A Novel Group Recommendation Model With Two-Stage Deep Learning

Zhenhua Huang, Yajun Liu, Choujun Zhan, Chen Lin, Weiwei Cai, *Member, IEEE*, and Yunwen Chen

***Abstract*—Group recommendation has recently drawn a lot of attention to the recommender system community. Currently, several deep learning-based approaches are leveraged to learn preferences of groups for items and predict next items in which groups may be interested. Yet, their recommendation performance is still unsatisfactory due to sparse group–item interactions. To address this challenge, this study presents a novel model, called group recommendation model with two-stage deep learning (GRMTDL), which encompasses two sequential stages: 1) group representation learning (GRL) and 2) group preference learning (GPL). In GRL, we first construct an undirected tripartite graph over group–user–item interactions, and then employ it to accurately learn group semantic features through a spatial-based variational graph autoencoder network. While in GPL, we first introduce a dual PL-network that contains two structure-sharing subnetworks: 1) group PL-network employed for GPL and 2) user PL-network utilized for user preference learning. Then, we design a novel layered transfer learning (LTL) method to learn group preferences by alternately optimizing these two subnetworks. In particular, it can effectively absorb knowledge of user preferences into the process of GPL. Furthermore, extensive experiments on four real-world datasets demonstrate that the proposed GRMTDL model outperforms the state-of-the-art baselines for group recommendation.**

***Index Terms*—Deep learning, graph autoencoder, group recommendation, knowledge transferring, representation learning.**

I. Introduction

CURRENTLY, most researchers focus on individual recommendation. The term "individual" generally refers to a particular user in the recommender system [1]. The task of individual recommendation is to predict a user's possible interested items based on her historical preferences and recommend top-k items to her [2]. In recent years, the wide application of social networks and online communities, such as Meetup, Facebook, and YouTube, has made it convenient for people to organize and participate in group activities [3]. Providing accurate recommendations for group activities is an important task for modern recommender systems. We refer to this task as a "group recommendation" problem that is challenging and has drawn a lot of attention in industry and academia [4].

According to our investigation, the existing studies on group recommendation include two broad categories: 1) memory-based and 2) model-based approaches [5]. More specifically, the former simply aggregates preferences of members (i.e., users) without considering interactions among members in a group [6]–[10]. The latter focuses on modeling groups' decision-making processes, and produces the most favorable items for groups [11]–[18].

With great successes in individual recommendation [19], [20], deep learning begins to attract much interest to group recommendation and brings more opportunities to raise recommendation effectiveness. To the best of our knowledge, they basically use various deep neural network components to model groups' decision-making processes [21]–[35]. Specifically, several neural attentive approaches [28]–[35] have shown the superior performance among existing studies. Their core idea is to adopt an attention mechanism [36] to learn to capture the different impacts of different members in a group, which can improve the final recommendation effectiveness.

Nevertheless, they still have two drawbacks that greatly affect the effectiveness of group recommendation. First, for a group, they only consider the impacts of members in this group, and ignore the impacts of other groups and external members. That is, they focus on the internal characteristics of a group and ignore its external characteristics. This directly affects the adequacy and semantic accuracy of group feature representation. Second, although they have presented various ingenious neural networks for performing group preference learning (GPL), they cannot effectively capture group preferences for items, which is mainly because unlike user–item interaction data, group–item interaction data are usually very sparse. It will lead to insufficient model training, which greatly affects the accuracy of recommendation.

To address the above drawbacks, in this article, we introduce a novel model, namely, the group recommendation model with

Manuscript received 12 March 2020; revised 27 November 2020, 16 March 2021, and 4 August 2021; accepted 16 November 2021. Date of publication 10 December 2021; date of current version 18 August 2022. This work was supported by the National Natural Science Foundation of China under Grant 62172166 and Grant 61772366. This article was recommended by Associate Editor E. Chen. *(Corresponding author: Zhenhua Huang.)*

Zhenhua Huang, Yajun Liu, and Choujun Zhan are with the School of Computer Science, South China Normal University, Guangzhou 510631, China (e-mail: jukiehuang@163.com; 75856599@qq.com; zchoujun2@my.cityu.edu.hk).

Chen Lin is with the School of Informatics, Xiamen University, Xiamen 361000, China (e-mail: chenlin@xmu.edu.cn).

Weiwei Cai is with the School of Logistics and Transportation, Central South University of Forestry and Technology, Changsha 410004, China, and also with the Graduate School, Northern Arizona University, Flagstaff, AZ 86011 USA (e-mail: vivitsai@ieee.org).

Yunwen Chen is with the Research and Development Department, DataGrand Inc., Shenzhen 518063, China (e-mail: chenyunwen@datagrand.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2021.3131349.

Digital Object Identifier 10.1109/TSMC.2021.3131349

two-stage deep learning (GRMTDL). It mainly encompasses two sequential stages: 1) group representation learning (GRL) and 2) GPL. In the GRL stage, we first construct an undirected tripartite graph for group–user–item interactions by using group and user historical data in a given recommender system. Then, inspired by [37], we take it as input, and introduce a spatial-based variational graph autoencoder (SVGAE) network to learn group semantic features accurately. We use a two-layer graph convolutional network (GCN) as an encoder and employ a simple inner-product layer as a decoder [37]–[42]. Meanwhile, the initial features of all groups, users, and items can be obtained through existing superior group recommendation models. At the end of GRL, we can not only obtain accurate semantic features of groups, but also adequately optimize semantic features of users and items. Hence, the GRL stage has an additional function, namely, that it can raise recommendation effectiveness of existing individual models.

In the GPL stage, we first introduce a dual PL-network that comprises two structure-sharing three-layer subnetworks. One is a group PL-network that takes a feature pair in the form of (group, item) as an input and is employed for GPL. The other is a user PL-network that uses a feature pair in the form of (user, item) as an input and is leveraged for user preference learning. Generally, the user–item interaction data are much denser than the group–item interaction data. Based on this fact, we further design a novel layered transfer learning (LTL) method to learn group preferences, which alternately optimizes two subnetworks and transfers parameter gradients (i.e., metaknowledge) of the user PL-network to the group PL-network with a certain probability. According to the previous works [43], [44], we know that compared with a lower layer of neural networks, an upper layer is generally more abstract and contains more information about preferences. Therefore, the transferring probability of an upper layer is set to be greater than that of a lower layer in the LTL method. In this way, we can effectively absorb knowledge of user preferences into the process of GPL.

In summary, the main contributions of this article are as follows.

1) The accuracy of group feature representation remarkably affects the final recommendation effectiveness. We make the first attempt to construct an undirected tripartite graph from group–user–item interactions and to accurately learn group semantic features via an SVGAE neural network. In addition, it simultaneously refines user and item semantic features, which can heighten the recommendation performance of individual models.
2) To learn accurate group preferences on sparse group–item interaction data, we first introduce a dual PL-network that contains two structure-sharing subnetworks, i.e., group and user PL-networks. Then, we propose a novel LTL method to alternately optimize two subnetworks and to transfer knowledge of user preferences into the process of GPL.
3) We propose GRMTDL and comprehensively investigate its effectiveness through extensive experiments over four real-world datasets. Our experimental results show that it significantly outperforms existing state-of-the-art models in terms of various evaluation metrics.

Section II introduces the related works to this article. Section III provides the details of the proposed GRMTDL model. The experimental results are presented in Section IV. Finally, Section V concludes this article and outlines future work.

II. Related Works

Existing works on group recommendation involve two categories: 1) memory-based and 2) model-based approaches.

A. Memory-Based Approaches

Memory-based approaches simply aggregate preferences of members without considering interactions among members in a group. They can be further divided into two subcategories. The first one contains preference aggregation-based approaches that get a group's profile by combining all member preferences and recommend potential items to this group [6]–[8].

While the second one comprises score aggregation-based approaches that get the most favorable items for each member and then combine all obtained items to make the final recommendations through the strategies, such as average (AVG), least misery (LM), and maximum satisfaction (MS). AVG calculates the AVG score of an item across group members as the final recommendation score of this item [7]. LM obtains the lowest score of an item among all group members' scores as the final recommendation score of this item [5]. MS is similar to AVG. The difference is that the scores of selected items in MS need to be greater than a specified threshold [9], [10].

B. Model-Based Approaches

Unlike memory-based approaches, model-based ones focus on modeling groups' decision-making processes and then provide potential items for groups. Usually, their effectiveness is better than that of memory-based approaches [5]. Specifically, they comprise traditional and deep learning-based approaches. Traditional approaches perform groups' decision-making processes by using information fusion [11]–[15], game theory [16], probabilistic models [17], [18], etc.

Deep learning-based approaches generally employ various deep neural network components for modeling groups' decision-making processes [21]–[27]. Recently, several neural attentive approaches [28]–[35] have been introduced to further improve recommendation performance. Among them, the works [28] and [29] are the first two to employ an attention mechanism. Attention-based group recommendation (AGR) [28] is introduced to capture the influence of each group member via learning its importance weight and to perform group recommendation based on learned members' weights. While attentive group recommendation (AGREE) [29] is proposed to adopt an attention mechanism to optimize group feature representation and to learn interactions between groups and items with a neural collaborative filtering framework [39].

