Bundle Recommendation and Generation With Graph Neural Networks

Jianxin Chang, Chen Gao, Xiangnan He, Depeng Jin, *Member, IEEE*, and Yong Li, *Senior Member, IEEE*

Abstract—Bundle recommendation aims to recommend a bundle of items for a user to consume as a whole. Related work can be divided into two categories: 1) to recommend the platform's prebuilt bundles to users; 2) generate personalized bundles for users. In this work, we propose two graph neural network models, a BGCN model (short for *Bundle Graph Convolutional Network*) for prebuilt bundle recommendation, and a BGGN model (short for *Bundle Graph Generation Network*) for personalized bundle generation. First, BGCN unifies the user-item interaction, the user-bundle interaction and the bundle-item affiliation into a heterogeneous graph. With item nodes as the bridge, graph convolutional propagation between user and bundle nodes makes the learned representations capture the item-level semantics. Second, BGGN re-constructs bundles into graphs based on the item co-occurrence pattern and the user's supervision signal. The complex and high-order item-item relationships in the bundle graph are explicitly modeled through graph generation. Empirical results demonstrate the substantial performance gains of BGCN and BGGN, which outperforms the state-of-the-art baselines by 10.77% to 23.18% and 20.90% to 64.52%, respectively. We have released the datasets and codes at this link: https://github.com/cjx0525/BGCN.

Index Terms—Bundle recommendation, bundle generation, graph generation, graph convolutional networks

1 Introduction

A BUNDLE is multiple items that are similar or complementary to meet users' complex needs. Therefore, a good bundle can improve the user experience by satisfying multiple interests of users at the same time. Besides, it can also increase business sales by setting overall discounts to attract users to interact with larger orders. The prevalence of bundled items on e-commerce and content platforms makes bundle recommendation an important task.

However, the substantial increase of users and items makes bundle recommender systems still face non-trivial challenges. Since a bundle is composed of multiple items, users will consider more factors when purchasing the bundle: 1) the attractiveness of a bundle depends on its items; 2) the users need to be satisfied with most items in a bundle; 3) the semantics expressed by the bundle will affect the user's choice. Also, as a new form of interaction, the sparseness between the user and the bundle brings greater difficulties to infer the user's preference and interest for bundles.

Existing bundle recommendation works can be divided into two categories: 1) to recommend pre-built bundles and 2) to recommend generated bundles. For simplicity, we refer to the former as bundle recommendation and the latter as bundle generation. In the following sections, bundle recommendation refers to prebuilt bundle recommendation (to recommend prebuilt bundles) and bundle generation refers to generated bundle recommendation (to generate bundles first and then recommend them), unless otherwise specified. in this paper. It is worth mentioning that both of them have wide applications in real-world systems. For example, Apple can pre-build one iPhone and AirPods in a bundle; Amazon can generate a bundle that contains one iPhone and one third-party earphone in a bundle. There are essential differences between the two categories. Bundle recommendation is to predict the user's preference for prebuilt bundles based on their interacted bundles. Its main challenge is to reduce the impact of bundle interaction's sparseness, so it is necessary to additionally use the user's interaction information on the item and the affiliation information of the bundle and the item as an auxiliary. Bundle generation is to generate a bundle that maximizes the user's preference according to the user's preference for the existing bundle. To effectively supervise the quality of the generated bundle, the user's interaction with the bundle is more important to explore which items are more suitable for bundling.

Bundle Recommendation. Most existing works for bundle recommendation [1], [2], [3] regard item and bundle

- *Jianxin Chang, Chen Gao, Depeng Jin, and Yong Li are with the Beijing National Research Center for Information Science and Technology, Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. E-mail: changjx18@mails.tsinghua.edu.cn, chgao96@gmail.com, {jindp, liyong07}@tsinghua.edu.cn.*
- *Xiangnan He is with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230052, China. E-mail: xiangnanhe@gmail.com.*

Manuscript received 12 Nov. 2020; revised 12 July 2021; accepted 11 Sept. 2021. Date of publication 22 Sept. 2021; date of current version 3 Feb. 2023. This work was supported in part by The National Key Research and Development Program of China under Grant 2020AAA0106000, in part by the National Natural Science Foundation of China under Grants U1936217, 61971267, 61972223, 61941117, and 61861136003, in part by Beijing Natural Science Foundation under Grant L182038, in part by the Beijing National Research Center for Information Science and Technology under Grant 20031887521, and in part by the National Natural Science Foundation of China Grant U19A2079.
(Corresponding author: Chen Gao.)
Recommended for acceptance by T. Weninger.
Digital Object Identifier no. 10.1109/TKDE.2021.3114586

1041-4347 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.
Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

recommendation as two separate tasks, and associate them by sharing model parameters. A recent study [4] proposed a multi-task framework that transfers the benefits of the item recommendation task to the bundle recommendation to alleviate the scarcity of user-bundle interactions. Despite effectiveness, they suffer from three major limitations:

- *Separated modeling of two affiliated entities.* Parameters sharing does not explicitly model the relationship between users, items and bundles, and the combination of multi-task way is difficult to balance the weights of the main task and auxiliary task.
- *Substitution of bundles is not considered.* Existing works only consider the correlation between items in a bundle to enhance the item task. However, the association between the bundles as the recommended target is more critical.
- *Decision-making is ignored when users interact with bundles.* At the item-level, even though a user likes most items in a bundle, but may refuse the bundle because of the existence of one disliked item. At the bundle level, for two highly similar bundles, the key to the user's final selection is their non-overlapping parts.

To address these limitations, we propose a solution named *Bundle Graph Convolutional Network* (BGCN). Utilizing the strong power of graph neural networks in learning from complicated topology and higher-order connectivity, our BGCN effectively incorporates item-awareness into bundle recommender as follows, a) It unifies user nodes, item nodes and bundle nodes into a heterogeneous graph, propagating embeddings between users and bundles with items as the bridge; b) The bundle-item-bundle meta-path is built on the graph to capture an alternative relationship between bundles; c) Through training with hard-negative samples, the user's decision-making when choosing bundles is further explored.

Bundle Generation. There are currently only a few bundle generation works, and most of them focus on processing a single bundle generation[3], [5]. The most recent work proposes a bundle list generation solution[6]. It formalizes bundle generation as a structured prediction problem. Specifically, it structures the bundle into a sequence and sorts the internal items by price. Despite effectiveness, they suffer from three major limitations:

- *The generation is affected by the sequential ordering.* The sequence-based method requires that items be sorted by importance in advance. Price is a good indicator of importance on e-commerce platforms, but it cannot be used for content platforms. And two items related in the bundle could be far apart in the sequence generation process.
- *Only smaller bundles can be generated.* Due to the long-term bottleneck of the sequence-based method, such as RNN and LSTM, it tends to generate short sentences. Therefore, it is more challenging to capture the correlation between a large number of items. This means that it cannot generate large-scale bundles well.
- *Unable to distinguish high-order associations between items.* Not all items in a bundle are directly related. Two items may be connected through another item, such as an iPhone bundle, "Phone Case" and "Airpods" are related through "iPhone". The sequence cannot clearly indicate the direct and indirect relationships between items.

To address these limitations, we propose a solution named *Bundle Graph Generation Network* (BGGN). Utilizing the strong power of graph neural networks in learning from complicated topology and higher-order connectivity, our BGGN explicitly quantifies the direct and indirect relationships between items in the bundle as follows, a) It uses the co-occurrence relationship of items to re-construct the bundle into a graph, and gradually generates the graph in an auto-regressive manner; b) In each generation step, the next item most likely to be closely related to the generated bundle is predicted through the embedded propagation between the generated graph and the retrieved nodes; c) Through the diversified beam search on the graph, the generation process's robustness and the generated results' diversity are increased.

To summarize, the main contributions of this work are as follows.

- To the best of our knowledge, we are the first to propose graph neural network models to solve the problem of prebuilt bundle recommendation and personalized bundle generation. For the prebuilt bundle recommendation, we unify the user, item and bundle into a heterogeneous graph to explicitly model their complex relationships. For the personalized bundle generation, we re-construct the bundle into graphs to distinguish the complex associations between its internal items.
- In the recommender, we develop embedding propagation at two levels on the constructed graph, which distinguish affiliation between item and bundle nodes to learn the representation of users and bundles with item-level information. Training with hard negatives further explores the fine-grained differences between bundles. In the generator, we develop an autoregressive generation algorithm that can generate graphs with entity difference nodes. In each step, the embedding propagation on the augmented graph can select the current optimal item from the retrieved nodes and associate it with the generated graph. The diversified beam search on the graph ensures robustness while increasing diversity.
- Extensive experiments on two real-world datasets show that our proposed recommendation method outperforms existing state-of-the-art baselines by 10.77% to 23.18%, and our proposed generation method outperforms existing state-of-the-art baselines by 20.90% to 64.52%.

Differences With Conference Version. The preliminary conference version of this work has been published in SIGIR-2020 [7]. Comparing with the previous version, we further explore the second category of task, generating personalized bundles. We further proposed an effective model and extensive experiments reflect the superiority of the new proposed BGGN, which can be regarded as an extension of BGCN.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

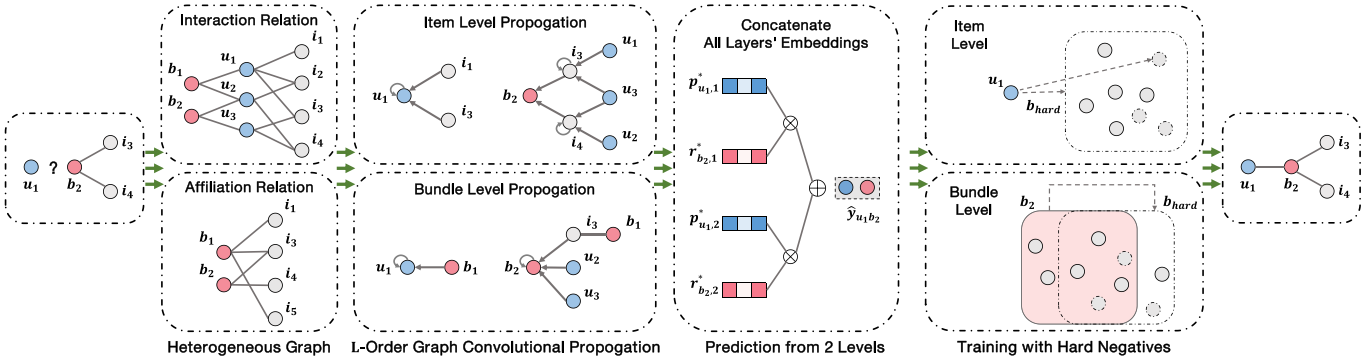


Fig. 1. The illustration of BGCN model, where u_1 is the target user and b_2 is the target bundle. For a clear display, we divide the unified heterogeneous graph into two kinds of relationships and draw them separately. Best view in color.

