Purify and Generate: Learning Faithful Item-to-Item Graph from Noisy User-Item Interaction Behaviors

Yue He
Tsinghua University
heyue18@mails.tsinghua.edu.cn

Yancheng Dong
Tsinghua University
dongyc20@mails.tsinghua.edu.cn

Peng Cui
Tsinghua University
cuip@tsinghua.edu.cn

Yuhang Jiao
Beijing Kuaishou Technology Co., Ltd.
jiaoyuhang@kuaishou.com

Xiaowei Wang
Beijing Kuaishou Technology Co., Ltd.
wangxiaowei03@kuaishou.com

Ji Liu
Beijing Kuaishou Technology Co., Ltd.
jiliu@kwai.com

Philip S. Yu
University of Illinois at Chicago
psyu@cs.uic.edu

ABSTRACT

Matching is almost the first and most fundamental step in recommender systems, that is to quickly select hundreds or thousands of related entities from the whole commodity pool. Among all the matching methods, item-to-item (I2I) graph based matching is a handy and highly effective approach and is widely used in most applications, owing to the essential relationships of entities described in a powerful I2I graph. Yet, the I2I graph is not a ready-made product in a data source. To obtain it from users' behaviors, a common practice in the industry is to construct the graph based on the similarity of item embeddings or co-occurrence frequency directly. However, these methods tend to lose the complicated correlations (high-ordered or nonlinear) inside decision-making actions and cannot achieve the global optimal solution. Moreover, the correlations between items are usually contained in users' short-term actions, which are full of noise information (e.g. spurious association, missing connection). It is vitally important to filter out noise while generating the graph. In this paper, we propose a novel framework called Purified Graph Generation (PGG) dedicated to learn faithful I2I graph from sparse and noisy behavior data. We capture the 'confidence value' between user and item to get rid of exception action during decision making, and leverage it to re-sample purified sets that are fed into an unsupervised I2I graph structure learning framework called GPBG. Extensive experimental results from both simulation and real data demonstrate that our method could significantly benefit the performance of I2I graph compared to the typical baselines.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**;

KEYWORDS

Item2Item Graph, Recommender, Structure Learning, Denoise

ACM Reference Format:
Yue He, Yancheng Dong, Peng Cui, Yuhang Jiao, Xiaowei Wang, Ji Liu, and Philip S. Yu. 2021. Purify and Generate: Learning Faithful Item-to-Item Graph from Noisy User-Item Interaction Behaviors. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467205

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467205

1 INTRODUCTION

The first step in the recommendation is matching (or recall in other statements), that is to select a minority set of most relevant products in the commodity pool before ranking or else. Matching can greatly reduce the size of the candidate set and thus improve the efficiency of subsequent tasks. A most widely used and handiest way of matching is to select items according to importance weight in I2I graph that describes the essential relationships between entities. Although the I2I graph is not directly available from data source, the quality of it is of huge impact on the overall performance of recommendation.

Different from the domains where a graph is manually defined by experts, there is no acknowledged way to define a graph in the scene of recommendation. We can simply derive the I2I graph from the attributes data, that is to link videos with similar content or products with the same seller, but personalization will be in shortage in this static way. So it is preferred to construct the graph from a collection of interaction logs, i.e. historical user behaviors, in a data-driven manner. In industry, the common practices to build this data-driven graph include: 1) measuring the co-occurrence frequency among entities and heuristically connecting edges linking highly co-occurred entities; 2) learning the item embeddings from other recommendation tasks first and computing the edge weights according to the similarity between embeddings. However, both types of methods use a predefined deterministic function to construct the I2I graph. These construction methods cannot fully encode the complicated correlations (high-ordered or nonlinear) of items underlying users' decision-making process into the graph structure, causing I2I graph only arriving at the local optimal solution with insufficient performance in applications. It is highly

needed to propose a better paradigm of learning faithful I2I graph. To our knowledge, few works have focused on the problem of learning I2I graph, although it plays a great role in recommender.