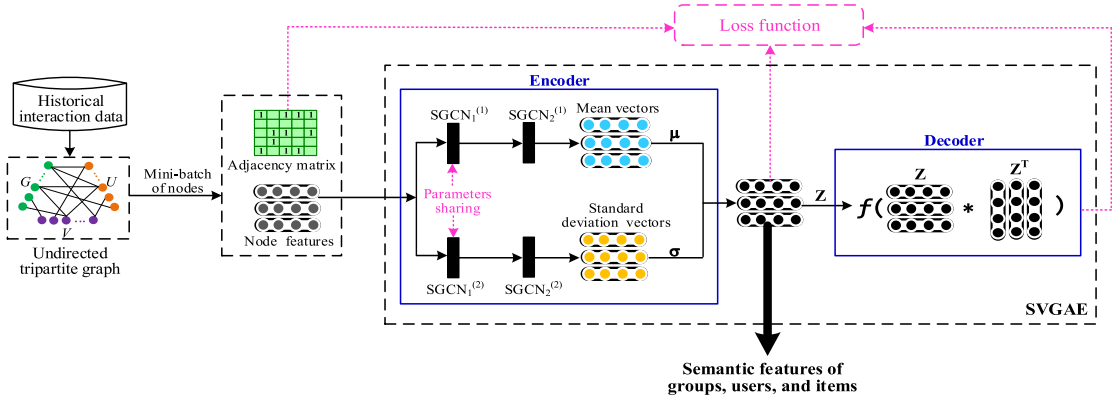

Fig. 1. Overall framework used in the GRL stage.

Based on the study of [29], Cao *et al.* [30] incorporated a social attribute "followee" into group recommendation, and presented social-enhanced AGREE (SoAGREE) that employs an attention network to aggregate each group member's followees. Huang *et al.* [31] proposed a novel model multiattention-based group recommendation model (MAGRM), which fully considers groups' sociality and preference interactions between groups and their members. Yin *et al.* [32] introduced a framework social influence-based group recommender (SIGR) that utilizes bipartite graph embedding and attention mechanism as building blocks. Specifically, every group member is associated with a virtual social influence, and then members' social information is employed to evaluate their social influences. Wang *et al.* [33] introduced a socially driven multiinteraction attentive group representation approach for learning static and dynamic group preference coherently. Zan *et al.* [34] presented an effective model user difference attention (UDA) that explicitly simulates the comparisons between group members through relational attention. Specifically, several user relational kernels (URKs) are given to simulate different types of relations during group decision making. In addition, Chen *et al.* [35] introduced a model attentive multitask learning-based group itinerary recommendation (AMT-IRE) that employs the attention mechanism to learn the inner relations between group members dynamically and obtain consensus group preferences.

Nevertheless, as discussed in Section I, the above-mentioned attention-based approaches still have two major drawbacks that may lead to poor recommendation performance, and addressing these two drawbacks partially motivates this study.

III. Details of the Proposed GRMTDL Model

GRMTDL encompasses two sequential stages: 1) GRL and 2) GPL. In this section, we first formulate the group recommendation problem to be solved. We then present the two key ingredients of GRMTDL.

A. Problem Statement

Suppose that the sets of groups, users, and items are G, U, and V, respectively, in a given recommender system, and for each $g \in G$, $g \subseteq U$. We use D_g and D_u to denote group–item interactions and user–item interactions, respectively. Let $D = D_g \cup D_u$. Then, given a target group g, the group recommendation task is defined as recommending a list of items that g may be interested in, which is formally defined as [29] follows.

Input: The sets of groups, users, and items G, U, and V; group–item interactions and user–item interactions D.

Output: The personalized ranking function f that map an item to a real value for each group $f_g(g, v) \to \mathbb{R}$.

In this article, we realize a personalized ranking function f_g by training the proposed model GRMTDL, and each training instance corresponds to a group. Unlike user–item interaction data, group–item interaction data are usually very sparse. Hence, in this work, based on the idea of knowledge transferring, we consider using an additional individual (i.e., user) recommendation task to improve effectiveness of group recommendation. Similarly, to perform this task, user–item interactions D_u are used to train a personalized ranking function $f_u : f_u(u, v) \to \mathbb{R}$, and each training instance corresponds to a user. Note that f_u is also realized in our GRMTDL model.

B. GRL: Group Representation Learning

The GRL stage learns group semantic features accurately via an SVGAE network. User and item semantic features can be simultaneously optimized in GRL, which is an additional gain. Fig. 1 shows the framework used in this stage.

In GRL, we first generate an undirected tripartite graph $\mathcal{H} = (N, E)$ for group–user–item interactions based on D. Here, the node set $N = G \cup U \cup V$, and the edge set $E = \{(g, u)|g \in G$ and $u \in U$ and u is a member of $g\} \cup \{(g, v)|g \in G$ and $v \in V$ and g has interacted with $v\} \cup \{(u, v)|u \in U$ and $v \in V$ and u has interacted with $v\}$. In our study, $\mathcal{H}$ is used to depict the correlation among groups, users, and items. Then, inspired by [37], we employ an SVGAE network to learn group semantic features accurately as well as to optimize user and item semantic features from $\mathcal{H}$. As shown in Fig. 1, the encoder in SVGAE has two branches, and either of them contains a two-layer SGCN. For simplicity, the upper and lower branches are denoted as Υ_u and Υ_l, respectively. Meanwhile, to improve the training efficiency, their first layers share weight parameters.

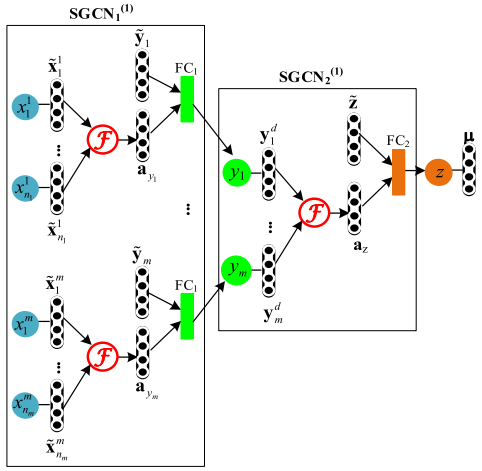

Fig. 2. Process performed by the branch Υ_u.

On the other hand, the decoder uses a simple inner product layer [37].

When training the SVGAE network, we sample a minibatch of nodes from $\mathcal{H}$ at a time, denoted as $\mathcal{B} = \{z_1, z_2, \ldots, z_b\}$. Here, b is the size of minibatch. By leveraging existing models, we can usually get more accurate user and item semantic features than those of groups [28]–[35]. So, to learn group semantic features adequately, the number n_g of groups in $\mathcal{B}$ needs to be greater than the sum of the number of users and items. In this study, we set $n_g = \lceil 60\% \cdot b \rceil$, and the selection of its optimal value will be one of our future work. After node sampling, we can get an adjacency matrix $\mathcal{A} \in \mathbb{R}^{b\times b}$ of $\mathcal{B}$ and a d_0-dimensional initial feature $\widetilde{\mathbf{z}}$ of each node $z \in \mathcal{B}$.

Both branches Υ_u and Υ_l of the encoder take each node $z \in \mathcal{B}$ as an input sample. They are leveraged for constructing a d_m-dimensional mean vector $\boldsymbol{\mu}$ and a d_t-dimensional standard deviation vector $\boldsymbol{\sigma}$, respectively. In this way, the encoder can eventually generate a d_s-dimensional latent vector $\mathbf{z}$ for node z. Since these two branches perform the same process, we use Υ_u as an example to show this process, which is shown in Fig. 2.

For each node $z \in \mathcal{B}$, we obtain all its direct neighbors $N_z = \{y_1, y_2, \ldots, y_m\}$, where m is the cardinality of N_z. Then, for each $y_i \in N_z$, we further get all its direct neighbors $N_{y_i} = \{x_1^{y_i}, x_2^{y_i}, \ldots, x_{n_{y_i}}^{y_i}\}$, where n_{y_i} is the cardinality of N_{y_i}. Based on [45], in the first SGCN layer $\mathrm{SGCN}_1^{(1)}$, we generate a d_a-dimensional aggregation vector $\mathbf{a}_{y_i}$ for each y_i by employing the following aggregation function ($1 \le t \le n_{y_i}$):

$$\mathbf{a}_{y_i} = \mathcal{F}\left(N_{y_i}\right) = \sum_{x_t^{y_i} \in N_{y_i}} \left(\frac{\mathrm{ReLU}\left(\mathbf{W}_1^{(1)} \widetilde{\mathbf{x}}_t^{y_i} + \mathbf{b}_1^{(1)}\right)}{J\left(x_t^{y_i},\ y_i\right)} \right). \quad (1)$$

$\widetilde{\mathbf{x}}_t^{y_i}$ is the initial feature of $x_t^{y_i}$ and the rectified linear unit (ReLU) [42] is an activation function. $\mathbf{W}_1^{(1)} \in \mathbb{R}^{d_a\times d_0}$ and $\mathbf{b}_1^{(1)} \in \mathbb{R}^{d_a}$ are the two trained parameters. While $J(x_t^{y_i},\ y_i)$ is a graph Laplacian norm [45]

$$J\left(x_t^{y_i},\ y_i\right) = \left(d\left(x_t^{y_i}\right) \cdot d(y_i)\right)^{1/2} \quad (2)$$

where $d(y_i)$ is the degree of node y_i. On this basis, we take $\mathbf{a}_{y_i}$ and $\widetilde{\mathbf{y}}_i$ (the initial feature of y_i) as an input, and employ a fully connected layer FC_1 to produce a d_e-dimensional derivation vector $\mathbf{y}_i^d$ for y_i

$$\mathbf{y}_i^d = \mathrm{Sigmoid}\left(\mathbf{Q}_1^{(1)}\left(\mathbf{a}_{y_i} \bigoplus \widetilde{\mathbf{y}}_i\right) + \mathbf{p}_1^{(1)}\right). \quad (3)$$

Here, "$\bigoplus$" is a concatenation operation, and $\mathbf{Q}_1^{(1)} \in \mathbb{R}^{d_e\times(d_a+d_0)}$ and $\mathbf{p}_1^{(1)} \in \mathbb{R}^{d_e}$ are the two trained parameters.