2 Problem Formulation

In order to integrate item-level information to improve bundle recommendation accuracy, there are two types of important side information that need to be modeled, the user's preference to item and bundle's composition. We use $\mathcal{U}$, $\mathcal{B}$ and $\mathcal{I}$ to denote the set of users, bundles and items, and define user-bundle interaction matrix, user-item interaction matrix, and bundle-item affiliation matrix as $\mathbf{X}_{M\times N} = \{\mathbf{x}_{ub}|u \in \mathcal{U}, b \in \mathcal{B}\}$, $\mathbf{Y}_{M\times O} = \{\mathbf{y}_{ui}|u \in \mathcal{U}, i \in \mathcal{I}\}$, and $\mathbf{Z}_{N\times O} = \{\mathbf{z}_{bi}|b \in \mathcal{B}, i \in \mathcal{I}\}$ with a binary value at each entry, respectively. An observed interaction $\mathbf{x}_{ub}$ means user u once interacted bundle b, and an observed interaction $\mathbf{y}_{ui}$ means user u once interacted item i. Similarly, an entry $\mathbf{z}_{bi} = 1$ means bundle b contains item i. M, N and O denote the number of users, bundles and items, respectively. Based on the above definition, *the problem of bundle recommendation* is then formulated as follows:

Input. user-bundle interaction data $\mathbf{X}_{M\times N}$, user-item interaction data $\mathbf{Y}_{M\times O}$, and bundle-item affiliation data $\mathbf{Z}_{N\times O}$.

Output. A recommendation model that estimates the probability that user u will interact with bundle b.

Unlike the bundle recommendation, bundle generation focuses on mining the associations between items in existing bundles, and u-i interaction may cause negative migration. And for a fair comparison with the baseline, we have not considered u-i interaction in the bundle generation. Based on the above definition, *the problem of bundle generation* is then formulated as follows:

Input. user-bundle interaction data $\mathbf{X}_{M\times N}$, and bundle-item affiliation data $\mathbf{Z}_{N\times O}$.

Output. A generation model that selects the most likely items to be interacted with user u simultaneously to form a bundle.

3 Methods

3.1 Our BGCN Model for Bundle Recommendation

Fig. 1 illustrates our proposed BGCN model which is made up of the following three parts.

- *Heterogeneous Graph Construction.* We explicitly model the interaction and affiliation between users, bundles, and items by unifying them into a heterogeneous graph.
- *Levels Propagation.* The propagation at two levels on the constructed graph can capture the CF signal between users and items/bundles, the semantics of bundles and the alternatives between bundles through distinguishing the affiliation relationship between item and bundle nodes.
- *Training with Hard Negatives.* Considering the user's caution when choosing a bundle, the hard negatives further encode fine-grained representations of the user and the bundle.

3.1.1 Heterogeneous Graph Construction

To explicitly model the relationship between users, bundles, and items, we first build a unified heterogeneous graph. The interaction and affiliation data can be represented by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where nodes are $\mathcal{V}$ consisting of user nodes $u \in \mathcal{U}$, bundle nodes $b \in \mathcal{B}$ and item nodes $i \in \mathcal{I}$, and edges are $\mathcal{E}$ consisting of user-bundle interaction edges (u, b) with $\mathbf{x}_{ub} = 1$, user-item interaction edges (u, i) with $\mathbf{y}_{ui} = 1$, and bundle-item affiliation edges (b, i) with $\mathbf{z}_{bi} = 1$. The second block in Fig. 1 illustrates our constructed graph.

For user, item and bundle nodes on the constructed graph, we apply one-hot encoding to encode the input and compress them to dense real-value vectors as follows:

$$\mathbf{p}_u = \mathbf{P}^T\mathbf{v}_u^U, \quad \mathbf{q}_i = \mathbf{Q}^T\mathbf{v}_i^I, \quad \mathbf{r}_b = \mathbf{R}^T\mathbf{v}_b^B, \tag{1}$$

where $\mathbf{v}_u^U$, $\mathbf{v}_i^I$, $\mathbf{v}_b^B \in \mathbb{R}^N$ denotes the one-hot feature vector for user u, item i, and bundle b. $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{R}$ denote the matrices of user embedding, item embedding, and bundle embedding, respectively.

3.1.2 Item Level Propagation

The user's preference for an item in the bundle can attract the user's attention and interest to this bundle. Since the bundled items are often carefully designed, they are often compatible with each other in function and compose some semantics to affect the user's selection context. For example, the bundle with a mattress and a bed frame reflects the meaning of bedroom furniture, and the bundle with a suit and a tie reflects the meaning of workplace dressing.

To capture user preferences for the item and characteristic of the item itself, we build upon an embedded propagation layer between user and item. Then information pooling

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

from item to bundle can seize the semantic information of the bundle from the item-level. The propagation-based and pooling-based embedding updating rules for user u, item i and bundle b can be formulated as follows:

$$\begin{aligned}
\mathbf{p}_{u,1}^{(\ell+1)} &= \sigma\Big(W_1^{(\ell+1)}\Big(\mathbf{p}_{u,1}^{(\ell)} + aggregate\Big(\mathbf{q}_{i,1}^{(\ell)}|i \in \mathcal{N}_u\Big)\Big) + b_1^{(\ell+1)}\Big),\\
\mathbf{q}_{i,1}^{(\ell+1)} &= \sigma\Big(W_1^{(\ell+1)}\Big(\mathbf{q}_{i,1}^{(\ell)} + aggregate\Big(\mathbf{p}_{u,1}^{(\ell)}|u \in \mathcal{N}_i\Big)\Big) + b_1^{(\ell+1)}\Big),\\
\mathbf{r}_{b,1}^{(\ell+1)} &= aggregate\Big(\mathbf{q}_{i,1}^{(\ell+1)}|i \in \mathcal{N}_b\Big),\\
\mathbf{p}_{u,1}^{(0)} &= \mathbf{p}_u, \mathbf{q}_{i,1}^{(0)} = \mathbf{q}_i,
\end{aligned} \tag{2}$$

where W_1 is learned weight, b_1 is learned bias, σ is non-linear activation function $LeakyReLU$. $\mathcal{N}_u, \mathcal{N}_i, \mathcal{N}_b$ represent neighbors of user u, item i and bundle b, respectively. We use the simple mean function as the aggregation function and leave other more complex functions for future exploration. Through the special propagation mechanism, the impact of bundle data sparsity can be weakened and the cold start capability of the model may be improved naturally.

3.1.3 Bundle Level Propagation

The close association between items in a bundle makes two bundles that share some items very similar. The degree of similarity can be inferred from how many items they share. For example, computer sets that share five components are closer in performance than two, and movie lists that share ten films are closer in theme than five. For users, such bundles that share more items can often be considered at the same time.

We design a bundle-to-user embedding propagation module to learn preference for bundles from the bundle level. Then, a user-to-bundle embedding propagation is performed to extract bundle overall attributes. Because highly overlapping bundles exhibit similar patterns in attracting users, we develop weighted propagation based on the degree of overlap on the bundle-item-bundle meta-path to seize alternative relationships between bundles. The embedding updating rules at the bundle level can be formulated as follows:

$$\begin{aligned}
\mathbf{p}_{u,2}^{(\ell+1)} &= \sigma\Big(W_2^{(\ell+1)}\Big(\mathbf{p}_{u,2}^{(\ell)} + aggregate\Big(\mathbf{r}_{b,2}^{(\ell)}|b \in \mathcal{N}_u\Big)\Big) + b_2^{(\ell+1)}\Big),\\
\mathbf{r}_{b,2}^{(\ell+1)} &= \sigma\Big(W_2^{(\ell+1)}\Big(\mathbf{r}_{b,2}^{(\ell)} + aggregate\Big(\mathbf{p}_{u,2}^{(\ell)}|u \in \mathcal{N}_b\Big)\\
&\quad + aggregate\Big(\beta_{bb'} \cdot \mathbf{r}_{b',2}^{(\ell)}|b' \in \mathcal{M}_b\Big)\Big) + b_2^{(\ell+1)}\Big),\\
\mathbf{p}_{u,2}^{(0)} &= \mathbf{p}_u,\ \mathbf{r}_{b,2}^{(0)} = \mathbf{r}_b,
\end{aligned} \tag{3}$$

where W_2 and b_2 are the trainable transformation matrix and bias, respectively. $\mathcal{M}_b$ represents neighbors of bundle b on the bundle-item-bundle meta-path. $\beta_{bb'}$ represents the overlap intensity between bundles after normalization. Propagation of attributes expressed by similar b helps bundles learn better representations and further enhance message-passing between u and b.

3.1.4 Prediction

After we iteratively do such propagation for L times, we obtain L user/bundle embeddings. We concatenate all layers' embeddings to combine the information received from neighbors of different depths for prediction

$$\begin{aligned}
\mathbf{p}_{u,1}^{*} &= \mathbf{p}_{u,1}^{(0)}||\cdots||\mathbf{p}_{u,1}^{(L)}, \mathbf{r}_{b,1}^{*} = \mathbf{r}_{b,1}^{(0)}||\cdots||\mathbf{r}_{b,1}^{(L)},\\
\mathbf{p}_{u,2}^{*} &= \mathbf{p}_{u,2}^{(0)}||\cdots||\mathbf{p}_{u,2}^{(L)}, \mathbf{r}_{b,2}^{*} = \mathbf{r}_{b,2}^{(0)}||\cdots||\mathbf{r}_{b,2}^{(L)}.
\end{aligned} \tag{4}$$

Finally, we use the inner product for final prediction and combine bundle and item-levels as follows:

$$\hat{y}_{ub} = {\mathbf{p}_{u,1}^{*}}^{\top}\mathbf{r}_{b,1}^{*} + {\mathbf{p}_{u,2}^{*}}^{\top}\mathbf{r}_{b,2}^{*}. \tag{5}$$

3.1.5 Training With Hard Negatives

Because bundles contain more items and have higher prices, users are often cautious when making decisions or spending money in bundle scenarios to avoid unnecessary risks. For example, even though a user likes most items in a bundle, but may refuse the bundle because of the existence of one disliked item. For two highly similar bundles, the key to the user's final selection is their non-overlapping parts.