On the other hand, internal relationships of products are mainly stored in short-term behaviors, e.g. the video list watched by a staff in tea time, which are full of noise information brought by exceptional actions. Promotion strategy on recommendation platform might lead a user watching sport videos to suddenly turn to a Game Ad. Then sport contents will have spurious association with that game. This issue is even worse in reality because the user log is too sparse to offset noise by itself. If the mistaken correlations are embedded into I2I graph, the biased structure is sure to seriously hurt the matching performance. So, it is the premise of generating quality graph to purify the behavior data.

Recently, some literature from graph neural networks (GNNs) [10] treats the input graph structure as learnable parameters and optimize it together with the parameters of neural networks to minimize a well-designed objective function of supervised ending task. These approaches can help both graph structure and model parameters to achieve their best solution perfectly recovering the input data distribution. Inspired by the thought, we propose a novel framework **PGG** that gets rid of exceptional action during decision making process, e.g. the process of user watching videos in above example, and generates I2I graph through a model called GPBG inversely transferring purified behavior data to graph structure in an unsupervised fashion. For more generality, we consider set data form in this work, where a piece of user log is represented by a set consisting of several items (i.e. we ignore temporal sequence between items in the set, which is subtly helpful but redundant to learn I2I graph).

Specifically, we define a confidence value to denote global preference between users and items, and use a neural network to capture it by effective positive and negative pairwise sampling. Given a raw set data from a user log, we reweigh each inner item with confidence value and re-sample a new set according to the selection probability for denoising. Further, we design an unsupervised I2I graph learning model called GPBG composed of two components, including a GCN to catch structural pattern into item embeddings and represent a set into pooled embedding vector (i.e. a set embedding), and a conditional variational auto-encoder (CVAE) architecture for behavior data reconstruction. With access to GPBG, we can generate personalized user behavior based on the I2I graph, and hence refine the graph structure to maximize the likelihood of observational data. Because the set re-sampling process and set to graph transfer process benefit each other, we finally iterative optimize them in our unified framework **PGG**.

In summary, **our contributions are highlighted as follows** :

- We investigate the problem of generating the faithful I2I graph in recommender scenario, where high concentration noise of observed data confound the identifiability of effective user behavior, resulting in insufficient performance of matching, as well as the overall recommender;
- We understand the impact of noisy behavior in a lager-scale real application dataset with fine-grained interaction.
- We devise an unified framework called **PGG** to purify noisy data and learn I2I graph in a benign circle of two interactional modules, data re-sampling with item reweighing and graph-based personalized behavior generation.
- We conduct extensive experiments on both simulation and large-scale real datasets to verify the effectiveness and robustness of our algorithm to agnostic environment.

The rest of this paper is organized as follows. Section2 reviews the literature of related fields. Section3 introduces our proposed novel framework. Section4 gives the detailed experiment settings and results to show the advantage of our model. Finally, we conclude this work at the end of the paper.

2 RELATED WORK

Item2Item Graph construction. Item2Item methods in recommender system allow us to filter out a small set of items from the whole commodity pool efficiently. There are a number of research aiming at constructing an I2I graph directly or based on embeddings. On one hand, using co-occurrence statistical information of items lead to correlation scores between item pairs, and can generate top related items for every candidate directly. In [12], they use chi-square measure to calculate the bias of word co-occurrences, which is a representation of the probability of being a keyword. Swing, a powerful matching method in Alibaba, counts the "swing" structures in the user-item graph as the relatedness between item pairs. On the other hand, many works learn item embeddings from user-item interactions or session information, and the correlation score is calculated by dot product between embedding pair. LightGCN[7] builds a simplified GCN model to learn user and item embeddings. EGES[17] constructs an item graph from user's actions and gets item embeddings using random walk and skip gram. GCE-GNN[18] proposes to learn item embeddings from session graph and global graph in session-based recommendation. Previous works focus on generating the item embeddings given the graph (user2item, user2session, item2item) fixed first and using a predefined deterministic function to construct I2I graph (e.g. dot product) finally. But both of data and graph from the original user behaviors are noisy, which will mislead the learning process, causing spurious association and missing connection. The constructive function easily loses information when encoding the complicated correlations into graph structure. As a result, the constructed I2I graph could only reach a local optimal solution with insufficient performance.