$\mathrm{SGCN}_1^{(1)}$ finally outputs m derivation vectors for N_z, i.e., $\mathbf{y}_1^d, \mathbf{y}_2^d, \ldots, \mathbf{y}_m^d$. The second SGCN layer $\mathrm{SGCN}_2^{(1)}$ uses $\mathcal{F}$ on these m derivation vectors to get a d_a-dimensional aggregation vector $\mathbf{a}_z$ for z. Then, it is combined with $\widetilde{\mathbf{z}}$ to produce a mean vector $\boldsymbol{\mu}$ through a fully connected layer FC_2

$$\mathbf{a}_z = \mathcal{F}(N_z) = \sum_{y_i \in N_z} \left(\frac{\mathrm{ReLU}\left(\mathbf{W}_2^{(1)} \mathbf{y}_i^d + \mathbf{b}_2^{(1)}\right)}{J(y_i,\ z)} \right) \quad (4)$$

$$\boldsymbol{\mu} = \mathrm{Sigmoid}\left(\mathbf{Q}_2^{(1)}\left(\mathbf{a}_z \bigoplus \widetilde{\mathbf{z}}\right) + \mathbf{p}_2^{(1)}\right). \quad (5)$$

$\mathbf{W}_2^{(1)} \in \mathbb{R}^{d_a\times d_0}$, $\mathbf{b}_2^{(1)} \in \mathbb{R}^{d_a}$, $\mathbf{Q}_2^{(1)} \in \mathbb{R}^{d_e\times(d_a+d_0)}$, and $\mathbf{p}_2^{(1)} \in \mathbb{R}^{d_e}$ are the four trained parameters, and $L(y_i,\ z)$ is calculated via (2).

Similarly, we generate a standard deviation vector $\boldsymbol{\sigma}$ for z via the branch Υ_l containing $\mathrm{SGCN}_1^{(2)}$ and $\mathrm{SGCN}_2^{(2)}$. Specifically, we replace $\boldsymbol{\sigma}$ by $\log\boldsymbol{\sigma}$ to ensure that the standard deviation is nonnegative. Because $\mathrm{SGCN}_1^{(1)}$ and $\mathrm{SGCN}_1^{(2)}$ share parameters, Υ_l totally has eight parameters in $\mathrm{SGCN}_2^{(2)}$, denoted as: $\mathbf{W}_1^{(2)} \in \mathbb{R}^{d_a\times d_0}$, $\mathbf{b}_1^{(2)} \in \mathbb{R}^{d_a}$, $\mathbf{Q}_1^{(2)} \in \mathbb{R}^{d_e\times(d_a+d_0)}$, $\mathbf{p}_1^{(2)} \in \mathbb{R}^{d_e}$, $\mathbf{W}_2^{(2)} \in \mathbb{R}^{d_a\times d_0}$, $\mathbf{b}_2^{(2)} \in \mathbb{R}^{d_a}$, $\mathbf{Q}_2^{(2)} \in \mathbb{R}^{d_e\times(d_a+d_0)}$, and $\mathbf{p}_2^{(2)} \in \mathbb{R}^{d_e}$. Based on $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, a Gaussian distribution of latent vector $\mathbf{z}$ can be obtained

$$\phi(\mathbf{z}) = \mathcal{N}\left(\mathbf{z}|\mu,\ \mathrm{diag}\left(\boldsymbol{\sigma}^2\right)\right) \quad (6)$$

where $\mathrm{diag}(\cdot)$ is a diagonal matrix.

Thus, the encoder eventually outputs a Gaussian distribution of latent vectors for minibatch $\mathcal{B}$

$$\phi(\mathbf{Z}) = \phi([\mathbf{z}_1,\ \mathbf{z}_2, \ldots, \mathbf{z}_b]) = \prod_{i=1}^{b} \mathcal{N}\left(\mathbf{z}_i|\boldsymbol{\mu}_i,\ \mathrm{diag}\left(\boldsymbol{\sigma}_i^2\right)\right). \quad (7)$$

$\mathbf{z}_i$ is the latent vector of node $z_i \in \mathcal{B}$, and $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i^2$ are its mean and standard deviation vectors, respectively. The decoder performs an inner product between latent vectors and generates the following conditional probability distribution:

$$\begin{cases} p(\mathcal{A}|\mathbf{Z}) = \prod_{i=1}^{b}\prod_{j=1}^{b} p(\mathcal{A}_{ij}|\ \mathbf{z}_i, \mathbf{z}_j) \\ p(\mathcal{A}_{ij} = 1|\ \mathbf{z}_i, \mathbf{z}_j) = f\left(\mathbf{z}_i \cdot \mathbf{z}_j^{\mathrm{T}}\right) \end{cases} \quad (8)$$

where $\mathcal{A}_{ij}$ are the elements of the adjacency matrix $\mathcal{A}$ and $f(\cdot)$ is a logistic sigmoid function.

Then, similar to the work [37], the loss function $\mathcal{L}$ used for training SVGAE is defined as

$$\mathcal{L}(\mathcal{B}; \mathcal{R}_{\mathrm{all}}) = -\left(\lambda \mathbb{E}_{\phi(\mathbf{Z})}\left[\log p\left(\mathcal{A}|\mathbf{Z}\right)\right] - (1-\lambda)\mathrm{KL}[\phi(\mathbf{Z})\|\psi(\mathbf{Z})]\right). \quad (9)$$

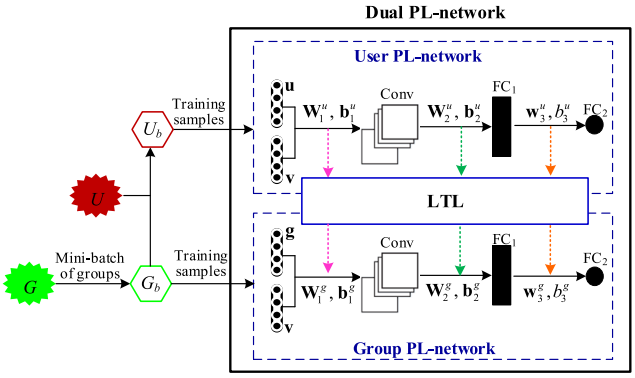

Fig. 3. Overall framework used in the GPL stage.

$\psi(\mathbf{Z}) = \prod_{i=1}^{b} \mathcal{N}(\mathbf{z}_i) = \prod_{i=1}^{b} \mathcal{N}(\mathbf{z}_i|0, \mathbf{I})$ is a Gaussian prior, and $\mathrm{KL}[\phi(\cdot)\|\psi(\cdot)]$ is the Kullback–Leibler divergence between $\phi(\cdot)$ and $\psi(\cdot)$. $\lambda \in [0, 1]$ is a hyperparameter. $\mathcal{R}_{\text{all}}$ is the set of all the parameters in SVGAE. That is, $\mathcal{R}_{\text{all}} = \{\mathbf{W}_1^{(1)}, \mathbf{b}_1^{(1)}, \mathbf{Q}_1^{(1)}, \mathbf{p}_1^{(1)}, \mathbf{W}_2^{(1)}, \mathbf{b}_2^{(1)}, \mathbf{Q}_2^{(1)}, \mathbf{p}_2^{(1)}, \mathbf{W}_1^{(2)}, \mathbf{b}_1^{(2)}, \mathbf{Q}_1^{(2)}, \mathbf{p}_1^{(2)}, \mathbf{W}_2^{(2)}, \mathbf{b}_2^{(2)}, \mathbf{Q}_2^{(2)}, \mathbf{p}_2^{(2)}\}$.

We leverage a minibatch stochastic gradient descent (SGD) method [46] to minimize $\mathcal{L}$. Each parameter is randomly initialized with the Glorot strategy [47]

$$\mathbf{\Delta} \leftarrow \mathcal{T}_{\text{udf}}\left(-\sqrt{\frac{6}{d_0}}, \sqrt{\frac{6}{d_0}}\right), \text{ for each } \mathbf{\Delta} \in \mathcal{R}_{\text{all}}. \quad (10)$$

Here, $\mathcal{T}_{\text{udf}}(\ )$ is a uniform distribution function and d_0 is the dimensionality of node initial features. At the same time, an RMSprop optimizer [48] is used to update each parameter

$$\begin{cases} [r]_t = 0.9[r]_{t-1} + 0.1\left(\frac{\partial \mathcal{L}}{\partial [\mathbf{\Delta}]_t}\right)^2 \\ [\mathbf{\Delta}]_t = [\mathbf{\Delta}]_{t-1} - \left(\frac{\alpha}{\sqrt{[r]_t + \gamma}}\right) \frac{\partial \mathcal{L}}{\partial [\mathbf{\Delta}]_t} \end{cases} \text{for each } \mathbf{\Delta} \in \mathcal{R}_{\text{all}}. \quad (11)$$

Here, t is the current iteration index, r is a gradient cumulative variable, α is an initial learning rate, and $\gamma = 10^{-8}$ is a constant to ensure a nonzero denominator.

