hat{R}$
U128	MV-GAN-w/V	0.250	0.250	0.250	0.250	0.3443
	MV-GAN	0.264	**0.362**	0.193	0.181	**0.5599**
U407	MV-GAN-w/V	0.250	0.250	0.250	0.250	0.4231
	MV-GAN	0.264	0.185	**0.291**	0.260	**0.6967**
U649	MV-GAN-w/V	0.250	0.250	0.250	0.250	0.4248
	MV-GAN	0.235	0.226	0.232	**0.307**	**0.5701**

From the results in Tables 6 and 7, we obtain the following lessons: (1) All these types of graph information improve the preferences of MV-GAN, which shows the effectiveness of our proposed GNN-based travel recommendation method and demonstrates that these three kinds of graph information provide effective information to enrich the representation of users and products. (2) Comparing these three types of graph information, the influence factors in order to sort as follows: destination, price and departure, indicating that users are more concerned with destination and financial cost but relatively care less about departure, when choosing travel products. We argue that this is mainly because that some departures do not have relevant destination routes or price factors, tourists may also choose a departure that is farther away. Nevertheless, the financial cost is still an important factor that tourists care about.

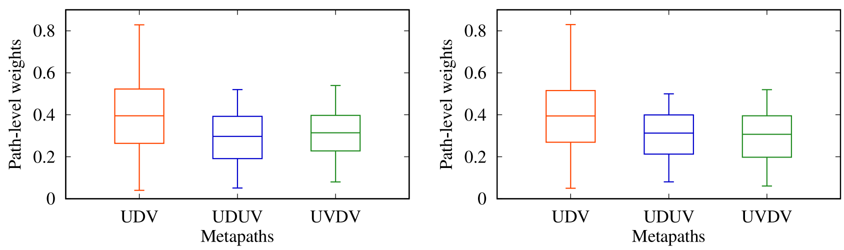
Case Studies. MV-GAN can learn the attention weights between nodes and their neighbors under a metapath. Some meaningful neighbors that are useful for node representation tend to have larger attention weights. To better understand the node-level attention weights learning, we take the metapath UDUV (users with the same searched destination have similar interest preferences) as an illustrative example, and randomly select a user (U5251) and its metapath-based neighbors (V22767, V22768, V1780, V957, V2316, V20259 and V3154). Please note that only users U5251 and U24752 searched destination D49. The attention weights are shown in Fig. 8, we can observe that travel product nodes V22767, V22768, V1780, V957 and V2316 get higher attention weights than V20259 and V3154, which means these nodes make more important contributions to learn user representation. We argue that this is mainly because these nodes' destination is D49 (Xishuangbanna) which is searched by user U5251, while the other two nodes' destination is D142 (Qinghai).

To deeply investigate the path-level attention weights, we present the macro-level analysis by a boxplot graph of path-level attention distributions on datasets Tuniu_D_1 and Tuniu_D_2. The results are shown in Fig. 9, from which we can see that the attention distributions of metapaths are indeed different. Metapath UDV gets the largest average path-level attention, indicating this metapath is more important to consider than the others. This is probably because metapath UDV is more related to users' intended destinations which reflects the strongest user interest preference. These attention weights can provide potential interpretability for recommendation results.



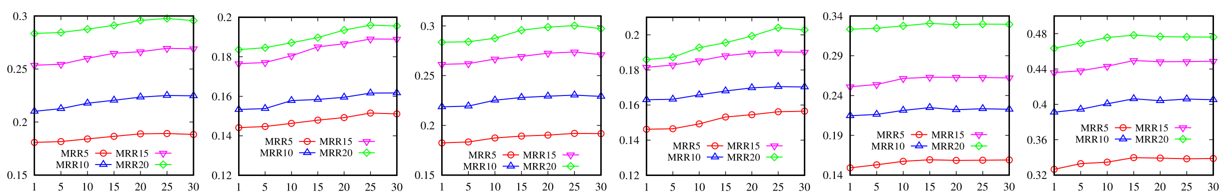
(a) Metapath-based neighbors of U5251 (b) Attention values of U5251's neighbors

Fig. 8. Metapath-based neighbors of user U5251 and corresponding node-level attention values.