Graph Structure Learning. In recent years, numerous generative graph-based methods have been proposed to treat the graph structure as a learnable module and optimize it together with the parameters of GNN. Specifically, GraphRNN [19] models variable-sized chemical graphs by collapsing distinct nodes to unique BFS trees in an autoregressive manner. A GAN-based method MolGAN [2] learns molecular graphs with a reinforcement learning objective that encourages the generation of molecules with specific chemical properties. [11] formulates penalty terms of a VAE model to regularize the generated molecular graphs to meet the real-world validity constraints.

On the other hand, the properties of some real-world graphs are not clearly restricted and defined. Methods like NetGAN[1] introduce to learn the distribution of biased random walks over the original graphs using the Wasserstein GAN, thereby generating graphs without strict real-world constraints. GraphOpt [16] poses

Table 1: Notation and Definitions

Notation	Annotation
M	the number of users.
D	the number of items.
N	the number of sets/behavior data.
$\mathbf{G} \in \mathbb{R}^{D \times D}$	the adjacency matrix of Item2Item graph.
U_i	the symbol of i_{th} user .
I_d	the symbol of d_{th} item.
$\mathbf{s}_j \in \mathbb{N}^{1 \times D}$	the vector of j_{th} set, each position denotes the count of corresponding item in set.
$\mathbf{s}_j^* \in \mathbb{N}^{1 \times D}$	the new set re-sampled from $\mathbf{s}_j$.
$\mathbf{x}_i^U \in \mathbb{R}^{1 \times F^U}$	the feature of user U_i with dimension F^U.
$\mathbf{x}_d^I \in \mathbb{R}^{1 \times F^I}$	the feature of item I_d with dimension F^I.
L	the size of batch data.

link formation in graphs as a sequential decision-making process and solves it using reinforcement learning algorithm, the learned objective can serve as an explanation for the observed graph properties. GLN [14] utilizes graph convolutions to propose expected node features and treats the structure of each graph as ground truth to enhance the predicted structures and embeddings recursively. In some scenarios, the ground truth of graph structures is not available, LDS [4] proposes to jointly learn the graph structure and the parameters of the neural network by approximately solving a bilevel program, thus allowing GNN to be applied to data without graphs.

However, in recommender scene of our problem, none of the structure property and the specific ending task is known. Only the observational data of user behaviors is available. SGL [8] proposes to learn element graph from input set data directly, but does not consider about the personalized property of recommender. Hence, we devise a model called GPBG, which takes an unsupervised way to learn an I2I graph by recovering personalized user behavior.

Deep Generative Models. Deep generative models have exerted a tremendous fascination on research community with the capability of learning the distribution of observed data. Roughly speaking, there are two types of methods, including Generative Adversarial Networks (GAN) [6] and Variational AutoEncoder (VAE) [9]. GAN plays a zero-sum game between generator and discriminator to generate synthetic data as realistic as possible. In contrast, VAE directly captures the implicit distribution by data reconstruction in an encoder-decoder architecture, making it more suitable for high-dimensional sparse data. Inputting additional information into both encoder and decoder, Conditional VAE (CVAE) [15] further learns the conditional distribution. In this paper, we adapt CVAE in our framework to generate personalized user behavior.

3 ALGORITHM

In this section, we introduce the problem we investigate and our proposed framework **PGG** in details.