It is not difficult to see that the computational complexity in the GRL stage contains two main parts: 1) $O(K_{in})$ for generating an undirected tripartite graph $\mathcal{H}$ and 2) $O([K_{in}^2/K_{nd}]+K_{nd})$ for training the SVGAE network. Here, K_{nd} is the sum of the number of groups, users, and items, and K_{in} is the total number of group–user–item interactions. Hence, we have the complexity of $O([K_{in}{}^2/K_{nd}] + K_{nd} + K_{in}) = O([K_{in}^2/K_{nd}])$ in GRL.

C. GPL: Group Preference Learning

In the GPL stage, we introduce a dual PL-network for learning group preferences via a novel LTL method. Fig. 3 shows the framework utilized in this stage. Specifically, group–item interaction data are usually much sparser than user–item interaction data, and hence, we use the LTL method to absorb knowledge of user preferences into the process of GPL. In this way, GRMTDL can learn group preferences more accurately than current models.

As shown in Fig. 3, the dual PL-network encompasses two structure-sharing three-layer subnetworks: 1) group PL-network and 2) user PL-network, which are utilized to learn group and user preferences, respectively. Either subnetwork is composed of one convolutional layer (Conv) and two fully connected layers (FC_1 and FC_2). Meanwhile, FC_2 has only one neuron leveraged to output a preference probability.

When training this network, we first sample a minibatch G_b of groups from G at a time. On this basis, we produce a training sample set S_g for group PL-network.

1: $S_g \leftarrow \emptyset$;
2: **for** each $g \in G_b$ **do**
Randomly choose one positive item v^+ from V;
3: $S_g \leftarrow S_g \cup \{(\mathbf{g}, \mathbf{v}^+)\}$;
4: Randomly choose r negative items from V, denoted as V_n;
5: **for** each $v \in V_n$ **do** $S_g \leftarrow S_g \cup \{(\mathbf{g}, \mathbf{v})\}$

Note that $\mathbf{g}$ and $\mathbf{v}$ are the semantic features of g and v, respectively. In line 4, r is a negative sampling ratio, and by following the previous works, it is set to 4 here. Meanwhile, we get a user set based on G_b : $U_b = \{u|\text{for each } g \in G_b, u \in U \text{ and } u \text{ is a member of } g\}$. Similarly, we generate a training sample set S_u for user PL-network via U_b. Because these two subnetworks share the same structure, we next use the group PL-network as an example to show the process of forward propagation. For each training sample $(\mathbf{g}, \mathbf{v}) \in S_g$, Conv is defined as

$$\mathbf{C} = \text{ReLU}\left(\mathbf{W}_1^g \bigotimes \left(\mathbf{g} \bigoplus \mathbf{v}\right) + \mathbf{b}_1^g\right) \quad (12)$$

where "$\bigotimes$" is a convolution operation, and $\mathbf{W}_1^g \in \mathbb{R}^{n_c \times c_h \times c_w}$ and $\mathbf{b}_1^g \in \mathbb{R}^{n_c}$ are the two trained parameters. Based on this, $\mathbf{C} \in \mathbb{R}^{([2d_0 - c_h c_w/l_c]+1) \times n_c}$. Here, n_c is the number of convolution kernels, c_h and c_w are the height and width of kernels, respectively, and l_c is the step size. Then, FC_1 and FC_2 are defined as

$$\mathbf{F}_1 = \text{Sigmoid}\left(\mathbf{W}_2^g \overleftrightarrow{\mathbf{C}} + \mathbf{b}_2^g\right) \quad (13)$$

$$y_g = \text{Sigmoid}\left(\mathbf{w}_3^g \mathbf{F}_1 + b_3^g\right). \quad (14)$$

In (13) and (14), $\overleftrightarrow{\mathbf{C}} \in \mathbb{R}^{([2d_0 - c_h c_w/l_c]+1)n_c}$ is the flat vector of $\mathbf{C}$. $\mathbf{W}_2^g \in \mathbb{R}^{n_f^1 \times ([2d_0 - c_h c_w/l_c]+1)n_c}$, $\mathbf{b}_2^g \in \mathbb{R}^{n_f^1}$, $\mathbf{w}_3^g \in \mathbb{R}^{n_f^2 \times n_f^1}$, and $b_3^g \in \mathbb{R}^{n_f^2}$ are the four trained parameters. Here, n_f^1 and n_f^2 are the number of neurons in FC_1 and FC_2, respectively. We set $n_f^2 = 1$ (i.e., y_g is a scalar).

Then, the loss function $\mathcal{L}^g$ is defined as

$$\mathcal{L}^g\left(S_g; \mathcal{R}_{\text{all}}^g\right) = -\sum_{s \in S_g} \left(\underbrace{\log y_g}_{s=(g,\mathbf{v}^+)} + \underbrace{\log\left(1 - y_g\right)}_{s=(g,\mathbf{v}^-)} \right). \quad (15)$$

$\mathcal{R}_{\text{all}}^g$ is the set of all the parameters in the group PL-network, i.e., $\mathcal{R}_{\text{all}}^g = \{\mathbf{W}_1^g, \mathbf{b}_1^g, \mathbf{W}_2^g, \mathbf{b}_2^g, \mathbf{w}_3^g, b_3^g\}$. Similarly, we can define a loss function for user PL-network: $\mathcal{L}^u(S_u; \mathcal{R}_{\text{all}}^u)$. Here, $\mathcal{R}_{\text{all}}^u = \{\mathbf{W}_1^u, \mathbf{b}_1^u, \mathbf{W}_2^u, \mathbf{b}_2^u, \mathbf{w}_3^u, b_3^u\}$.

Similar to SVGAE, we utilize a minibatch SGD method to minimize $\mathcal{L}^g$ and $\mathcal{L}^u$. Meanwhile, all the parameters are randomly initialized with the Glorot strategy and updated with an RMSprop optimizer.

To absorb knowledge of user preferences into the process of GPL, we design an LTL method to train the dual PL-network. It optimizes two subnetworks alternately, and transfers parameter gradients (i.e., metaknowledge) of the user PL-network to group PL-network with a certain probability. The specific process is described below. For the tth iteration, we first take S_g as input to group PL-network and update each parameter $\mathbf{\Delta}_l^g \in \mathcal{R}_{\text{all}}^g$. Here, l ($1 \le l \le 3$) is the layer number of network, and each layer has two trained parameters. That is, $\mathbf{\Delta}_l^g$ represents $\mathbf{W}_l^g$ or $\mathbf{b}_l^g$, for $l = 1{\sim}2$; and $\mathbf{\Delta}_3^g$ represents $\mathbf{w}_3^g$ or b_3^g. Next, we take S_u as input to user PL-network and calculate a gradient $\partial \mathcal{L}^u / \partial [\mathbf{\Delta}_l^u]_t$ for each parameter $\mathbf{\Delta}_l^u \in \mathcal{R}_{\text{all}}^u$.

On this basis, we further update each parameter $\mathbf{\Delta}_l^g \in \mathcal{R}_{\text{all}}^g$ with a transferring probability $p^{(l)}$

$$\begin{cases} [\mathbf{\Delta}_l^g]_t = [\mathbf{\Delta}_l^g]_t - \rho\left(\frac{\alpha}{\sqrt{[r^g]_t + \gamma}}\right) \frac{\partial \mathcal{L}^u}{\partial [\mathbf{\Delta}_l^u]_t} \\ p^{(l)} = 0.8/\sqrt{2^{3-l}} \end{cases} \tag{16}$$

where ρ is a gradient transfer rate, r, α, and γ have the same meaning as in (11). For example, $\mathbf{W}_1^g$ is updated (employing $\partial \mathcal{L}^u / \partial \mathbf{W}_1^u$) with a probability of 0.4, while $\mathbf{w}_3^g$ is updated (using $\partial \mathcal{L}^u / \partial \mathbf{w}_3^u$) with a probability of 0.8. Note that in (15), the transferring probability of an upper layer is set to be greater than that of a lower layer. It is mainly because that compared to a lower layer, an upper layer is generally more abstract and contains more information about preferences [43], [44]. How to determine the optimal probability in each layer will be a future work. Finally, each parameter $\mathbf{\Delta}_l^u \in \mathcal{R}_{\text{all}}^u$ is updated through an RMSprop optimizer.

The whole training process ends when the group PL-network converges. Note that if the user PL-network converges first, then only group PL-network continues to be trained until it also converges. While making predictions, given a group g and an item v, we directly input their semantic features (i.e., $\mathbf{g}$ and $\mathbf{v}$) into the group PL-network, and its output (i.e., y_g) is the predicted value of group preference.

Assume that there are K_g groups, K_u users, and K_v items in a recommender system. On AVG, each group has rated A_g items, and each user has rated A_u items ($A_g \ll K_v$, $A_u \ll K_v$). Then, we require $O(K_g A_g)$ and $O(K_u A_u)$ for training group and user PL-networks, respectively. Hence, we have the complexity of $O(K_g A_g + K_u A_u) = O(K_{in})$ in the GPL stage. K_{in} is the total number of group–user–item interactions.