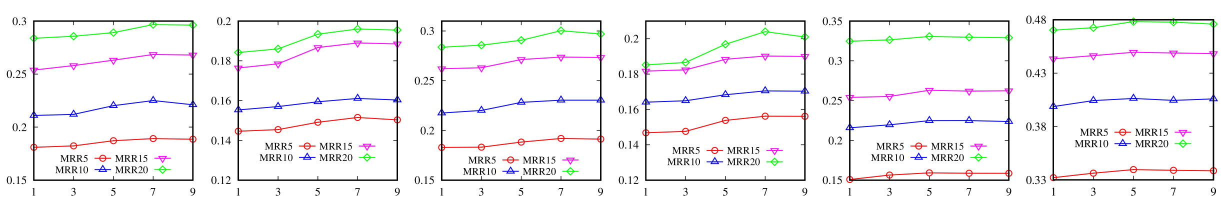
(a) Tuniu_D_1 (b) Tuniu_D_2

Fig. 9. Path-level attention distributions.



(a) Tuniu_D_1:Recall (b) Tuniu_D_1:NDCG (c) Tuniu_D_2:Recall (d) Tuniu_D_2:NDCG (e) MovieLens:Recall (f) MovieLens:NDCG

Fig. 10. Impact of the number of metapath-based neighbors N.



(a) Tuniu_D_1:Recall (b) Tuniu_D_1:NDCG (c) Tuniu_D_2:Recall (d) Tuniu_D_2:NDCG (e) MovieLens:Recall (f) MovieLens:NDCG

Fig. 11. Impact of the number of negative samples ρ.

To study the effect of view-level attention, we randomly select a travel product (V606) and three testing users (U128, U407 and U649) that have clicked this travel product. Each user has the clicking history[5] (*i.e.,* user–product interactions). Experimental results are presented in Table 8. From the results, we have the following observations. For different users, the view-level attention weights with respect to different views vary significantly in MV-GAN. For user U128, the view-level attention weight of departure is relatively high. It is reasonable because U128 has clicked many travel products with the same departure but different destinations and price ranges (V601, V602, V77, V221, V603, V604, V605, V549 and V551 have same departure city Nanjing, but their destination includes West Zhejiang, Qinghai, Rizhao, Guilin, Xiangxi, Huangshan and Zhangjiajie, and their prices cover three ranges). Meanwhile, for user U407, the destination factor gets the largest view-level attention value. This is because U407 has clicked many travel products with same destination Huangshan. For user U649, the price factor gets the highest attention value from view-level attention, indicating U649 pays more attention on price ranges. We argue that this is because the prices of travel products that U649 has clicked are within 2000 RMB. The prediction scores of MV-GAN are much better than that of MV-GAN-w/ V. The result demonstrates the effectiveness of fusing the view-level attention network into MV-GAN.

5.4. Parameter analysis of MV-GAN (RQ3)

Since the maximum number of metapath-based neighbors N plays an important role in MV-GAN, we investigate its influence on the

[5] U128 has clicked V601, V602, V77, V221, V603, V604, V605, V549 and V551; U407 has clicked V1675, V1676, V1677, V1678, V1679, V1680, V18795, V1564 and V941; U649 has clicked V1866, V2514, V2515, V2516, V6463 and V8866.

recommendation performance, and then we evaluate the impact of the number of negative samples.

Number of Metapath-based Neighbors N. In the experiments, we sample a fixed number of metapath-based neighbors to obtain node embedding instead of aggregating the information from all the metapath-based neighbors. To analyze the influence of the maximum number of metapath-based neighbors N, we vary N in the range of $\{1, 5, 10, 15, 25, 30\}$. The performance results in terms of Recall and NDCG metrics on both datasets are plotted in Fig. 10. We can see that increasing metapath-based neighbor samples can improve the performance, because metapath-based neighbors provide more information about representations of user and product, and the results of travel recommendation are more accurate. When metapath-based neighbor samples increase again, the recommendation performance has lesser improvements or even decreases. This is probably because node embeddings may be impaired by weakly correlated metapath-based neighbors. Considering both recommendation performance and training efficiency, we set N to 25, 25 and 15 for datasets Tuniu_D_1, Tuniu_D_2 and MovieLens.

Number of Negative Samples ρ. We vary the number of negative samples from 1 to 9 with a step size 2 to study the effect on the performance of the travel recommendation. The performance for travel recommendation on datasets Tuniu_D_1, Tuniu_D_2 and MovieLens are presented in Fig. 11. From the figure, we can observe that too small or too large number of negative samples are not good for the performance. It is reasonable because few negative samples cannot provide enough information for representations of user and product, while too many negative samples will lead negative samples to achieve a dominant position and may cause low recommendation accuracy. In general, when ρ is set within [5,7], almost the inverted U-shaped curves reach their high point, *i.e.,* the maximal recommendation accuracy.