To optimize our BGCN model and take into account the user's decision-making when interacting with the bundle, we design a learning strategy based on hard-negative samples. First, we adopt a pairwise learning manner that is widely used in implicit recommender systems [8]. Then after the model converges, the hard negative samples are introduced with a certain probability for more detailed training. Thus, we define the objective function as follows:

$$\text{Loss} = \sum_{(u,b,c)\in\mathcal{Q}} -\ln\sigma(\hat{y}_{ub} - \hat{y}_{uc}) + \lambda \cdot \|\Theta\|^2, \tag{6}$$

where $\mathcal{Q} = \{(u,b,c)|(u,b) \in \mathcal{Y}^+, (u,c) \in \mathcal{Y}^-\}$ denote the pairwise training data with negative sampling. $\mathcal{Y}^+$ and $\mathcal{Y}^-$ denote the observed and unobserved user-bundle interaction, respectively. In the hard-negative sampler, for every $(u,b) \in \mathcal{Y}^+$, $c \in \mathcal{Y}^-$ is the bundle that u has not interacted with but interacted with most of its internal items or that overlaps with b, as shown in the fifth block in Fig. 1. To prevent over-fitting, we adopt L2 regularization where Θ stands for model parameters and λ controls the penalty strength.

3.2 Our BGGN Model for Bundle Generation

Fig. 2 illustrates our proposed BGGN model which is made up of the following three parts.

- *Bundle Graph.* We re-construct the bundle as bundle graphs and formalized the bundle list generation as graph generation problem.
- *Nodes Retrieval.* To decrease the space complexity caused by generating graphs with entity differences between nodes, we first retrieve some items that the user is interested in as candidate nodes for the graph to be generated.
- *Edges Construction.* We adopt an auto-regressive method to gradually select an item node from candidate nodes to construct edges between this item node and the existing bundle graph. The model learns the complex and high-order relationships in the bundle graph through GCN [9].

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

- *Search and Stop.* We propose a diversified beam search method on the graph to reduce the impact of item-order and balance accuracy and diversity. We design a stopping criterion based on graph's density changes to adaptively meet different requirements of the bundle size.

3.2.1 Bundle Graph

Unlike the bundle recommendation task, we cannot generate a bundle by predicting the link between i and b. Because the bundle is the target of generation, its entity representation cannot be determined in advance. The current state-of-the-art bundle generation method[6] structure the bundle into a sequence, but the sequence-based method has a long-term bottleneck. It cannot model the long-range dependence between items with longer distances in the sequence. It may not work when the sequence is long, i.e., it cannot be used on large-sized bundles.

We re-construct the bundle as a graph and use graph convolutional neural networks to generate graphs to solve the sequence-based method's limitations. A simple idea is to construct the bundled items as a complete graph, but this is inefficient and unreasonable under a large-size bundle. First, the number of edges of the complete graph will take up a lot of memory; second, it is not that any two items in the bundle have the same degree of correlation. Two items may be connected through another item, such as a Marvel movie bundle, "Iron Man" and "Thor" are related through "Avengers", and an iPhone bundle, "Phone Case" and "Airpods" are related through "iPhone".

To explicitly distinguish the above-mentioned high-order correlations between items, we construct bundle graphs based on the user's supervision signal. First, we calculate the degree of association between item pairs based on the item co-occurrence frequency and bundle popularity. Second, we use Bernoulli distribution to convert each item's correlation degree within the bundle to a value of $0/1$, determining whether there are directly connected edges between items. The formula for calculating the degree of correlation between item i and j is as follows:

$$\mu_{ij} = \frac{\sum_{b=1}^{N} \mathbf{z}_{bi} * \mathbf{z}_{bj} * \left(\sum_{u=1}^{M} \mathbf{x}_{ub}/M\right)}{N}. \tag{7}$$

3.2.2 Nodes Retrieval

The essence of the graph generation task is to predict possible element 1 in the adjacency matrix. Traditional graph generation methods are mostly used in drug molecules and protein networks, and their nodes (atoms and amino acids) are not different. The dimensions of the adjacency matrix of these graphs are close to the number of their nodes. However, in the recommendation scenario where items are regarded as nodes, they have a difference in physical meaning. If all items are considered as candidates for the bundle, the matrix with hundreds of thousands of dimensions will cause huge consumption.

We learn from the matching stage in the recommended field. Specifically, in the training process, we design the node loss considering user preferences. First, we perform sampling to obtain negative sample items with the same number of items in the bundle. Then, we use the BPR learning framework [8] to make the user's preference score for the items in the bundle higher than the items obtained by sampling. The formula is as follows:

$$\mathrm{L}_{\mathrm{pref}} = \sum_{(u,i,c)\in\mathcal{P}} -\ln\sigma(\mathbf{p}_u^\top \mathbf{q}_i \ - \mathbf{p}_u^\top \mathbf{q}_j), \tag{8}$$

where $\mathcal{P} = \{(u,i,j)|i \in b, j \notin b, (u,b) \in \mathcal{Z}^+\}$. $\mathcal{Z}^+$ denotes the observed user-bundle interaction. i and j denote the observed and unobserved item in bundle b, respectively.

In the generating process, we first retrieve the items that the user cares about as candidate nodes to form an adjacency matrix of acceptable size, and then we retrieve matching items from them for bundling. The set of retrieved items $\mathrm{I_u} \in \mathbb{R}^\eta$ and the user's preference score for these items $\mathrm{S_u} \in \mathbb{R}^\eta$ are calculated using the following formula, where η denotes the number of retrieved items

$$\mathrm{I_u} = \arg\max_i(\mathbf{p}_u^\top \mathbf{q}_i, \eta), i \in \mathcal{I}, \tag{9}$$

$$\mathrm{S_u} = \max_i(\mathbf{p}_u^\top \mathbf{q}_i, \eta), i \in \mathcal{I}. \tag{10}$$

3.2.3 Edges Construction

To distinguish the fine-grained correlation between items to form a more appropriate bundle, inspired by [10], we use GCN(Graph Convolutional Network) to learn the structural information and high-order item-item relationships in the bundle graph. Specifically, we use the generated graph as a condition and adopt an auto-regressive method to gradually select nodes from the retrieved items to construct edges. We generate an adjacency matrix A_b corresponding to the graph from the retrieved node I_u of interest for each user. Due to the undirected graph's symmetry, we can only model the lower triangular part of its adjacency matrix A_b, denoted as L_b. The third block in Fig. 2 represents each generation step of the edge construction process.

The graph generation method in traditional application scenarios does not need to consider the differences of nodes, adding nodes to the graph at each generation step, that is, adding entries to the adjacency matrix. However, items as nodes have physical differences in the recommendation scenario. Our method first searches for the optimal entity node from retrieved nodes at each step and then replaces entries in the adjacency matrix.

Initialization. Intuitively, we initialize $L_b \in \mathbb{R}^{\eta\times\eta}$, whose dimension is consistent with the number of retrieved items, and each row of L_b corresponds to an item entity. Our method generates entries in certain rows of L_b at a time and swaps the row where the best entry is generated with the row corresponding to the current step. This generates a series of vectors $\{L_{b,i}|i = 1, \ldots, \eta\}$, where $L_{b,i} \in \mathbb{R}^{1\times\eta}$ is the ith row of L_b padded with zeros. The process continues until the stop generation condition (i.e., step T) is reached and L_b is obtained. Further, we can obtain $A_b = L_b + L_b^T$.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

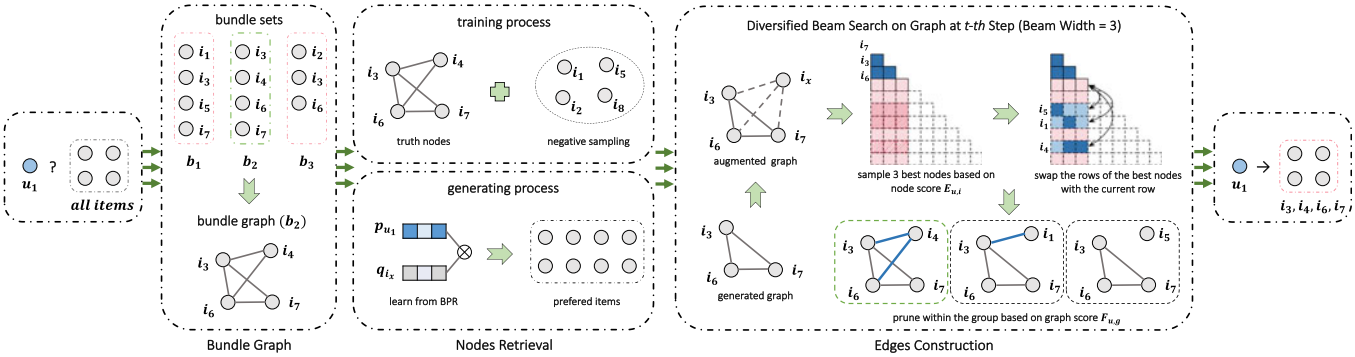


Fig. 2. The illustration of BGGN model, where u_1 is the target user and $\{i_3, i_4, i_6, i_7\}$ are the items in the bundle that the user has interacted with. For a clear display, we draw the two processes in node retrieval separately. Best view in color.

We factorize the probability of generating L_b as

$$p(L_b) = \prod_{t=1}^{T} p(L_b^t | L_b^1, \ldots, L_b^{t-1}). \tag{11}$$

The conditional distribution $p(L_b^t | L_b^1, \ldots, L_b^{t-1})$ defines the probability of generating the current item (all edges from the current node to the earlier generated nodes) based on the existing graph.

Embedding Layer. We represent the retrieved items embedding as the initial node representation of the graph

$$\mathbf{h}_i = \mathbf{H}^T \mathbf{v}_i^{I_u}. \tag{12}$$

Among them, $\mathbf{v}_i^{I_u}$ denotes the one-hot feature vector for item i in the retrieved sets I_u of user u. $\mathbf{H}$ represents another item embedding matrix and is used to capture the correlation between items, which is different from the previous $\mathbf{Q}$ used to capture user preferences.

Refine Embedding. According to these node representations, we use GCN to generate edges associated with the current graph. At the tth generation step, we have $(t-1)$ nodes and the edges between them. We construct a temporarily augmented graph $\tilde{G}_{b,t}$, which contains the generated subgraph of $(t-1)$ nodes, the edges between these nodes, and all unused retrieval nodes. For these $(\eta - t + 1)$ new nodes, we add edges to connect them with the previous $(t-1)$ node, which is the *augmented edges* [10].