IV. Experiments

A. Experimental Settings

1) Datasets: The first two datasets are mainly based on previous works [29], [30]: 1) Mafengwo and 2) CAMRa2011. Mafengwo is collected from a tourism website (*http://www.mafengwo.cn*) in which users can record their traveled venues, and create or join a group travel. It is a popular event-based dataset where an event encompasses a user group, and is occurred at a traveled venue. CAMRa2011 (*http://2011.camrachallenge.com/2011*) is a real-world dataset encompassing the movie rating records of individual users and households (i.e., groups). It is also a popular event-based dataset where an event contains a user household, and is occurred at a movie-watching venue. For simplicity, we denote these two datasets as MFW and CAM, respectively. Similar to previous works, in these two datasets, we treat an event as a group, where the users in the event are the group members, and the venue of the event is the item chosen by them. Our task is to recommend an appropriate venue for the group event.

TABLE I
Statistics of the Four Datasets

	MFW	CAM	ML-Simi	ML-Rand
Total groups	995	290	3,000	3,000
Total users	5,275	602	4,619	4,629
Total items	1,513	7,710	782	30659
Avg. group size	7.19	2.08	5	3
Avg. record for a user	7.54	193.26	191.92	166.55
Avg. record for an item	26.28	15.09	1,133.59	210.71

The last two datasets are extracted from MovieLens-1M (*http://grouplens.org/datasets/movielens/*) through the method in previous works [18], [28], [30]. MovieLens-1M is a popular database in the recommendation community, encompassing the information about users, movies, and ratings that users give to movies. Similar to previous works, we mainly focus on two types of groups in experimental evaluation: 1) similar and 2) random. Their corresponding datasets are denoted as ML-Simi and ML-Rand, respectively. For ML-Simi, users have high within-group similarities and low between-group similarities. While for ML-Rand, users are grouped randomly. Specifically, they can effectively represent two types of real-world groups: one is that a group consists of users having similar preferences, e.g., family. The other is that a group is composed of users gathering together temporarily, e.g., passengers on the same airplane. For each dataset, we randomly choose 3000 groups. Following previous works [28], [30], for a group, if each member gives four stars or above to a movie, we assume that the movie is adopted by this group.

The statistics of the four datasets are shown in Table I.

2) Evaluation Protocols: In experiments, each dataset is randomly divided into training (80%) and test (20%) sets. We conduct experimental evaluation on Pytorch platform [50], [51] and utilize an RMSprop optimizer [48] to update every parameter. We perform hyperparameter tuning to find the optimal value for each hyperparameter via random search [52]. Table II shows the hyperparameter settings of GRMTDL in experiments.

As for evaluation metrics, we adopt two well-known metrics HR@k (Hit Ratio) [25] and NDCG@k (normalized discounted cumulative gain) [53], which are widely used for top-k recommendation evaluation. In experiments, we consider four values of k, i.e., 1, 5, 10, and 15. For both of the two metrics, a larger metric value indicates a better effectiveness.

3) Compared Baselines: To verify effectiveness of GRMTDL, we compare it with the following state-of-the-art baselines.

a) Memory-based approaches: It includes DFM-AVG, DFM-LM, and DFM-MS. For a group, all these three

TABLE II
HYPERPARAMETER SETTINGS OF GRMTDL IN EXPERIMENTS

Stage	Hyper-parameter	MFW	CAM	ML-Simi	ML-Rand
GRL	d_0: dim. of initial features	64	64	64	64
	d_a: dim. of aggregation vectors	64	64	64	64
	d_e: dim. of derivation vectors	64	64	64	64
	d_m: dim. of mean vectors	64	64	64	64
	d_t: dim. of standard deviation vectors	64	64	64	64
	d_s: dim. of semantic features	64	64	64	64
	b: size of a mini-batch of nodes	100	100	100	100
	λ: reconstruction loss ratio	0.5	0.5	0.5	0.5
	α_r: initial learning rate	0.001	0.001	0.001	0.001
GPL	n_c: number of convolution kernels	10	10	10	10
	c_h: height of convolution kernels	2	2	2	2
	c_w: width of convolution kernels	4	4	4	4
	l_c: step size	1	1	1	1
	n_f^1: number of neurons in FC_1	640	640	640	640
	n_f^2: number of neurons in FC_2	1	1	1	1
	b_g: size of a mini-batch of groups	64	64	64	64
	ρ: gradient transfer rate	0.01	0.01	0.01	0.01
	α_r: initial learning rate	0.001	0.001	0.001	0.001

TABLE III
TOP-k RECOMMENDATION PERFORMANCE OF GRMTDL AND ITS PEERS ON MFW (BEST RESULTS ARE BOLD FACED)

	k=1		k=5		k=10		k=15	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
DFM-AVG	0.1529	0.1529	0.3294	0.2228	0.6294	0.3418	0.7941	0.4357
DFM-LM	0.1567	0.1567	0.3165	0.2341	0.6391	0.3526	0.8050	0.4627
DFM-MS	0.1588	0.1588	0.3705	0.2380	0.6117	0.3141	0.8017	0.4229
COM	0.1845	0.1845	0.4420	0.3297	0.5434	0.3727	0.7531	0.4436
DPMF-CNN	0.2584	0.2584	0.4675	0.3453	0.6342	0.3766	0.8078	0.4521
AGR	0.2704	0.2704	0.4803	0.4056	0.6413	0.4461	0.8068	0.4659
AGREE	0.2724	0.2724	0.4814	0.3747	0.6400	0.4244	0.8081	0.4663
SoAGREE	0.2812	0.2812	0.4898	0.3807	0.6481	0.4301	0.8099	0.4675
GRMTDL	**0.3242**	**0.3242**	**0.5314**	**0.4321**	**0.6847**	**0.4613**	**0.8120**	**0.5001**

TABLE IV
TOP-k RECOMMENDATION PERFORMANCE OF GRMTDL AND ITS PEERS ON CAM (BEST RESULTS ARE BOLD FACED)

	k=1		k=5		k=10		k=15	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
DFM-AVG	0.0655	0.0655	0.3000	0.1705	0.5724	0.2750	0.6241	0.2733
DFM-LM	0.1965	0.1965	0.5827	0.3307	0.7689	0.4490	0.8448	0.4629
DFM-MS	0.1206	0.1206	0.3103	0.1653	0.5724	0.2636	0.6482	0.2618
COM	0.1841	0.1841	0.5793	0.3762	0.7682	0.4368	0.8397	0.4649
DPMF-CNN	0.1989	0.1989	0.5802	0.3845	0.7532	0.4545	0.8304	0.4632
AGR	0.2000	0.2000	0.5862	0.3951	0.7690	0.4587	0.8503	0.4761
AGREE	0.1890	0.1890	0.5821	0.3889	0.7710	0.4576	0.8517	0.4779
SoAGREE	0.1890	0.1890	0.5821	0.3889	0.7710	0.4576	0.8517	0.4779
GRMTDL	**0.2276**	**0.2276**	**0.6217**	**0.4232**	**0.7918**	**0.4847**	**0.8522**	**0.5059**

baselines first utilize the model deep factorization machines (DFMs) [39] to get a recommendation score for its each member, and then use the AVG [7], LM [5], and MS [9] strategies to yield a final recommendation score for the group, respectively. In particular, DFM is a prevalent individual recommendation model based on a deep neural network.

b) Model-based approaches: It includes consensus model (COM) [18], DPMF-CNN [24], AGR [28], AGREE [29], and SoAGREE [30]. Among them, COM is a mainstream probabilistic model, while the other four models are all based on deep learning. The last three models also use the attention mechanism.

B. Performance Comparison With Baselines

We compare the effectiveness of the proposed model with its peers. Tables III–VI show HR@k and NDCG@k values for four real-world datasets. From these four tables, we can have the following observations.

1) The experimental results clearly show that the proposed model GRMTDL has superior recommendation accuracy than all the nine baselines. The main reasons

TABLE V
TOP-k RECOMMENDATION PERFORMANCE OF GRMTDL AND ITS PEERS ON ML-SIMI (BEST RESULTS ARE BOLD FACED)

	k=1		k=5		k=10		k=15	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
DFM-AVG	0.3269	0.3269	0.7273	0.5424	0.9168	0.6202	0.9350	0.6311
DFM-LM	0.3314	0.3314	0.7387	0.5477	0.8718	0.5847	0.9411	0.6122
DFM-MS	0.3233	0.3233	0.7184	0.5291	0.8924	0.5884	0.9464	0.6066
COM	0.2457	0.2457	0.5768	0.3972	0.7831	0.4664	0.8315	0.4903
DPMF-CNN	0.3714	0.3714	0.7613	0.5732	0.9076	0.6212	0.9525	0.6419
AGR	0.4024	0.4024	0.8008	0.6145	0.9246	0.6543	0.9630	0.6662
AGREE	0.3915	0.3915	0.8048	0.6177	0.9227	0.6561	0.9618	0.6583
SoAGREE	0.3915	0.3915	0.8048	0.6177	0.9227	0.6561	0.9618	0.6583
GRMTDL	**0.4201**	**0.4201**	**0.8317**	**0.6321**	**0.9524**	**0.6711**	**0.9723**	**0.6802**