6. Conclusion and future work

In this article, we propose a novel multi-view graph embedding framework called MV-GAN to solve the travel product recommendation problem. MV-GAN overcomes the data sparsity problem in travel recommendation by aggregating the related context in GNN and fusing the multi-view information. Specifically, we design a node-level and path-level attention network for learning user and product representations from each individual view, and propose view-level attention networks to promote the collaboration and fusion of different views. Extensive experiments are conducted on real-world datasets, and the results demonstrate that MV-GAN outperforms the state-of-the-art methods on travel recommendation tasks. By analyzing the node-level, path-level and view-level attention values, MV-GAN method has shown its potentially good interpretability.

In the future, we will extend our work in the following aspects. First, we plan to consider multiple modalities of information (*e.g.,* textual and image information) for further accuracy of recommendation boost. Second, we intend to incorporate more complex graph representation learning techniques that can potentially lead to better user and product representations.

CRediT authorship contribution statement

Lei Chen: Conceptualization, Data curation, Software, Writing – original draft. **Jie Cao:** Resources, Project administration, Funding acquisition. **Youquan Wang:** Conceptualization, Methodology, Validation, Writing – review & editing. **Weichao Liang:** Methodology, Formal analysis, Supervision, Writing – review & editing. **Guixiang Zhu:** Visualization, Investigation.

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.

Acknowledgments

This research is partially supported by the National Key Research and Development Program of China under grant 2017YFD0401001, in part by the National Natural Science Foundation of China under grant 92046026, 72172057 and 71701089, in part by the Jiangsu Provincial Key Research and Development Program, China under grant BE2020001-3, in part by the International Innovation Cooperation Project of Jiangsu Province, China under Grant BZ2020008.

References

Cao, Y., Peng, H., Wu, J., Dou, Y., Li, J., & Yu, P. S. (2021). Knowledge-preserving incremental social event detection via heterogeneous GNNs. In *Proceedings of the web conference 2021* (pp. 3383–3395).

Chen, L., Cao, J., Chen, H., Liang, W., Tao, H., & Zhu, G. (2021). Attentive multi-task learning for group itinerary recommendation. *Knowledge and Information Systems*, *63*, 1687–1716.

Chen, L., Cao, J., Zhu, G., Wang, Y., & Liang, W. (2021). A multi-task learning approach for improving travel recommendation with keywords generation. *Knowledge-Based Systems*, *233*, Article 107521.

Chen, L., Wu, Z., Cao, J., Zhu, G., & Ge, Y. (2020). Travel recommendation via fusing multi-auxiliary information into matrix factorization. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *11*(2), 1–24.

Chen, L., Zhang, L., Cao, S., Wu, Z., & Cao, J. (2020). Personalized itinerary recommendation: Deep and collaborative learning with textual information. *Expert Systems with Applications*, *144*, Article 113070.

Fan, S., Zhu, J., Han, X., Shi, C., Hu, L., Ma, B., & Li, Y. (2019). Metapath-guided heterogeneous graph neural network for intent recommendation. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery and data mining* (pp. 2478–2486).

Ge, Y., Xiong, H., Tuzhilin, A., & Liu, Q. (2014). Cost-aware collaborative filtering for travel tour recommendations. *ACM Transactions on Information Systems*, *32*(1), 1–31.

Guo, H., Tang, R., Ye, Y., Li, Z., & He, X. (2017). Deepfm: A factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th international joint conference on artificial intelligence* (pp. 1725–1731).

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (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).

He, J., Liu, H., & Xiong, H. (2016). Socotraveler: Travel-package recommendations leveraging social influence of different relationship types. *Information and Management*, *53*(8), 934–950.

Jin, B., Gao, C., He, X., Jin, D., & Li, Y. (2020). Multi-behavior recommendation with graph convolutional networks. In *Proceedings of the 43rd international acm sigir conference on research and development in information retrieval* (pp. 659–668).

Kapetanakis, S., Polatidis, N., Alshammari, G., & Petridis, M. (2020). A novel recommendation method based on general matrix factorization and artificial neural networks. *Neural Computing and Applications*, *32*(16), 12327–12334.

Kolahkaj, M., Harounabadi, A., Nikravanshalmani, A., & Chinipardaz, R. (2020). A hybrid context-aware approach for e-tourism package recommendation based on asymmetric similarity measurement and sequential pattern mining. *Electronic Commerce Research and Applications*, *42*, Article 100978.

Liu, Q., Chen, E., Xiong, H., Ge, Y., Li, Z., & Wu, X. (2014). A cocktail approach for travel package recommendation. *IEEE Transactions on Knowledge and Data Engineering*, *26*(2), 278–293.

Liu, F., Xue, S., Wu, J., Zhou, C., Hu, W., Paris, C., Nepal, S., Yang, J., & Yu, P. S. (2020). Deep learning for community detection: Progress, challenges and opportunities. In *Proceedings of the 29th international joint conference on artificial intelligence* (pp. 4981–4987).