3.1 Notations and Problem

The major notations used in this paper are standardized in Table 1. The most common and easily available data in recommender system is user historical behaviors in data collection, with the form like $(U_i, \mathbf{s}_j = \{I_d, ...\})$, where $\mathbf{s}_j$ is the set of items purchased/rated by user U_i. For brief description in following sections, we explicitly formulate user behavior data as four matrixs:

- a mapping matrix $\mathbf{A} \in \{0, 1\}^{M \times N}$ where $a_{i,j} = 1$ indicates multi-set $\mathbf{s}_j$ is made by user U_i.
- a set matrix $\mathbf{S} \in \{\mathbb{N}\}^{N \times D}$ consists of the count $s_{j,d}$ of item I_d appearing in set $\mathbf{s}_j$.
- an user feature matrix $\mathbf{X}^U \in \mathbb{R}^{M \times F^U}$
- an item feature matrix $\mathbf{X}^I \in \mathbb{R}^{D \times F^I}$

The initial user feature and item feature could be obtained from profile. We default $a_{i,j} = 1$ if not specified. The purpose of this work is to solve the Problem 1.

Problem 1. *Given* $\mathbf{A}$, $\mathbf{S}$, $\mathbf{X}^U$ *and* $\mathbf{X}^I$, *the goal of this work is learn an item2item graph* $\mathbf{G} \in \mathbb{R}^{D \times D}$ *that faithfully describes the correlations between items.*

3.2 Motivation

The personalized behavior distribution $P(\mathbf{s}|U_i)$ of an user U_i is a function of marginal distribution $P(\mathbf{s})$ and U_i's personal preference $\mathbf{x}_i^U$. Because the likelihood of set $\mathbf{s}_j$ is caused by the inner correlations in $\mathbf{s}_j$, we hence define the likelihood function as,

$$P(\mathbf{s} = \mathbf{s}_j|U_i) = h(\mathbf{x}_i^U, \mathbf{s}_j, \mathbf{X}^I, \mathbf{G}). \tag{1}$$

where $\mathbf{X}^I$ denotes the raw semantic item features, $\mathbf{G}$ is the I2I graph encoding all the relative patterns of items in topology structure, and h is a stationary mapping function denoting the generation mechanism.

Equation (1) indicates if we can build the connection between set data and I2I graph by function h, the remaining work is to directly embed complicated correlations (high-ordered and non-linear) between items in observed behavior into the graph structure. Compared to graph construction methods, such learnable approach is much possible to achieve the best optimal solution of $\mathbf{G}$ with the advantage of less model misspecification.

However, behavior data is always noisy because of its sparsity and kinds of realistic exceptional actions during decision making process. Directly using all the input data $\{\mathbf{s}_j\}$ (s.t. $a_{i,j} = 1$) of user U_i as the estimation of $P(\mathbf{s}|U_i)$ to learn I2I graph will certainly lead to biased structure of $\mathbf{G}$. So we have to correct this conditional distribution before taking it as supervised information to learn $\mathbf{G}$.

In summary, we should purify the input set data and fit the generation mechanism h to accomplish the goal of learning a high-quality I2I graph. Hence, we propose a novel framework called PGG to solve these two problems in a unified framework, which can be decomposed into two modules including data re-sampling with item reweighing and graph-based personalized behavior generation. These two modules promote each other by iterative optimization in PGG framework. But for ease of understanding, we introduce their details separately in following sections.