TABLE VI
TOP-k RECOMMENDATION PERFORMANCE OF GRMTDL AND ITS PEERS ON ML-RAND (BEST RESULTS ARE BOLD FACED)

	k=1		k=5		k=10		k=15	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
DFM-AVG	0.3030	0.3030	0.7034	0.5292	0.8084	0.6086	0.8410	0.6097
DFM-LM	0.3149	0.3149	0.7176	0.5607	0.8161	0.6185	0.8422	0.6181
DFM-MS	0.3125	0.3125	0.7170	0.5780	0.8090	0.6102	0.8368	0.5857
COM	0.2034	0.2034	0.4245	0.3486	0.6312	0.3751	0.6628	0.4183
DPMF-CNN	0.3150	0.3150	0.7178	0.5674	0.8089	0.6101	0.8385	0.6195
AGR	0.3152	0.3152	0.7201	0.5783	0.8163	0.6185	0.8435	0.6220
AGREE	0.3154	0.3154	0.7204	0.5756	0.8171	0.6188	0.8457	0.6224
SoAGREE	0.3154	0.3154	0.7204	0.5756	0.8171	0.6188	0.8457	0.6224
GRMTDL	**0.3387**	**0.3387**	**0.7445**	**0.5913**	**0.8565**	**0.6352**	**0.8732**	**0.6423**

are three-folds. First, GRMTDL can model not only the impacts of internal members on a group but also the impacts of other groups and external members on it by using the group–user–item interaction graph in GRL. Second, for a group, GRMTDL integrates the semantic features of other groups and external members to learn its feature representation through the SVGAE neural network in GRL. Third, GRMTDL employs an LTL method in GPL to absorb knowledge of user preferences into the process of GPL.

2) Among the baselines, three memory-based models DFM- AVG, DFM-LM, and DFM-MS perform worst on all four datasets. It is mainly because that they simply aggregate preferences of group members and ignore the group decision-making process. This observation is consistent with [28] and [29].
3) Compared to other five baselines, three attention-based models (AGR, AGREE, and SoAGREE) have superior accuracy. The main reason is that they utilize the attention mechanism to capture the different impacts of different members in a group, which can improve the accuracy of group feature representation to a certain extent. Yet unlike GRMTDL, for each group, they only consider the impacts of its internal members on it. In addition, we can find that for the last three datasets, AGREE and SoAGREE have the same accuracy. The main reason is that the last three datasets do not have social information about followee, and therefore, SoAGREE degenerates into AGREE.

TABLE VII
EFFECTS OF L ON MFW AND CAM (BEST RESULTS ARE BOLD FACED)

		SVGAE-1	SVGAE-2	SVGAE-3	SVGAE-4
MFW	k=1	0.3171	0.3242	**0.3289**	0.3252
	k=5	0.5159	0.5314	**0.5427**	0.5313
	k=10	0.6519	0.6847	**0.6899**	0.6697
	k=15	0.7411	**0.8120**	0.8074	0.7425
CAM	k=1	0.2201	**0.2276**	0.2213	0.2215
	k=5	0.6161	**0.6217**	0.6201	0.6196
	k=10	0.7725	**0.7918**	0.7783	0.7734
	k=15	0.8390	**0.8522**	0.8511	0.8501

C. Performance Study for Group Representation Learning

Since the encoder plays a pivotal role in the SVGAE network, we study its impact on the final recommendation effectiveness. We first investigate the influence of the number L of layers in the encoder. L is set between 1 and 4. We use "SVGAE-L" to represent the model in which the encoder of SVGAE has L layers. It is easy to see that SVGAE-2 is exactly the GRMTDL model. Tables VII and VIII show HR@k values for four datasets. We see a similar accuracy trend for NDCG@k values and omit them here due to space limitation.

From the above two tables, we can have the following observations.

1) SVGAE-1 presents the worst recommendation accuracy over all four datasets. It indicates that SVGAE-1 is unable to produce group semantic features accurately, which is mainly because that the encoder with only one layer cannot effectively capture the impacts of a group's external members and other groups on it.

TABLE VIII
EFFECTS OF L ON ML-SIMI AND ML-RAND
(BEST RESULTS ARE BOLD FACED)

		SVGAE-1	SVGAE-2	SVGAE-3	SVGAE-4
ML-Simi	k=1	0.4155	0.4201	**0.4234**	0.4163
	k=5	0.8232	0.8317	**0.8341**	0.8248
	k=10	0.9477	0.9524	**0.9526**	0.9496
	k=15	0.9651	0.9723	**0.9729**	0.9695
ML-Rand	k=1	0.3280	0.3387	0.3308	**0.3398**
	k=5	0.7231	0.7445	0.7322	**0.7778**
	k=10	0.8412	0.8565	0.8432	**0.8601**
	k=15	0.8611	0.8732	0.8628	**0.8788**

TABLE IX
EFFECTIVENESS OF GROUP SEMANTIC FEATURES
ON MFW (BEST RESULTS ARE BOLD FACED)

	HR		NDCG	
	k=1	k=10	k=15	k=10
AGR	0.2704	0.6413	0.2704	0.4461
AGR+	**0.2856**	**0.6563**	**0.2856**	**0.4498**
AGREE	0.2724	0.6400	0.2724	0.4244
AGREE+	**0.2794**	**0.6485**	**0.2794**	**0.4378**
SoAGREE	0.2812	0.6481	0.2812	0.4301
SoAGREE+	**0.2956**	**0.6546**	**0.2956**	**0.4432**

2) When adding layers to SVGAE-2, we can observe that SVGAE-3 and SVGAE-4 lead to overfitting over the dataset CAM. A possible reason is that the group–user–item interaction graph on CAM is relatively dense and a too deep network may introduce noise to GRL. Meanwhile, experimental results over the other three datasets indicate that SVGAE-3 and SVGAE-4 are not much improved compared to SVGAE-2, and SVGAE-2 is sufficient to produce group semantic features accurately. Based on this, we set L to 2 in our experimental evaluation.
3) By jointly analyzing Tables III–VIII, we can find that SVGAE-1 consistently outperforms the existing baselines over all four datasets. It verifies the effectiveness of the SVGAE network on group recommendation again.

Next, we evaluate whether new semantic features of groups generated in the GRL stage are beneficial for existing baselines. We mainly consider three attention-based models. For a model "XXX," we utilize "XXX+" to denote its corresponding variant that uses new semantic features. Table IX presents HR@k and NDCG@k values on the dataset MFW. We see a similar accuracy trend for other three datasets and omit them here due to space limitation. We can find that for each existing baseline, its recommendation accuracy is worse than that of its corresponding variant. It shows that new semantic features can help existing baselines to improve recommendation accuracy. This clearly reflects that GRMTDL has superior accuracy than existing baselines on GPL.

D. Performance Study for Group Preference Learning

Recall from the GPL stage in Section III-B that the LTL method plays an important role in GPL. Therefore, we first evaluate its impact on the final recommendation effectiveness. We mainly consider the following three variants here.

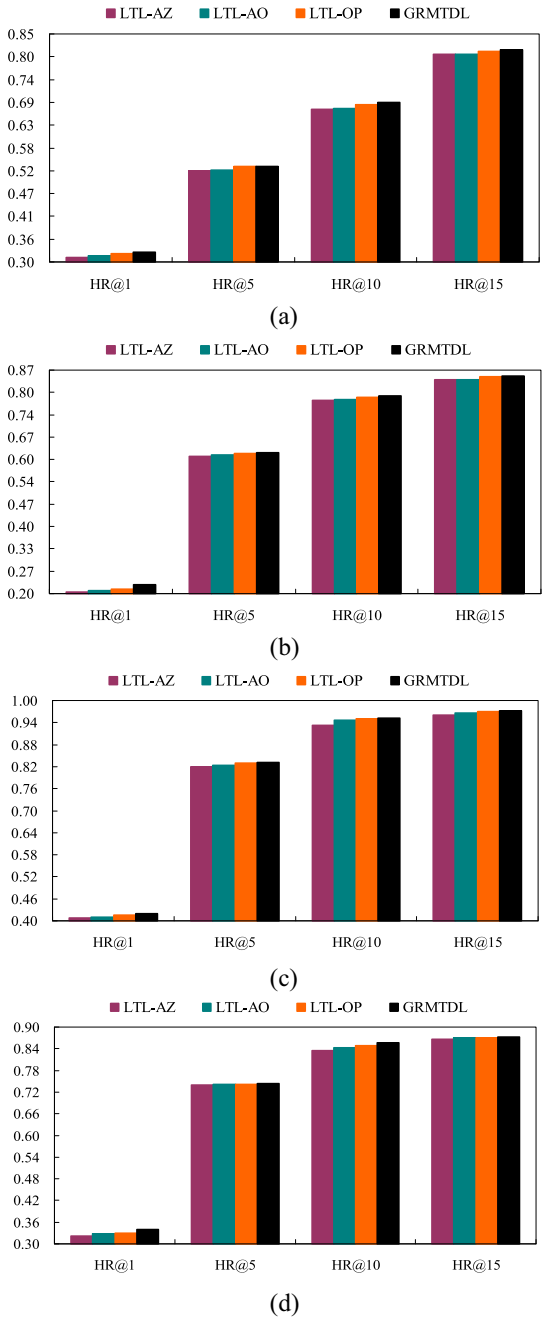

Fig. 4. Effectiveness of GRMTDL and its three variants in terms of HR@k. (a) MFW. (b) CAM. (c) ML-Simi. (d) ML-Rand.