Liu, C.-Y., Zhou, C., Wu, J., Hu, Y., & Guo, L. (2018). Social recommendation with an essential preference space. In *Proceedings of the 32th aaai Conference on Artificial Intelligence* (pp. 346–353).

Ma, X., Wu, J., Xue, S., Yang, J., Sheng, Q. Z., & Xiong, H. (2021). A comprehensive survey on graph anomaly detection with deep learning. CoRR abs/2106.07178.

Rendle, S. (2010). Factorization machines. In *IEEE international conference on data mining* (pp. 995–1000). IEEE.

Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2012). *BPR: Bayesian personalized ranking from implicit feedback*. UAI.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on world wide web* (pp. 285–295).

Shi, C., Li, Y., Zhang, J., Sun, Y., & Philip, S. Y. (2016). A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, *29*(1), 17–37.

Sojahrood, Z. B., & Taleai, M. (2021). A POI group recommendation method in location-based social networks based on user influence. *Expert Systems with Applications*, *171*, Article 114593.

Su, X., Xue, S., Liu, F., Wu, J., Yang, J., Zhou, C., Hu, W., Paris, C., Nepal, S., Jin, D., Sheng, Q. Z., & Yu, P. S. (2021). A comprehensive survey on community detection with deep learning. CoRR abs/2105.12584.

Tong, N., Tang, Y., Chen, B., & Xiong, L. (2021). Representation learning using attention network and CNN for heterogeneous networks. *Expert Systems with Applications*, Article 115628.

Wan, G., Du, B., Pan, S., & Haffari, G. (2020). Reinforcement learning based meta-path discovery in large-scale heterogeneous information networks. In *Proceedings of the aaai conference on artificial intelligence, Vol. 34* (pp. 6094–6101).

Wan, M., Wang, D., Goldman, M., Taddy, M., Rao, J., Liu, J., Lymberopoulos, D., & McAuley, J. (2017). Modeling consumer preferences and price sensitivities from large-scale grocery shopping transaction logs. In *Proceedings of the 26th international conference on world wide web* (pp. 1103–1112).

Wang, X., He, X., Wang, M., Feng, F., & Chua, T.-S. (2019). Neural graph collaborative filtering. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval* (pp. 165–174).

Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., & Yu, P. S. (2019). Heterogeneous graph attention network. In *Proceedings of the 28th international conference on world wide web* (pp. 2022–2032).

Wang, X., Wang, R., Shi, C., Song, G., & Li, Q. (2020). Multi-component graph convolutional collaborative filtering. In *Proceedings of the aaai conference on artificial intelligence* (pp. 6267–6274).

Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., & Tan, T. (2019). Session-based recommendation with graph neural networks. In *Proceedings of the aaai conference on artificial intelligence, Vol. 33* (pp. 346–353.

Wu, L., Yang, Y., Zhang, K., Hong, R., Fu, Y., & Wang, M. (2020). Joint item recommendation and attribute inference: An adaptive graph convolutional network approach. In *Proceedings of the 43rd international acm sigir conference on research and development in information retrieval* (pp. 679–688).

Xu, C., Zhao, P., Liu, Y., Sheng, V. S., Xu, J., Zhuang, F., Fang, J., & Zhou, X. (2019). Graph contextualized self-attention network for session-based recommendation. In *Proceedings of the 28th international joint conference on artificial intelligence, Vol. 19* (pp. 3940–3946).

Yang, C., Bai, L., Zhang, C., Yuan, Q., & Han, J. (2017). Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1245–1254).

Zhang, M., & Chen, Y. (2019). Inductive matrix completion based on graph neural networks. In *International conference on learning representations*.

Zhao, W. X., Fan, F., Wen, J.-R., & Chang, E. Y. (2018). Joint representation learning for location-based social networks with multi-grained sequential contexts. *ACM Transactions on Knowledge Discovery from Data*, *12*(2), 1–21.

Zheng, Y., Gao, C., He, X., Li, Y., & Jin, D. (2020). Price-aware recommendation with graph convolutional networks. In *Proceedings of the 36th international conference on data engineering* (pp. 133–144). IEEE.

Zhu, H., Li, X., Zhang, P., Li, G., He, J., Li, H., & Gai, K. (2018). Learning tree-based deep model for recommender systems. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery and data mining* (pp. 1079–1088).

Zhu, G., Wang, Y., Cao, J., Bu, Z., Yang, S., Liang, W., & Liu, J. (2021). Neural attentive travel package recommendation via exploiting long-term and short-term behaviors. *Knowledge-Based Systems*, *211*, Article 106511.