Then, we use the GNN framework similar to [10] on this augmented graph $G_{b,t}$ to obtain the updated node representation that encodes the graph structure. More concretely, the lth layer of message passing in the GNN breaks down into steps as listed below:

$$\mathbf{m}_{ij}^{\ell} = f(\mathbf{h}_i^{\ell} - \mathbf{h}_j^{\ell}), \tag{13}$$

$$\mathbf{a}_{ij}^{\ell} = \mathrm{Sigmoid}\Big(e(\mathbf{h}_i^{\ell} - \mathbf{h}_j^{\ell})\Big), \tag{14}$$

$$\mathbf{h}_i^{\ell+1} = \mathrm{GRU}(\mathbf{h}_i^{\ell}, \textstyle\sum_{j \in \mathcal{N}(i)} \mathbf{a}_{ij}^{\ell} \mathbf{m}_{ij}^{\ell}). \tag{15}$$

Here $\mathbf{h}_i^{\ell}$ is the hidden representation for node i after layer ℓ, and $\mathbf{m}_{ij}^{l}$ is the message vector from node i to j. $\mathbf{a}_{ij}^{l}$ is an attention weight associated with edge (i, j). Both the message function f and the attention function e are implemented as 2-layer MLPs with ReLU nonlinearities. Finally, the node representations are updated through a GRU similar to[11] after aggregating all the incoming messages through an attention-weighted sum over the neighborhood $\mathcal{N}(i)$.

Edges Probability. After L layers of message passing, we obtain the final node representation vectors $\mathbf{h}_i^L$ for each node i, and then model the probability of generating edges between each retrieved node and the current graph via the mixture of Bernoulli distributions

$$p(L_{b,i}^t | L_b^1, \ldots, L_b^{t-1}) = \sum_{\phi=1}^{\Phi} \alpha_\phi \prod_{j \in J_{u,b}} \theta_{\phi,i,j}, \tag{16}$$

$$\alpha_1, \ldots, \alpha_\Phi = \mathrm{Softmax}\Big(\textstyle\sum_{j \in J_{u,b}} \mathrm{MLP}_\alpha(h_i^L - h_j^L)\Big), \tag{17}$$

$$\theta_{1,i,j}, \ldots, \theta_{\Phi,i,j} = \mathrm{Sigmoid}\Big(\mathrm{MLP}_\theta(h_i^L - h_j^L)\Big), \tag{18}$$

where both MLP_α and MLP_θ contain two hidden layers with ReLU nonlinearities and have Φ-dimensional outputs. Here Φ is the number of mixture components. $J_{u,b}$ is the node-set of the currently generated bundle graph.

3.2.4 Search and Stop

Traditional graph generation predicts the adjacency matrix row by row, but the generated graph will be affected by the initial node order in the adjacency matrix. To add the current optimal item to the bundle in the recommendation scenario, we designed the beam search method on the graph. It is a kind of greedy search algorithm with a local goal rather than a global optimization goal. The search goal of each step is to maximize the current objective (determined by node score and graph score), which can both largely reduce the search cost and obtain promising search results.

Node Score. For the node score, it refers to the score of the searched new item. It is made up of two parts: matching score between the new item and the current user; matching score between the new item and the other items. As mentioned above, we construct the augmented edges between all remaining nodes and the generated graph in the t-th generation step (The red shaded part in the third block in Fig. 2). The probabilities of having edges between each remaining nodes and the existing nodes are obtained by

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

performing a graph convolutional propagation. We also consider the user's preference for these items and define the following node score $E_{u,i}$

$$E_{u,i} = S_{u,i} + p(L_{b,i}^t | L_b^1, \ldots, L_b^{t-1}). \tag{19}$$

We use this score as the weight of sampling from retrieved nodes, reserve ω nodes for each graph, and combine them with G_b^{t-1} respectively as the $(t+1)th$ node, and ω represents the beam width (When $\omega = 1$, beam search degenerates into greedy search).

Graph Score. For the graph score, it refers to the score of the updated graph with the new item added. It is made up of two parts: matching score between the updated graph and the current user; matching score between all item pairs in the updated graph (we name it as the density). Larger density here means items are more related and thus a denser graph is more likely to be a better bundle. For the probability $p(L_{b,i}^t | L_b^1, \ldots, L_b^{t-1})$ of the edge corresponding to the $(t+1)th$ node, we generate entries in the adjacency matrix according to the Bernoulli distribution (The blue part in the third block in Fig. 2). Then we swap the corresponding row τ of the node in the adjacency matrix with the current row t (The arrow in the third block in Fig. 2). So far, we have obtained $\omega \times \omega$ temporary subgraphs $G_b^{t-1,i}$. Further, we consider the sum of the density of the subgraph and the user's preference for nodes in the subgraph as the subgraph score $F_{u,b}$ as follows:

$$F_{u,b} = \sum_{j \in J_{u,b}} S_{u,j} + density\big(G_b^{t-1}\big). \tag{20}$$

Then we select the ω subgraphs with the largest weights to prune the search tree and enter the next generation step. In short, the basic search strategy is as: we first choose K*K items according to the node score as the search space, and then choose the top-K ones with the highest graph scores to prune the search space.

Diversified Search. However, the above beam search mechanism tends to generate similar bundles. It is also necessary to control cross-bundle diversity between the generated bundles, which can prevent the model from generating bundles with only high-scoring items. At the same time, it can evaluate the coverage of the items in the platform. Inspired by the Diversified Beam Search[12], we propose a Diversified Beam Search on Graph that decodes diverse bundles by dividing the beam budget into groups and enforcing diversity between groups of beams. In the same group, it is the ordinary beam search used to enhance the robustness of generating a single bundle. Between different groups, the diversity between multiple bundles is controlled according to the number of used nodes. Specifically, we use the number of node usage $C_{u,i}$ to modify the node score

$$E_{u,i} = \frac{S_{u,i} + p(L_{b,i}^t | L_b^1, \ldots, L_b^{t-1})}{C_{u,i}}. \tag{21}$$

Adaptive Stop. The final generated bundle size depends on when the generation process stops. The existing bundle generation method stops the generation process by predicting the end character, but due to the long-term bottleneck of the sequence-based method, it tends to stop prematurely and generate short sequences. In the graph generation method, the generation process is terminated by reaching the preset maximum generation step, and the number of nodes is controlled by slicing the final adjacency matrix. The slice size depends on the probability mass function (PMF) of the number of nodes in the training set. Such filtering is too violent and rough for item nodes with entity differences.

To model the bundle size distribution under the current platform and fine-tune it adaptively, we designed a special stopping criterion consisting of two conditions. First, we perform base constraints based on the accumulation of PMF. We use accumulation instead of direct indexing, which will make the bundle size close to but slightly smaller than the distribution in training set. Second, we use the graph density changes to fine-tune the base. When the drop rate of graph density is less than a certain threshold, there are no more retrieved nodes that can establish a strong association with the bundle that has been built in the graph. The overall algorithm is shown in Algorithm 1.

Algorithm 1. Bundle Graph Generation Network

Input: User ID u, Number of retrieval items η, Threshold for the drop rate of graph density δ, Beam search width ω, Output bundle list size K

Output: Diversified bundle list B with size K

$I_u, S_u \leftarrow$ Node Retrieval$(u, p^{\mathcal{U}}, q^{\mathcal{I}}, \eta)$
$\mathbf{h}_i \leftarrow H^T v^{I_u}, i \in I_u$
$C_u \leftarrow o(\eta)$
$L_b \leftarrow O(\eta \times \eta), b \in \{1, \ldots, K\}$
$g \leftarrow \varnothing$
for $t \in [1, \ldots, \eta]$ **do**
  $J_{u,b} \leftarrow G_b^{t-1}.nodes$
  $I_{u,b} \leftarrow I_u - J_{u,b}$
  $\tilde{G}_b^t, \tilde{L}_b^t \leftarrow$ Construct Augmented Graph$(G_b^{t-1}, L_b^{t-1}, I_{u,b})$
  $\mathbf{h}_i^t \leftarrow$ L-layers GCN$(\mathbf{h}_i^{t-1}, \tilde{G}_b^t), i \in I_u$
  $p(L_{b,i}^t) \leftarrow$ Mixture of $BernoulliDistributions(\mathbf{h}_i^t, i, j), i \in I_{u,b}, j \in J_{u,b}$
  $E_{u,i} \leftarrow$ Node Score$(S_{u,i}, p(L_{b,i}^t), C_{u,i}), i \in I_{u,b}$
  $\mathbf{\Omega}_{u,b} \leftarrow$ sample ω nodes from $I_{u,b}$ according to $E_{u,i}$
  $G_b^{t-1,i} \leftarrow$ add node i to $G_b^{t-1}, i \in \mathbf{\Omega}_{u,b}$
  $F_{u,b+i} \leftarrow$ Graph Score$(G_b^{t-1,i}, S_u)$
  $\tau \leftarrow \arg\max_i(F_{u,b+i}), i \in \mathbf{\Omega}_{u,b}$
  $G_b^t \leftarrow G_b^{t-1,\tau}$
  $L_b^t \leftarrow$ swap current row t with row τ in L_b^{t-1}
  $C_{u,\tau} \leftarrow C_{u,\tau} + 1$
  if Density Drop Rate $G_b^t, G_b^{t-1} \leq \delta$ **then**
    $g \leftarrow g \cup G_b^t$
  if $|g| = K$ **then**
    break;
rerank g according to graph score $F_{u,b}$
return $B \leftarrow$ [$G.nodes$ for G in g]

4 Experiments

4.1 Experimental Settings

4.1.1 Datasets and Evaluation Protocol

We use the following two real-world datasets for evaluation, with their statistics shown in Table 1.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

TABLE 1
Statistics of Two Utilized Real-World Datasets

Dataset	#U	#I	#B	#U-I	#U-B	#Avg. I in B
Netease	18,528	123,628	22,864	1,128,065	302,303	77.80
Youshu	8,039	32,770	4,771	138,515	51,377	37.03

- *Netease*. This is a dataset from the largest music platform in China collected by [1]. It enables users to bind songs with a specific theme or add any bundles to their favorites.
- *Youshu*. This dataset is constructed by [4] from Youshu, a famous book-review website. Similar to Netease, every bundle is a list of books that users desired.