1) LTL-AZ represents the model in which the transferring probability of each layer is set to 0. That is, we only train the group PL-network.
2) LTL-AO represents the model in which the transferring probability of each layer is set to 1.
3) LTL-OP represents the model in which the transferring probability of the lth layer $p^{(l)} = 0.8/\sqrt{2^{l-1}}$. It does exactly the opposite of LTL, and the transferring probability of an upper layer is set to be lower than that of a lower layer.

Fig. 4 shows HR@k values for four datasets. A similar accuracy trend can be observed for NDCG@k and we omit them

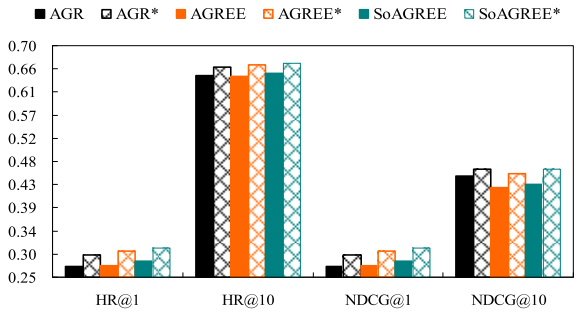

Fig. 5. Effectiveness of GPL on MFW.

due to space limitation. From Fig. 4, we can see that GRMTDL has superior recommendation accuracy than all three variants. It shows that: 1) the LTL method is effective for GPL and 2) neither the "zero transferring" strategy used in LTL-AZ nor the "complete transferring" strategy used in LTL-AO is good solution for GPL. It is worth mentioning that although the accuracy of GRMTDL is higher than that of LTL-OP, it does not mean that LTL is globally optimal. How to determine the optimal probability in each layer will be our future work.

Next, we study whether the GPL stage (i.e., GPL) is beneficial to existing baselines. Similarly, we consider the four attention-based models. For a model "XXX," we adopt "XXX*" to denote its corresponding variant that employs our GPL stage to learn group preferences. Fig. 5 demonstrates HR@k and NDCG@k values on the dataset MFW. We see a similar accuracy trend for other three datasets and omit them here due to space limitation.

Similar to Table IX, we can see from Fig. 5 that for every existing baseline, its recommendation accuracy is worse than that of its corresponding variant. This indicates that the process of GPL in GPL can help existing baselines to improve the recommendation accuracy. By jointly analyzing Table III and Fig. 5, we can further find that SoAGREE* has the best recommendation accuracy among the three variants, yet underperforms GRMTDL. It clearly reflects that GRMTDL has superior accuracy than existing baselines on GRL.

E. Performance Study for User and Item Semantic Features

Recall that GRL stage can also optimize user and item semantic features. Hence, in this section, we employ the user PL-network to evaluate whether new semantic features of users and items can improve effectiveness of individual recommendation. We compare the following four solutions.

1) UPL-O represents the solution that only takes old semantic features (i.e., initial features) of users and items as input to user PL-network.
2) UPL-U represents the solution that takes users' initial features and items' new semantic features as input to user PL-network.
3) UPL-I represents the solution that takes users' new semantic features and items' initial features as input to user PL-network.
4) UPL-N represents the solution that takes new semantic features of users and items as input to user PL-network.

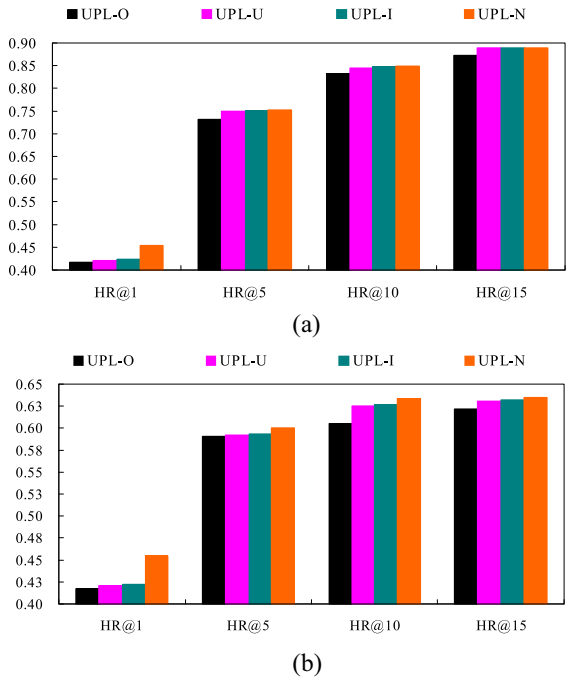

Fig. 6. Effectiveness of individual recommendation on MFW. (a) HR. (b) NDCG.

For simplicity, we only report HR@k and NDCG@k values on the dataset MFW here, shown in Fig. 6. We can see from the figure that UPL-N and UPL-O present the best and the worst recommendation accuracy, respectively. This shows that new semantic features can improve the effectiveness of individual recommendation. Meanwhile, we find that UPL-I outperforms UPL-U for all HR@k and NDCG@k values. A possible explanation is that compared to users, the interaction graph from MFW contains more semantic information about items.

V. Conclusion

This article introduces a novel model GRMTDL to improve the effectiveness of group recommendation, which mainly includes two sequential stages. In the GRL stage, we first construct a group–user–item interaction graph that is then fed to an SVGAE network to learn group semantic features accurately. In particular, SVGAE is able to simultaneously optimize user and item semantic features, which is an additional gain. On this basis, we first present a dual PL-network in the GPL stage, encompassing two structure-sharing three-layer subnetworks: 1) group and 2) user PL-networks. Then, we propose a novel LTL method to optimize two subnetworks alternately and to transfer knowledge of user preferences into the process of GPL. The experimental results on four real-world datasets show the effectiveness of our GRMTDL model.

In the future, we will continue to improve the recommendation performance of GRMTDL from two aspects. First, we will focus on determining the optimal value for n_g in the GRL stage. Second, we will consider how to obtain the optimal transferring probability of each layer in the dual PL-network.

REFERENCES

[1] L. Xin, Y. Yuan, M. Zhou, Z. Liu, and M. Shang, "Non-negative latent factor model based on β-divergence for recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 8, pp. 4612–4623, Aug. 2021.

[2] J. Ni, Z. Huang, J. Cheng, and S. Gao, "An effective recommendation model based on deep representation learning," *Inf. Sci.*, vol. 542, pp. 324–342, Jan. 2021.

[3] W. Wang, G. Zhang, and J. Lu, "Hierarchy visualization for group recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 6, pp. 1152–1163, Jun. 2019.

[4] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 754–765, 2009.

[5] T. D. Pessemier, S. Dooms, and L. Martens, "Comparison of group recommendation algorithms," *Multimedia Tools Appl.*, vol. 72, no. 3, pp. 2497–2541, Oct. 2014.

[6] Z. Yu, X. Zhou, Y. Hao, and J. Gu, "TV program recommendation for multiple viewers based on user profile merging," *User Model. User-Adapt. Interact.*, vol. 16, no. 1, pp. 63–82, Jun. 2006.

[7] K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth, and P. Nixon, "CATS: A synchronous approach to collaborative group recommendation," in *Proc. AAAI*, Boston, MA, USA, 2006, pp. 86–91.

[8] Y.-D. Seo, Y.-G. Kim, E. Lee, K.-S. Seol, and D.-K. Baik, "An enhanced aggregation method considering deviations for a group recommendation," *Expert Syst. Appl.*, vol. 93, pp. 299–312, Mar. 2018.

[9] L. Boratto and S. Carta, "Modeling the preferences of a group of users detected by clustering: A group recommendation case-study," in *Proc. WIMS*, Thessaloniki, Greece, 2014, pp. 1–7.

[10] Y. Jun and L. Peilin, "A new method of group information recommendation based on the user dynamic profile information optimization," in *Proc. ICIM*, London, U.K., 2021, pp. 57–61.

[11] S. Seko, T. Yagi, M. Motegi, and S. Muto "Group recommendation using feature space representing behavioral tendency and power balance among members," in *Proc. RecSys*, Chicago, IL, USA, 2011, pp. 101–108.

[12] J. Castro, J. Lu, G. Zhang, Y. Dong, and L. Martínez, "Opinion dynamics-based group recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 12, pp. 2394–2406, Dec. 2018.

[13] J. Guo, Y. Zhu, A. Li, Q. Wang, and W. Han, "A social influence approach for group user modeling in group recommendation systems," *IEEE Intell. Syst.*, vol. 31, no.5, pp. 40–48, Sep./Oct. 2016.

[14] L. Sun, X. Wang, Z. Wang, H. Zhao, and W. Zhu, "Social-aware video recommendation for online social groups," *IEEE Trans. Multimedia*, vol. 19, no. 3, pp. 609–618, Mar. 2017.

[15] D. Qin, X. Zhou, L. Chen, G. Huang, and Y. Zhang, "Dynamic connection-based social group recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 3, pp. 453–467, Mar. 2020, doi: 10.1109/TKDE.2018.2879658.

[16] L. A. M. C. Carvalho and H. T. Macedo, "Users' satisfaction in recommendation systems for groups: An approach based on noncooperative games," in *Proc. WWW*, Rio de Janeiro, Brazil, 2013, pp. 951–958.