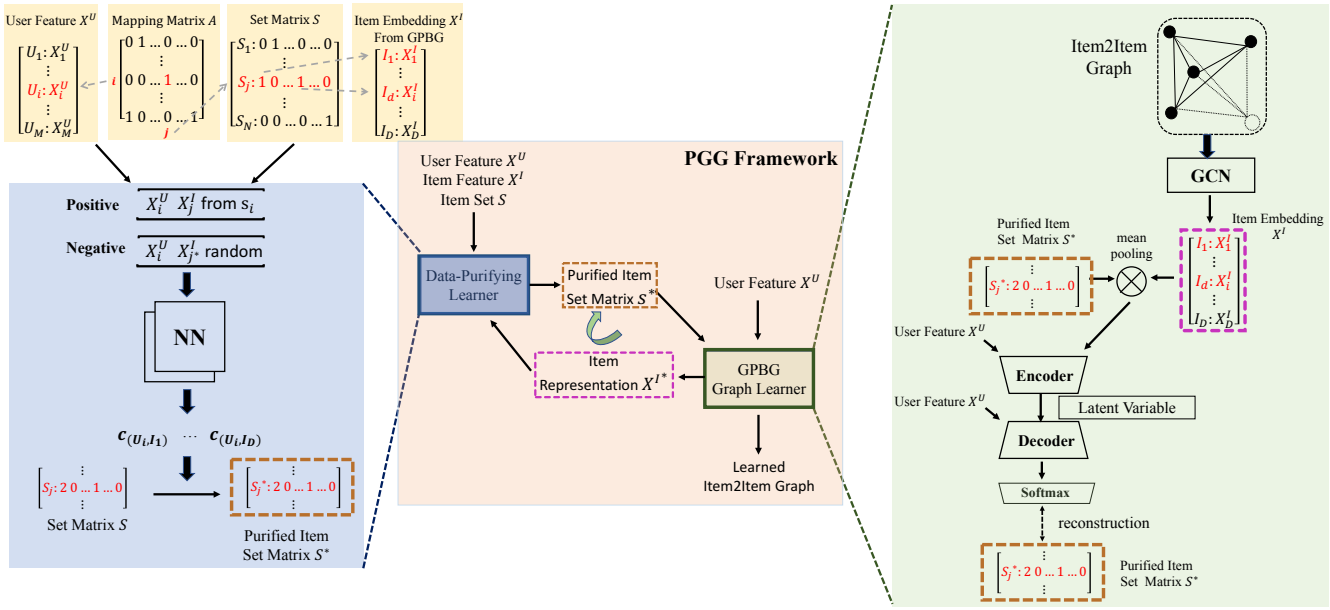

Figure 1: the overall framework of our proposed PGG: through pairwise positive and negative sample, we train a confidence neural network to capture the global preference of user-to-item. Catching a raw set s_1 of user U_1, we first re-sample a purified new set s_1^* according to the selection probability, then pass it with each item embedding learnt from I2I graph by GCN through the GPBG. Combining all the decoder's outputs, we obtain the item distribution in a set after a softmax layer. By maximizing likelihood of observed data, we can discovery the generation mechanism of personalized behavior and hence learn a best graph in turn. The model parameters are iteratively optimized along with clockwise data flow.

3.3 Data Re-sample with Item Reweighing

The probability space $\Omega(\mathbf{s}_j)$ (s.t. $a_{i,j} = 1$) is a always biased estimation of $P(\mathbf{s}|U_i)$ for each user U_i. More than that, the sampled input data is always too sparse to cover $P(\mathbf{s}|U_i)$. Thus, it's hard to remove the bias on $\Omega(\mathbf{s}_j)$ (s.t. $a_{i,j} = 1$). However, the exceptional choice of a user to specific item during decision process is the main reason of producing noise in $\mathbf{s}_j$. For example, some promotion policy would lead user to make an unexpected action, causing spurious association inside behavior data. Therefore, we propose to capture global preference to filter exception in user-to-item correlation first and use refined $\Omega(I_d)$ (s.t. $[\mathbf{A} \cdot \mathbf{S}]_{i,d} = 1$) to correct the probability space $\Omega(\mathbf{s}_j)$ (s.t. $a_{i,j} = 1$).

Specifically, we define a confidence value $c_{i,d}$ denoting the global preference between user U_i and item I_d, and learn it with a confidence neural network $f_c(\theta)$ using multilayer perception (MLP) [5] represented as,

$$\text{MLP}\left(\mathbf{x}, \theta = \left(\mathbf{W}^{(1)}, \cdots, \mathbf{W}^{(L_c)}\right)\right) = \sigma\left(\mathbf{W}^{(L_c)}\sigma\left(\cdots\sigma(\mathbf{W}^{(1)}\mathbf{x})\right)\right) \tag{2}$$

where $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function and L_c is the number of layers for this neural network. MLP has powerful capability to learn the similarity structure inside input data.