It should be noted that there are differences between traditional bundle platforms such as e-commerce websites and other platforms, such as music (Netease). In e-commerce websites, if a user consumes a bundle containing items we do not like, money is what he/she costs; in music websites, if a user consumes a bundle containing items we do not like, time is what he/she costs (imagine a user plays a song list but it contains many songs he/she wants to skip over). Therefore, the decision-making can also be well explained in our utilized datasets.

Bundle Recommendation. For the bundle recommendation task, we adopt two widely used metrics[13], [14], Recall and NDCG, to judge the performance of the ranking list, where Recall measures the ratio of test bundles that have been contained by the top-K ranking list, while NDCG complements Recall by assigning higher scores to the hits at higher positions of the list. In the formula below, K indicates that we perform top-K rank, and we generate a bundle list B of length K for each user. b_k indicates that the bundle at position k in the generated bundle list B, and d represents the bundle in bundle set D that the user has actually interacted with

$$Recall@K = \frac{|D \cap B|}{|D|} \tag{22}$$

$$DCG@K = \sum_{k=1}^{K} \frac{2^{\text{hit}_k} - 1}{\log_2(k+1)}, \tag{23}$$

where $hit_k = 1$ when $b_k \in D$, otherwise $hit_k = 0$. Furthermore,$NDCG@K = DCG@K/IDCG@K$, where IDCG (Ideal DCG) is the DCG that assumes that the returned results are sorted by relevance.

Bundle Generation. For the bundle generation task, when measuring the quality of the generated bundle for a given user, consistency between the generated bundle and the users' historically interacted bundles must be considered. It can also be understood as the item-level accuracy. On the other hand, the metric should pay more attention to top positions in the ranking list (similar to traditional item recommendation task), which is ignored by existing works. In this work, we innovatively proposed the ranking metrics for bundle generation, taking into account both the item-level accuracy and the bundle's ranking position. The basic logic is, a) Use Jaccard similarity to measure the similarity of two bundles; b) Take the best similarity as the hit degree between a single bundle and multiple bundles; c) Replace the hits that are 0 or 1 in the traditional ranking metric with hit degree. We define and formulate the new metrics as follows:

$$Pre^{+}@K = \frac{1}{K}\sum_{k=1}^{K} \max_{d \in D} \left| \frac{b_k \cap d}{b_k \cup d} \right|. \tag{24}$$

$Pre^{+}@K$ is the top-K precision, representing the average accuracy between each position in the top-K bundle list of each user and the best matching bundle in the ground-truth

$$Recall^{+}@K = \frac{1}{|D|}\sum_{d \in D} \max_{b_k \in B} \left| \frac{b_k \cap d}{b_k \cup d} \right|. \tag{25}$$

$Recall^{+}@K$ represents the top-K recall rate, which has a different perspective from precision, and is used to evaluate the matching degree between the best bundle recalled from the top-K bundle list and the ground truth

$$MRR^{+}@K = \frac{1}{K}\sum_{k=1}^{K} \frac{1}{k} \max_{d \in D} \left| \frac{b_k \cap d}{b_k \cup d} \right|. \tag{26}$$

$MRR^{+}@K$ is the mean reciprocal rank. The traditional MRR is the mean value of the inverse of the ranking of the first hit item. Here, the ranking of the best matching bundle and the matching degree of the bundle are considered

$$DCG^{+}@K = \sum_{k=1}^{K} \frac{2^{\max_{d \in D} \left| \frac{b_k \cap d}{b_k \cup d} \right|} - 1}{\log_2(k+1)}. \tag{27}$$

$DCG^{+}@K$ is Discounted Cumulative Gain. We take the matching degree of the best matching bundle as the gain and consider the ranking order, so that the higher-ranked bundles have higher gains and the lower-ranked bundles are discounted. Furthermore, we adopt Normalized Discounted Cumulative Gain, $NDCG^{+}@K = DCG^{+}@K/IDCG^{+}@K$, where IDCG (Ideal DCG) is the DCG that sorts the returned results by relevance

$$Div^{+}@K = \frac{1}{K(K-1)}\sum_{k=1}^{K-1}\sum_{j=k+1}^{K} \left(1 - \left| \frac{b_k \cap b_j}{b_k \cup b_j} \right|\right). \tag{28}$$

Div+@K is a cross-bundle diversity[6], which prevents the model from generating bundles with only high-scoring items. At the same time, it can evaluate the coverage of the items in the system. It is defined as the Jaccard Distance of each bundle pair in each final bundle list, that is, 1-Jaccard Similarity. There is a trade-off between the diversity and the above accuracy metrics, and models should increase diversity as much as possible while maintaining accuracy.

N.B. Since we use Jaccard Similarity to measure hit degree, each metric's unit is percent. Except for $NDCG^{+}@K$, it is normalized and has no fundamental unit.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

It is worth mentioning that the bundle generation task can be actually considered as a kind of bundle completion task without any item seeds. Actually, there are two reasons that we do not choose item seeds. First, the manner of how to choose the item seeds may have a large influence on the recommendation performance. Second, if some items are predefined, it seems that the major challenge in real-world's bundle recommendation task, estimating users interests, will be ignored. For example, if we have predefined iPhone as a seed, then to complete the bundle is much easier than generating a new bundle from zero, as the model can just predict items matching with iPhone. Therefore, to make our evaluation closer to real-world applications, we do not choose any item seed and follow the evaluation protocol of BGN.

4.1.2 Baselines

We compare BGCN with the following six state-of-the-art methods for prebuilt bundle recommendation.

- *MFBPR* [8]. This is a matrix factorization method under a Bayesian Personalized Ranking pairwise learning framework, which is widely used for implicit feedback.
- *GCN-BG* [15]. This is a widely used graph neural network model in the recommendation. In the method, we apply GCN to the user-bundle bipartite interaction graph.
- *GCN-TG* [15]. We use the same way with BGCN to build the user-item-bundle tripartite unified graph, but the difference is that the message is passed between all kinds of nodes at the same time.
- *NGCF-BG* [16]. This is the state-of-the-art model which uses graph neural networks to extract higher-order connectivity for the recommendation. Similar to GCN-BG, here only user and bundle are used to build the interaction bipartite graph.
- *NGCF-TG* [16]. NGCF-TG performs propagation in the same way as GCN-TG on the tripartite graph. Since item information is introduced, the embedding learned by the model might have stronger representation ability.
- *DAM* [4]. This is the state-of-the-art deep model in bundle recommendation. It uses the attention mechanism and multi-task learning framework to extend NCF [13].

We compare BGGN with the following three methods for personalized bundle generation.

- *Freq* [17]. This is a well-known frequent itemsets mining algorithm. According to frequent itemsets in bundles, we generate bundles and recommend the most frequent k ones to users without personalization.
- *BGN* [6]. This is the state-of-the-art bundle generation model. It formalizes the personalized bundle list generation as a structured prediction problem, and is solved by the sequence generation method.
- *BGCN* [7]. This is our bundle recommendation method based on GCN. According to the embedding learned, we choose each user's favorite items to form a fixed-size bundle. But this is a single-bundle generation method. Compared with the above methods, it additionally utilizes u-i interaction data.

4.1.3 Parameter Settings

For all bundle recommendation methods, we adopt BPR loss and set the negative sampling rate to 1. We employ Adam optimizer with the 2048-size mini-batch and fit the embedding size as 64. The learning rate is searched in {1e-5, 3e-5, 1e-4, 3e-4, 1e-3, 3e-3} and L_2 regularization term is tuned in {1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2}. For GCN-based methods, we search the message dropout and node dropout within {0, 0.1, 0.3, 0.5}. For training, the hard-negative samples are selected with an 80% probability.

For bundle generation methods, we use the best parameters in the BGN original paper. For the BGCN in the bundle generation task, the fixed bundle size we set is the average bundle size in the training set. For the BGGN method, the learning rate is searched in {1e-1, 1e-2, 1e-3} and GCN layers is tuned in {5, 6, 7, 8, 9, 10}. In the generating process, we will set the number of nodes retrieved by each user as the product of the number of bundles to be generated and the average bundle size. The threshold of the graph density decrease rate is simply set to 0.1.

4.2 Performance Comparison

The bundle recommendation experiment results are reported in Table 2. From the results, we have the following observations.

- *Our proposed BGCN achieves the best performance.* We can observe that our model BGCN significantly outperforms all baselines in terms of Recall and NDCG metrics. Specifically, BGCN outperforms the best baseline by 19.65%-23.18% and 10.77%-12.36% on the Netease dataset and Youshu dataset, respectively. Due to the special design, BGCN is the best model for utilizing graph structure and item interaction at the same time.
- *Graph models are effective but not enough.* Although the better performance of GCN and NGCF compared to MFBPR verifies the strong power of the graph model, they fail to exceed the state-of-the-art baseline DAM that leverage item interaction to recommend bundle. Therefore, designs to make graph neural network work in the bundle task are necessary, which validates our motivation to incorporate items into the user-bundle graph.
- *More input does not always mean better performance.* The performance of GCN-TG and NGCF-TG with the same constructed graph as BGCN is not always better than GCN-BG and NGCF-BG. This shows that if the model itself cannot distinguish the affiliation relation between item and bundle nodes, the extra item input may bring the noise. Therefore, our special design for propagation in the tripartite unified graph is necessary.