[17] X. Liu, Y. Tian, M. Ye, and W.-C. Lee, "Exploring personal impact for group recommendation," in *Proc. ICIKM*, Kuala Lumpur, Malaysia, 2012, pp. 674–683.

[18] Q. Yuan, G. Cong, and C.-Y. Lin, "COM: A generative model for group recommendation," in *Proc. SIGKDD*, New York, NY, USA, 2014, pp. 163–172.

[19] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and M. Zhou, "A deep latent factor model for high-dimensional and sparse matrices in recommender systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 7, pp. 4285–4296, Jul. 2021.

[20] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang, "Collaborative neural social recommendation," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 464–476, Jan. 2021.

[21] H. Zhu, Y. Ni, F. Tian, P. Feng, Y. Chen, and Q Zheng, "A group-oriented recommendation algorithm based on similarities of personal learning generative networks," *IEEE Access*, vol. 6, pp. 42729–42739, 2018.

[22] P.-P. Zhao *et al.*, "A generative model approach for geo-social group recommendation," *J. Comput. Sci. Technol.*, vol. 33, no. 4, pp. 727–738, Sep. 2018.

[23] T. V. Vo and H. Soh, "Generation meets recommendation: Proposing novel items for groups of users," in *Proc. RecSys*, Vancouver, BC, Canada, 2018, pp. 145–153.

[24] H. Wang and M. Dong, "Latent group recommendation based on dynamic probabilistic matrix factorization model integrated with CNN," *J. Comput. Res. Develop.*, vol. 54, no. 8, pp. 1852–1863, Aug. 2017.

[25] Z. Huang *et al.*, "Social group recommendation with TrAdaBoost," *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 5, pp. 1278–1287, Oct. 2020.

[26] L. Guo, H. Yin, T. Chen, X. Zhang, and K. Zheng, "Hierarchical hyperedge embedding-based representation learning for group recommendation," 2021, *arXiv:2103.13506*.

[27] G. Wang, X. Zhang, H. Wang, Y. Chu, and Z. Shao, "Group-oriented paper recommendation with probabilistic matrix factorization and evidential reasoning in scientific social network," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access, Apr. 29, 2021, doi: 10.1109/TSMC.2021.3072426.

[28] T. D. Q. Vinh, T.-A. N. Pham, G. Cong, and X.-L. Li, "Attention-based group recommendation," 2018, *arXiv:1804.04327*.

[29] D. Cao, X. He, L. Miao, Y. An, C. Yang, and R. Hong, "Attentive group recommendation," in *Proc. SIGIR*, Ann Arbor Michigan, MI, USA, 2018, pp. 645–654.

[30] D. Cao, X. He, L. Miao, G. Xiao, H. Chen, and J. Xu, "Social-enhanced attentive group recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 1195–1209, Mar. 2021.

[31] Z. Huang, X. Xu, H. Zhu, and M. Zhou, "An efficient group recommendation model with multiattention-based neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4461–4474, Nov. 2020.

[32] H. Yin, Q. Wang, K. Zheng, Z. Li, J. Yang, and X. Zhou, "Social influence-based group representation learning for group recommendation," in *Proc. ICDE*, Macao, China, 2019, pp. 566–577.

[33] P. Wang, L. Li, R. Wang, G. Xu, and J. Zhang, "Socially-driven multi-interaction attentive group representation learning for group recommendation," *Pattern Recognit. Lett.*, vol. 145, pp. 74–80, May 2021.

[34] S. Zan, Y. Zhang, X. Meng, P. Lv, and Y. Du, "UDA: A user-difference attention for group recommendation," *Inf. Sci.*, vol. 571, pp. 401–417, Sep. 2021.

[35] L. Chen, J. Cao, H. Chen, W. Liang, H. Tao, and G. Zhu, "Attentive multi-task learning for group itinerary recommendation," *Knowl. Inf. Syst.*, vol. 63, pp. 1687–1716, Apr. 2021.

[36] S. Ding, S. Hu, X. Li, Y. Zhang, and D. D. Wu, "Leveraging multimodal semantic fusion for gastric cancer screening via hierarchical attention mechanism," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access, Jul. 21, 2021, doi: 10.1109/TSMC.2021.3096974.

[37] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.

[38] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[39] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. WWW*, Perth, WA, Australia, 2017, pp. 173–182.

[40] J. Ni, Z. Huang, Y. Hu, and C. Lin, "A two-stage embedding model for recommendation with multimodal auxiliary information," *Inf. Sci.*, vol. 582, pp. 22–37, Jan. 2022.

[41] X. Luo, D. Wang, M. Zhou, and H. Yuan, "Latent factor-based recommenders relying on extended stochastic gradient descent algorithms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 2, pp. 916–926, Feb. 2021.

[42] Z. Huang, X. Xu, J. Ni, H. Zhu, and C. Wang, "Multimodal representation learning for recommendation in Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10675–10685, Dec. 2019.

[43] J.-T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *Proc. ICASSP*, Vancouver, BC, Canada, 2013, pp. 7304–7308.

[44] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Proc. NIPS*, Barcelona, Spain, 2016, pp. 136–144.

[45] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. SIGIR*, Paris, France, 2019, pp. 165–174.

[46] Z. Huang, G. Shan, J. Cheng, and J. Sun, "TRec: An efficient recommendation system for hunting passengers with deep neural networks," *Neural Comput. Appl.*, vol. 31, no. 1, pp. 209–222, Jan. 2019.

[47] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. AISTATS*, Sardinia, Italy, 2010, pp. 249–256.

[48] I. Karabayir, O. Akbilgic, and N. Tas, "A novel learning algorithm to optimize deep neural networks: Evolved gradient direction optimizer (EVGO)," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 2, pp. 685–694, Feb. 2021.

[49] T.-A. N. Pham, X. Li, G. Cong, and Z. Zhang, "A general graph-based model for recommendation in event-based social networks," in *Proc. ICDE*, Seoul, South Korea, 2015, pp. 567–578.
[50] Z. Li, W. Wei, T. Zhang, M. Wang, S. Hou, and X. Peng, "Online multi-expert learning for visual tracking," *IEEE Trans. Image Process.*, vol. 29, pp. 934–946, 2020.
[51] Z. Huang *et al.*, "Making accurate object detection at the edge: Review and new approach," *Artif. Intell. Rev.*, to be published, doi: 10.1007/s10462-021-10059-3.
[52] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, Feb. 2012.
[53] A. Lipani, D. E. Losada, G. Zuccon, and M. Lupu, "Fixed-cost pooling strategies," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1503–1522, Apr. 2021.

Zhenhua Huang received the Ph.D. degree in computer science from Fudan University, Shanghai, China, in 2008.

He is currently a Professor with the School of Computer Science, South China Normal University, Guangzhou, China. Since 2004, he has published three books and more than 100 papers in various journals and conference proceedings. His research interests mainly include deep learning, recommendation system, data mining, knowledge discovery, and big data.

Yajun Liu received the B.S. degree in computer science from the Nanchang Institute of Technology, Nanchang, China, in 2019. He is currently pursuing the master's degree in computer science with South China Normal University, Guangzhou, China.

His main research interests include recommendation system, big data, deep learning, and data mining.

Choujun Zhan received the B.S. degree in automatic control engineering from Sun Yat-sen University, Guangzhou, China, in 2007, and the Ph.D. degree in electronic engineering from the City University of Hong Kong, Hong Kong, in 2012.

After graduation, he worked as a Postdoctoral Fellow with the Hong Kong Polytechnic University, Hong Kong. Since Fall 2016, he has been an Associate Professor with the Department of Electronic Communication and Software Engineering, Nanfang College of Sun Yat-sen University, Guangzhou. He is currently a Professor with the School of Computer Science, South China Normal University, Guangzhou. His research interests include complex networks, time-series modeling and prediction, epidemic spreading, information diffusion, and machine learning.

Chen Lin received the B.Eng. and Ph.D. degrees in computer science from Fudan University, Shanghai, China, in 2004 and 2010, respectively.

She is currently a Professor with the School of Information Science and Technology, Xiamen University, Xiamen, China. Her research interests mainly include Web mining and recommender systems.

Weiwei Cai (Member, IEEE) is currently pursuing the master's degree with the Central South University of Forestry and Technology, Changsha, China, and studying with the Northern Arizona University, Flagstaff, AZ, USA.

Prior to that, he worked with IT industry for more than ten years in the roles of an System Architect and the Program Manager. His research interests include machine learning, deep learning, and computer vision.

Mr. Cai is an Associate Editor of *ASP Transactions on Pattern Recognition and Intelligent Systems*. He has also served as a Guest Editor for the *Wireless Communications and Mobile Computing*, *Frontiers in Psychology*, *Computer Modeling in Engineering & Sciences*, and *Journal of Healthcare Engineering*.

Yunwen Chen received the Ph.D. degree in computer science from Fudan University, Shanghai, China, in 2008.

He is the Founder and a CEO of DataGrand Inc., Shenzhen, China, a leading AI company in China. He had been the Chief Data Officer of Shanda Inc., Shanghai, China, the Senior Director of Tencent Inc., Shenzhen, and a Researcher of Baidu Inc., Beijing, China. He has 32 patents and many academic publications. His research interests include data mining, natural language processing, deep learning, knowledge graph, and recommendation system.

Dr. Chen is a member of ACM and a Senior Member of CCF.