The bundle generation experiment results are reported in Table 3. From the results, we have the following observations.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

TABLE 2
Bundle Recommendation Performance Comparisons on Two Real-World Datasets With Six Baselines

Method	Netease						Youshu					
	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
MFBPR	0.0355	0.0181	0.0600	0.0246	0.0948	0.0323	0.1959	0.1117	0.2735	0.1320	0.3710	0.1543
GCN-BG	0.0370	0.0189	0.0617	0.0255	0.1000	0.0342	0.1982	0.1141	0.2661	0.1322	0.3633	0.1541
GCN-TG	0.0402	0.0204	0.0657	0.0272	0.1051	0.0362	0.2032	0.1175	0.2770	0.1371	0.3804	0.1605
NGCF-BG	0.0395	0.0207	0.0646	0.0274	0.1021	0.0359	0.1985	0.1143	0.2658	0.1324	0.3542	0.1524
NGCF-TG	0.0384	0.0198	0.0636	0.0266	0.1015	0.0350	0.2119	0.1165	0.2761	0.1343	0.3743	0.1561
DAM	0.0411	0.0210	0.0690	0.0281	0.1090	0.0372	0.2082	0.1198	0.2890	0.1418	0.3915	0.1658
BGCN	**0.0491**	**0.0258**	**0.0829**	**0.0346**	**0.1304**	**0.0453**	**0.2347**	**0.1345**	**0.3248**	**0.1593**	**0.4355**	**0.1851**
% Improv.	19.67%	22.89%	20.17%	23.18%	19.65%	21.76%	10.77%	12.22%	12.36%	12.33%	11.23%	11.62%

TABLE 3
Bundle Generation Performance Comparisons on One Real-World Dataset With Three Baselines

Method	Youshu											
	Pre@5	Pre$^+$@5	Recall$^+$@5	NDCG$^+$@5	MRR$^+$@5	Div$^+$@5	Pre@10	Pre$^+$@10	Recall$^+$@10	NDCG$^+$@10	MRR$^+$@10	Div$^+$@10
Freq	0.2638%	0.3554%	1.1179%	0.3429	0.1733%	**96.67%**	0.2774%	0.3696%	1.3155%	0.3572	0.1114%	90.37%
BRP	0.2928%	0.3018%	0.4114%	0.1885	0.1168%	54.78%	0.2876%	0.3081%	0.4989%	0.1475	0.1093%	52.86%
BBPR	0.3295%	0.3359%	0.5116%	0.2098	0.1310%	64.95%	0.3379%	0.3373%	0.5957%	0.2135	0.1289%	68.87%
BGN (λ=0)	0.3619%	0.4042%	0.6134%	0.2393	0.1827%	44.14%	0.3658%	0.4011%	0.6923%	0.2446	0.1175%	48.85%
BGN (λ=5)	0.2861%	0.3607%	0.7709%	0.2572	0.1658%	69.18%	0.2990%	0.3719%	0.9467%	0.2703	0.1076%	67.71%
BGCN	0.7837%	0.9401%	0.5446%	0.5101	0.4293%	-	0.7996%	0.9401%	0.5446%	0.5101	0.2753%	-
BGGN	**0.9676%**	**1.1499%**	**1.7062%**	**0.6295**	**0.5314%**	93.29%	**0.9589%**	**1.1366%**	**2.1643%**	**0.6254**	**0.3384%**	**93.95%**
% Improv.	23.47%	22.32%	52.63%	23.40%	23.78%	-3.50%	19.92%	20.90%	64.52%	22.60%	22.92%	3.96%

- *Our proposed BGGN achieves the best performance.* We can observe that our model BGGN achieved precision improvements of 20.90%-22.32%, recall improvements of 52.63%-64.52%, NDCG improvements of 22.60%-23.40%, and MRR improvements of 22.92%-23.78%. This demonstrates the superiority of structuring bundles into graphs in large-size (37.03 in average) bundle scenarios.
- *Structuring the bundle into a sequence cannot work for large-size bundles.* The sequence-based method BGN ($\lambda = 0$) and its variants ($\lambda = 5$) for controlling diversity, due to the long-term bottleneck of the sequence-based method, performed unsatisfactorily in large-scale bundle scenarios. The sequence-based method requires that items be sorted by importance in advance. Suffering from the fact that the content platform does not have a priori of importance similar to price, bundles' generation will be affected by the sequence's random order.
- *Non-personalized and non-diversified methods perform poorly.* We can observe that since Freq generates the same bundle for all users, its accuracy-related metrics are low. Such non-personalized methods always have strong diversity, and our method achieves close and even better diversity than these non-personalized methods. we have also used BGCN in the bundle-generation task and it also achieves very good performance. Due to the novel design in graph generation, the newly-proposed BGGN can further significantly outperform BGCN. In fact, BGGN can be regarded as a kind of extension of BGCN when faced with the application scenario where there is no pre-built bundle. Specifically, in BGGN, we reserve BGCN's core design of multi-layer propagation to capture high-order correlation, and introduce new designs of bundle graph construction, nodes retrieval, and beam search method on the graph, to handle the challenge of generating new bundles.

Besides, as for the model size, we can directly calculate it according to dataset scale and hyper-parameter settings. As for the complexity, we report the time cost of each training epoch. We have presented the results of BGCN and BGGN compared with the most competitive baselines of two tasks in Table 2. We can find that compared with those competitive baselines, our proposed models' model size and efficiency is similar to baseline models, while our models significantly outperform baselines on recommendation performance.

We have also shown the model size and efficiency of our proposed models and baselines in Table 4.

4.3 Ablation and Data Sparsity Study in BGCN

We performed ablation studies as shown in Table 5.

- *Effectiveness of Levels Propagation.* We compare the performance of the model performing propagation at the only item-level, only bundle level, and both levels. The results show that the model with two levels outperforms the model with on only bundle level by 9.36%-11.57% and 2.75%-11.39% on the Netease dataset and Youshu dataset, respectively.

TABLE 4
Comparison of Model Size and Efficiency

Model	#MF	#NGCF	#DAM	#BGCN	#BGN	#BGGN
Model size(M)	4028	6652	7644	6913	8780	7743
Efficiency(min)	6.42	8.03	9.74	7.96	12.09	11.81

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

TABLE 5
Ablation Study of the Key Designs in BGCN

Model		Netease		Youshu	
		Recall@40	NDCG@40	Recall@40	NDCG@40
Levels Propagation	Item Level	0.0121	0.0046	0.0786	0.0419
	Bundle Level	0.0685	0.0284	0.2805	0.1387
	I&B Levels	0.0749	0.0317	0.3124	0.1425
B2B Propagation	No B2B	0.0708	0.0297	0.2866	0.1400
	Unweighted B2B	0.0738	0.0312	0.3040	0.1418
	Weighted B2B	0.0749	0.0317	0.3124	0.1425
Hard-negative Sample	No Hard	0.0749	0.0317	0.3124	0.1425
	Item Level	0.0807	0.0343	0.3235	0.1573
	Bundle Level	0.0816	0.0343	0.3240	0.1581
	I&B Levels	0.0829	0.0346	0.3248	0.1593

- *Effectiveness of B2B Propagation*. The impact of overlap-based bundle propagation is tested by comparing the performance of models with no b2b propagation, unweighted b2b propagation, and weighted b2b propagation. The model's performance gains introduced overlap-based b2b propagation are 5.84%-6.74% and 1.79%-9.02% on the Netease dataset and Youshu dataset, respectively. It is shown that adding b2b propagation can help extract bundle correlations, especially when the degree of overlap is injected into the model.
- *Effectiveness of Hard-negative sample*. We compare the model's performance under training with no hard samples, and hard samples at the item-level, the bundle level, and both levels, respectively. We can observe that the model with hard-negative sample performs better than the model with simple sample by 9.26%-10.64% and 3.95%-11.78% on the Netease dataset and Youshu dataset, respectively. This demonstrates that hard-negative sample improves performance, especially when considering hard-negative samples at both item and bundle levels.

Compared to items, bundles have extremely rare interactions, so data sparsity is a significant issue for bundle recommendation. To study whether BGCN can alleviate the issue, we divide users into three groups according to sparsity as shown in Fig. 3 and present the recommendation performance of the Netease dataset. We can observe that the performance gap between BGCN and other methods becomes larger when data become sparser, except for the NGCF-TG which is very similar to our model. Furthermore, even in the user group with only 0 to 3 purchase records, BGCN still keeps an excellent performance of 0.069 for Recall@40 and 0.023 for NDCG@40, which outperforms the best method NGCF-TG by 30.19% and 27.78%, respectively. As a summary, we conclude that our proposed BGCN model can alleviate the data sparsity issue.

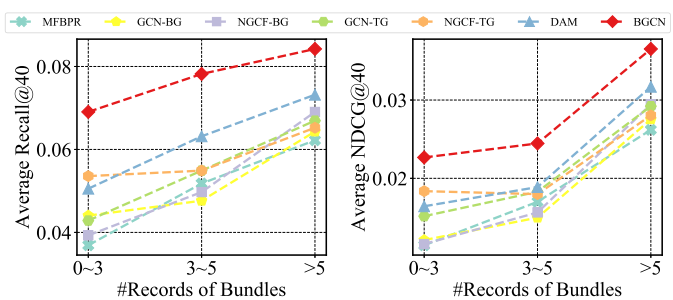


Fig. 3. Performance with different data sparsity (higher numbers indicate better performance).

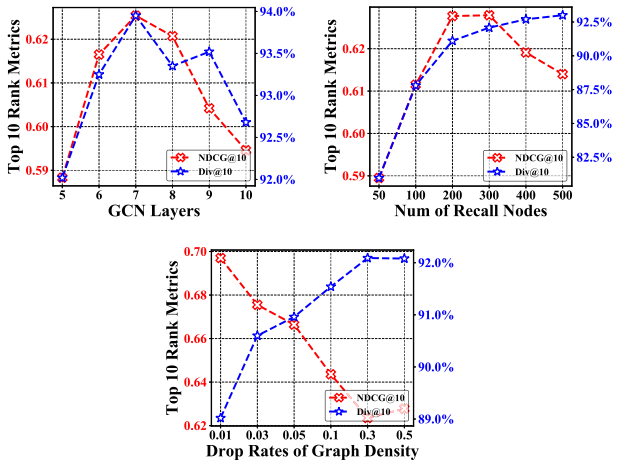


Fig. 4. Performance with different hyperparameters (higher numbers indicate better performance).

4.4 Impact of Hyper-Parameters and Visualization of Learned Embeddings and Generated Graphs in BGGN

To evaluate the impact of several hyperparameters on the BGGN method and provide practical suggestions of hyperparameters setting, we conducted hyperparameter comparison experiments, as shown in Fig. 4.

- *Impact of GCN Layers*. We compared the model's performance of different layers. It can be seen from the results that the performance is better as the number of layers increases at the beginning, but when the number of layers increases to a certain level, it will cause over-smooth [18], [19], [20]. The characterization of items tends to converge to the same value, and it is difficult to distinguish. In the small bundle graph, information can reach any node from any other node within a few hops[21], [22]. At this time, only the receptive fields of a few convolutional layers have covered the entire graph [23]. Therefore, adding more layers does not help reach the remote node, but the model's parameter amount and inference time will increase accordingly.
- *Impact of the Number of Retrieved Nodes*. We also tested the impact of different numbers of retrieved nodes on performance. The results show that as the number of retrieved nodes increases, the diversity will increase, but the accuracy will decrease. This is consistent with the intuition that there is a trade-off between precision and diversity. A larger number of retrieval will increase the range of options when constructing the bundle, resulting in greater diversity. But what follows is that it may introduce items that users don't like very much. Besides, it means a larger adjacency matrix. Although we use a sparse form, this is still a trial for computing memory consumption.
- *Impact of the Graph Density Drop Rate*. We compared the model's performance when setting different

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

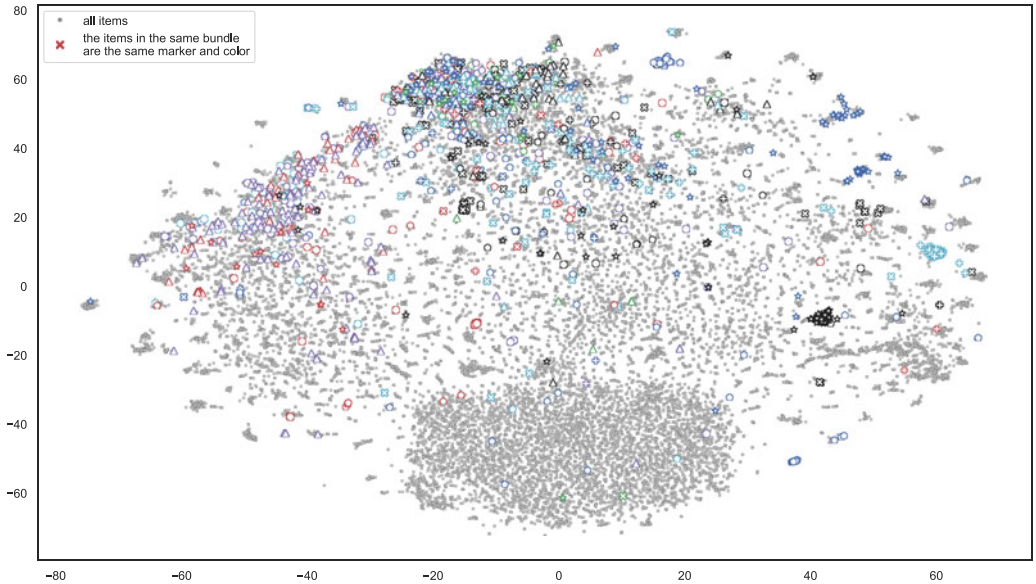


Fig. 5. t-SNE visualization of items in the same bundle. The clustering shown here shows that our method can learn the correlation between the items in the bundle.

thresholds for the drop rate of graph density. We can observe that as the threshold increases, the NDCG decreases but the diversity increases. The reason is that the change in bundle size affects performance. When the graph density drop threshold is set larger, the condition for stopping graph generation is looser and it tends to generate smaller graphs, and vice versa. When to properly stop the generation process is vital, however the sequence-based method tends to stop prematurely due to their long-term bottleneck. We can control the closeness of the item associations in the generated bundle by controlling the graph's density.

To further verify whether the graph generation method has learned the items correlation in one bundle, we performed a t-SNE embeddings visualization, as shown in Fig. 5. We use different markers and colors to combine and randomly select items in 50 bundles for marking. Items in the same bundle are marked with the same marker and color. It can be observed that the items in the same bundle show aggregation. This shows that our method can effectively learn the correlation between bundled items.

We also randomly selected a user to visualize the graph structure of some bundles generated for him, as shown in Fig. 6. We can see that different bundles have different graph densities. Some items have edges between most of the items in the bundle, and some items have edges between only one of the items. The bundle is structured as a graph, which thoroughly quantifies the high-order correlation between the bundled items, which was difficult to measure in the past. We can easily observe the core items (with higher degrees) and marginal items (with lower degrees or more hops from the core item nodes) in the bundle.

5 Related Work

5.1 Prebuilt Bundle Recommendation

The development of recommender systems has long been divided into two directions, namely, rating prediction and item recommendation. Early recommendation studies mainly focused on rating prediction with explicit feedback [24]. The recent research focus is shifting towards implicit data [13], [25]. Collaborative filtering (CF) with implicit feedback is usually treated as a Top-K item recommendation task. It is more practical and challenging [26], and accords more closely with the real-world recommendation scenario. However, in practice, users are often exposed to a set of items and the items are not independent. Along this line, a framework for recommending relevant and diverse items was proposed [27], which explicitly takes the coverage of users' interests into account. Another curiosity-based work was introduced [28], which combines relevance and curiosity in the recommendation process. Except for the work considering the correlation among items, as bundles are widely used everywhere, several efforts have been made to solve the problem of recommending bundles of items instead of isolated items. Sar Shalom *et al.* [29] aims to recommend a list of items to users by optimizing the list's click-through rate based on a two-layered CF framework. List Recommendation Model (LIRE) [2] simultaneously considered the users' previous interactions with both lists and individual items by optimizing a pairwise ranking loss. The relationship between the list and its contained items is built based on a linear combination. Embedding Factorization Machine (EFM) [1] treat a list as a sentence and items within the list as words. The word embedding algorithm is used to explore the co-occurrence information among items and lists. Recently, Deep Attentive Multi-Task Model (DAM) [4] jointly modeled user-bundle interactions and user-item interactions in a multi-task manner. The multi-task framework aims to transfer the benefits of the item task to the bundle task to alleviate the scarcity of user-bundle interactions. To our best of knowledge, we are the first to utilize graph-structure along with graph neural networks, to capture the high-order topology, which are ignored by existing works.

5.2 Graph-Based Recommendation

Graph-based recommendation constructs input data in the form of graph and design graph-learning models for

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

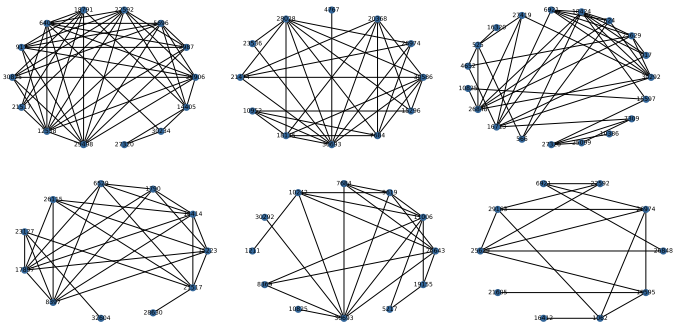


Fig. 6. Structure visualization of sample graphs generated. It thoroughly quantifies the high-level correlation between the bundled items, and the core items and marginal items in the bundle can be easily distinguished.

recommendation. To be specific, the possibility of two entities interacting can be transformed into a link prediction task on the graph, of which users and items are represented as nodes. For example, Some works [30], [31] apply random-walk based methods to graph-based recommendation. Some other works [32], [33] rely on the graph-embedding techniques to learn user/item embeddings for recommendation. Recently, graph convolutional networks (GCN) [9], [34] achieves a quantum leap in graph learning tasks. The basic idea of GCN is performing embedding propagation to capture node feature and graph structure simultaneously. It can reduce the high-dimensional adjacency information of a node in the graph to a low-dimensional vector representation. With the strong power of learning structure, GCN is widely applied in recommender systems. Berg *et al.* [15] first applied GCN to the recommendation to factorize several rating matrices. Ying *et al.* [35] extended it to web-scale recommender systems with neighbor-sampling to reduce computation costs. Wang *et al.* [16] approached a more general model that uses high-level connectivity learned by GCN to encode CF signals. Compared with existing works of graph-based recommendation, we propose new designs of graph convolutional networks for modeling user-item-bundle relations, including user-bundle interaction, user-item interaction, and bundle-item affiliation, for bundle recommendation.

5.3 Personalized Bundle Generation

For bundle generation, early methods model the item correlation in the bundle based only on the item co-occurrence frequency. Frequent itemsets mining algorithms such as Apriori [17] generate high-quality but non-personalized bundles. Parameswaran *et al.* [36] studied the cross-item dependencies in personalized curriculum bundling. The cost and usefulness of items to users and the compatibility between items are considered in [37]. However, these methods cannot utilize rich features and fail to generalize similar items and users to deal with the cold start problems in the recommendation system. For the single bundle recommendation, Zhu *et al.* [5] propose a personalized method based on integer quadratic programming, ignoring the higher-order terms of cross-item dependencies, which leads to a sub-optimal result. Beladev *et al.* [38] integrated the demand function and price model into collaborative filtering (CF) techniques to maximize both the user's probability of purchase and the revenue gained from selling the bundle. Qi *et al.* [39] studied the group's preference towards packages and incorporated factors such as user impact, package viability, and recommendation fairness. Liu *et al.* [40] formulated each purchased product bundle as a bundle graph and proposed a probabilistic model BPM (Bundle Purchases with Motives) to capture each bundle graph's composition. For the bundle list recommendation, Xie *et al.* [41] learn an additive utility function through user feedback to generate top-k bundles. However, the designed function is a linear combination of the corresponding features, which fails to capture the complicated dependencies. Pathak and Gupta[3] build an optimization criterion upon the latent factor model, combining the corresponding preference scores with a greedy annealing schedule, which might be caught in the local optimum due to randomness and instability of annealing. A recent work [6] formalized generating a bundle list into a structured prediction problem to capture the complex dependencies between items in the bundle. However, structuring the bundle into a sequence will be affected by the sequence order and has a long-term bottleneck. To this end, we structure the bundles into graphs and learn the complex dependencies of items between bundles , which are the first to utilize graph neural networks, to capture the high-order correlation.

5.4 Graph Generation

As the natural data structure to represent relational and structural information in many domains, the ability to generate graphs plays a significant role in drug design [42], [43], model architecture search [44], and network science [45], [46], [47]. But almost no related work has been applied to recommender systems. The research paradigms of graph generation models are divided into two categories. The first category of models generates the components of the graph independently or with only weak dependency structure of the generation decisions. Examples of these models include the variational auto-encoder (VAE) model with convolutional neural networks for sequentialized molecule graphs [48] and Graph VAE models [49], [50], [51]. These models generate the individual entries in the graph adjacency matrix (i.e., edges) independently given the latent; this independence assumption makes the models efficient and generally parallelizable but can seriously compromise the quality of the generated graphs [52], [53]. The second category of deep graph generative models makes auto-regressive decisions when generating the graphs. By modeling graph generation as a sequential process, these approaches naturally accommodate complex dependencies between generated edges. Previous work on auto-regressive graph generation utilize recurrent neural networks (RNNs) on domain-specific sequentializations of graphs (e.g., SMILES strings) [54], as well as auto-regressive models that sequentially add nodes and edges [52], [53] or small graph motifs [55] (via junction-tree VAEs). A recent work[10] utilizes graph structure information in the generation process by using graph neural networks, reducing the dependence on specific domain knowledge and node order. We have improved it to adapt to the recommendation scenario, in which nodes are a large number of items with entity differences. Graph generation has not been well considered in

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

recommender systems. Our work proposes a graph-generation based method for generating new bundles.

6 Conclusion

In this work, we study the task of bundle representation and generation systems. We propose a graph-based recommendation solution BGCN that re-constructs the two kinds of interaction and an affiliation into the graph. The model utilizes the graph neural network's powerful ability to learn the representation of two dependent entities from complex structures. And we propose a graph-based generation solution BGGN that re-constructs the bundle as a graph. The model learns the structure information and high-order item-item relationships in the bundle graph through GNN. Extensive experiments demonstrate their effectiveness on real-world datasets.

References

[1] D. Cao, L. Nie, X. He, X. Wei, S. Zhu, and T.-S. Chua, "Embedding factorization models for jointly recommending items and user generated lists," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 585–594.

[2] Y. Liu, M. Xie, and L. V. Lakshmanan, "Recommending user generated item lists," in *Proc. 8th ACM Conf. Recommender Syst.*, 2014, pp. 185–192.

[3] A. Pathak, K. Gupta, and J. McAuley, "Generating and personalizing bundle recommendations on steam," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 1073–1076.

[4] L. Chen, Y. Liu, X. He, L. Gao, and Z. Zheng, "Matching user with item set: Collaborative bundle recommendation with deep attention network," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2095–2101.

[5] T. Zhu, P. Harrington, J. Li, and L. Tang, "Bundle recommendation in ecommerce," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2014, pp. 657–666.

[6] J. Bai *et al.*, "Personalized bundle list recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 60–71.

[7] J. Chang, C. Gao, X. He, D. Jin, and Y. Li, "Bundle recommendation with graph convolutional networks," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, SIGIR, Xi'an, China*, 2020.

[8] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.

[9] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn Representations*, 2017, pp. 1–13.

[10] R. Liao *et al.*, "Efficient graph generation with graph recurrent attention networks," in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 4255–4265.

[11] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," 2015, *arXiv:1511.05493*.

[12] A. K. Vijayakumar *et al.*, "Diverse beam search: Decoding diverse solutions from neural sequence models," 2016, *arXiv:1610.02424*.

[13] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. World Wide Web*, 2017, pp. 173–182.

[14] M. Jamali and M. Ester, "Using a trust network to improve top-n recommendation," in *Proc. 3rd ACM Conf. Recommender Syst.*, 2009, pp. 181–188.

[15] R. V. D. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," in *Proc. Knowl. Discov. Data Mining Workshop*, 2017, pp. 1–7. [Online]. Available: https://www.kdd.org/kdd2018/files/deep-learning-day/DLDay18_paper_32.pdf

[16] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 165–174.

[17] R. Agarwal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th VLDB Conf.*, 1994, pp. 487–499.

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[19] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–37.

[20] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.

[21] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," *in Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–15.

[22] H. NT and T. Maehara, "Revisiting graph neural networks: All we have is low-pass filters," 2019, *arXiv:1905.09550*.

[23] P. Barceló, E. V. Kostylev, M. Monet, J. Pérez, J. Reutter, and J. P. Silva, "The logical expressiveness of graph neural networks," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–21.

[24] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 426–434.

[25] X. Li and J. She, "Collaborative variational autoencoder for recommender systems," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 305–314.

[26] R. Pan *et al.*, "One-class collaborative filtering," in *Proc. 8th IEEE Int. Conf. Data Mining*, 2008, pp. 502–511.

[27] C. Sha, X. Wu, and J. Niu, "A framework for recommending relevant and diverse items," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 3868–3874.

[28] P. Zhao and D. L. Lee, "How much novelty is relevant? It depends on your curiosity," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2016, pp. 315–324.

[29] O. Sar Shalom, N. Koenigstein, U. Paquet, and H. P. Vanchinathan, "Beyond collaborative filtering: The list recommendation problem," in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 63–72.

[30] C. Eksombatchai *et al.*, "Pixie: A system for recommending 3+ billion items to 200+ million users in real-time," in *Proc. World Wide Web Conf.*, 2018, pp. 1775–1784.

[31] T.-A. N. Pham, X. Li, G. Cong, and Z. Zhang, "A general recommendation model for heterogeneous networks," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3140–3153, Dec. 2016.

[32] J.-H. Yang, C.-M. Chen, C.-J. Wang, and M.-F. Tsai, "HOP-Rec: High-order proximity for implicit recommendation," in *Proc. 12th ACM Conf. Recommender Syst.*, 2018, pp. 140–144.

[33] C.-M. Chen, C.-J. Wang, M.-F. Tsai, and Y.-H. Yang, "Collaborative similarity embedding for recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 2637–2643.

[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *Proc. Int. Conf. Learn. Representations*, pp. 1–12, 2018.

[35] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 974–983.

[36] A. Parameswaran, P. Venetis, and H. Garcia-Molina, "Recommendation systems with complex constraints: A course recommendation perspective," *ACM Trans. Inf. Syst.*, vol. 29, no. 4, pp. 1–33, 2011.

[37] T. Deng, W. Fan, and F. Geerts, "On the complexity of package recommendation problems," *SIAM J. Comput.*, vol. 42, no. 5, pp. 1940–1986, 2013.

[38] M. Beladev, L. Rokach, and B. Shapira, "Recommender systems for product bundling," *Knowl. Based Syst.*, vol. 111, pp. 193–206, 2016.

[39] S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Recommending packages to groups," in *Proc. IEEE 16th Int. Conf. Data Mining*, 2016, pp. 449–458.

[40] G. Liu, Y. Fu, G. Chen, H. Xiong, and C. Chen, "Modeling buying motives for personalized product bundle recommendation," *ACM Trans. Knowl. Discov. Data*, vol. 11, no. 3, pp. 1–26, 2017.

[41] M. Xie, L. V. Lakshmanan, and P. T. Wood, "Generating top-k packages via preference elicitation," *Proc. VLDB Endowment*, vol. 7, no. 14, pp. 1941–1952, 2014.

[42] Y. Li, L. Zhang, and Z. Liu, "Multi-objective de novo drug design with conditional graph generative model," *J. Cheminformat.*, vol. 10, no. 1, 2018, Art. no. 33.

[43] J. You, B. Liu, Z. Ying, V. Pande, and J. Leskovec, "Graph convolutional policy network for goal-directed molecular graph generation," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 6410–6421.

[44] S. Xie, A. Kirillov, R. Girshick, and K. He, "Exploring randomly wired neural networks for image recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1284–1293.

[45] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Modern Phys.*, vol. 74, no. 1, pp. 47–97, 2002.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.

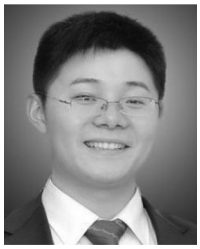
[46] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[47] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *J. Mach. Learn. Res.*, vol. 11, no. 33, pp. 985–1042, 2010.
[48] R. Gómez-Bombarelli *et al.*, "Automatic chemical design using a data-driven continuous representation of molecules," *ACS Central Sci.*, vol. 4, no. 2, pp. 268–276, 2018.
[49] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[50] M. Simonovsky and N. Komodakis, "Graphvae: Towards generation of small graphs using variational autoencoders," 2018, *arXiv:1802.03480*.
[51] Q. Liu, M. Allamanis, M. Brockschmidt, and A. Gaunt, "Constrained graph variational autoencoders for molecule design," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 7795–7804.
[52] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia, "Learning deep generative models of graphs," 2018, *arXiv:1803.03324*.
[53] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "GraphRNN: Generating realistic graphs with deep auto-regressive models," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5694–5703.
[54] M. H. Segler, T. Kogej, C. Tyrchan, and M. P. Waller, "Generating focused molecule libraries for drug discovery with recurrent neural networks," *ACS Central Sci.*, vol. 4, no. 1, pp. 120–131, 2017.
[55] W. Jin, R. Barzilay, and T. Jaakkola, "Junction tree variational autoencoder for molecular graph generation," 2018, *arXiv:1802.04364*.

Jianxin Chang is currently working toward the MS degree with the Department of Electronic Engineering, Tsinghua University, Beijing, China. His research interests include recommender systems and deep learning. His work on graph neural network based recommender system was the recipient of the Best Short Paper Honorable Mention Award in SIGIR 2020.

Chen Gao received the bachelor's and PhD degrees from the Department of Electronic Engineering, Tsinghua University, in 2016 and 2021, respectively. He is currently a postdoc researcher with the Department of Electronic Engineering, Tsinghua University. His research interests include data mining, information retrieval, and recommender system.

Xiangnan He received the PhD degree in computer science from the National University of Singapore. He is currently a professor with the University of Science and Technology of China. He has authored or coauthored more than 100 publications that appeared in top conferences, including the SIGIR, WWW, and KDD, and journals, including *IEEE Transactions on Knowledge and Data Engineering*, *ACM Transactions on Information Systems*, and *IEEE Transactions on Neural Networks and Learning Systems*. His research interests include information retrieval, data mining, and multimedia analytics.

Depeng Jin (Member, IEEE) received the BS and PhD degrees in electronics engineering from Tsinghua University, Beijing, China, in 1995 and 1999, respectively. He is currently a professor with Tsinghua University.



Yong Li (Senior Member, IEEE) received the PhD degree in electronic engineering from Tsinghua University in 2012. He is currently a tenured associate professor with the Department of Electronic Engineering, Tsinghua University. He has authored or coauthored more than 100 papers on first-tier international conferences and journals, including KDD, WWW, UbiComp, SIGIR, AAAI, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Mobile Computing* and his papers have total citations more than 8300. His research interests include machine learning and big data mining, particularly, automatic machine learning and spatial-temporal data mining for urban computing, recommender systems, and knowledge graphs. He was the general chair, the TPC chair, an SPC or a TPC member for several international workshops and conferences. He is on the editorial board of two IEEE journals. Among them, 10 are ESI Highly Cited Papers in computer science, and five were the recipient of conference Best Paper (run-up) awards.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.

Authorized licensed use limited to: Tsinghua University. Downloaded on July 31,2023 at 12:46:42 UTC from IEEE Xplore. Restrictions apply.