The user-to-item matrix $[\mathbf{A} \cdot \mathbf{S}]$ contains the partial connections between users and items. Together with the internal correlation of users (underlain in $\mathbf{X}^U$) and internal correlation of items (underlain in $\mathbf{X}^I$), we can use confidence network $f_c(\theta)$ to well capture the exact global preference of each user and item pair.

The confidence network takes $(\mathbf{x}_i^U, \mathbf{x}_d^I)$, the combination of user U_i's feature and item I_d's feature as input, and output the confidence value $c_{i,d} \in (0, 1)$ through sigmoid activation function. In practice, we don't calculate and store $[\mathbf{A} \cdot \mathbf{S}]$ explicitly. Randomly selecting an item I_d from $\mathbf{s}_j$, we consider $((\mathbf{x}_i^U, \mathbf{x}_d^I), 1)$ as positive sample, and assign arbitrary $d' \in D$, $((\mathbf{x}_i^U, \mathbf{x}_{d'}^I), 0)$ as negative sample, then use these pair samples to train the parameter θ to minimize the objective function $\mathcal{L}_c$ in Equation (3). Such a design can alleviate the problem of implicit feedback in recommender and effectively improve the computational efficiency

$$\min \mathcal{L}_c = -\log(f_c(\mathbf{x}_i^U, \mathbf{x}_d^I, \theta)) - \log(1 - f_c(\mathbf{x}_i^U, \mathbf{x}_{d'}^I, \theta)), \tag{3}$$

After training process, we can leverage the confidence value to reweigh the item count in original set and further re-sample a new but purified set data $\mathbf{s}_j^*$. The probability of each item I_d to be selected into the new data $\mathbf{s}_j^*$ is calculated by Equation (4),

$$\forall d \in D, \hat{P}(I_d) = \frac{s_{j,d} \cdot c_{i,d}}{\sum_{d=1}^{D} s_{j,d} \cdot c_{i,d}} \tag{4}$$

According to the selection probability, we sample $|\mathbf{s}_j|$ times so that $|\mathbf{s}_j| = |\mathbf{s}_j^*|$ and remove the exception action in observed behavior finally. That says we can better estimate $P(\mathbf{s}|U_i)$ with corrected $\Omega(\mathbf{s}_j^*)$ (s.t. $a_{i,j} = 1$).

3.4 Graph Based Personalized Behavior Generation

Assuming we have obtained the purified user log data $\mathbf{S}^*$, our next work is to design a model called GPBG capturing the generation function h from the I2I graph $\mathbf{G}$ to the personalized behavior data $\mathbf{s}_j^*$ conditioned on user U_i.

To catch the complex structural patterns among the nodes on graph, we first apply Graph Convolutional Networks (GCNs) with parameter χ to embed the local structural information of a node into its representation. The representation $\mathbf{x}_d^I(l+1)$ of the node d in the layer $(l+1)$ is transformed from the layer l by convolutional operator as

$$\mathbf{x}_d^{(l+1)} = \sigma\left(\sum_{j \in N_d} \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{G}}\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{x}_j^{(l)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)}\right), \tag{5}$$

where σ is a non-linear activation function, $\hat{\mathbf{G}}$ is the adjacency matrix with self loop in graph, $\hat{\mathbf{D}}$ is the degree matrix of $\hat{\mathbf{G}}$, N_d represents all the neighbours of node d (including node d itself), and $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the parameters of the convolution layer l. A GCN $f_g(\chi)$ of L_g layers can propagate message across L_g-hop neighbours. Initializing node features with $\mathbf{X}^I$, we eventually get a superior item embedding matrix $\mathbf{X}^{I*}$ with topology information.

With access to $\mathbf{X}^{I*}$, we further consider to generate set data $\mathbf{s}_j$ for user U_i. Compared to VAE method, CVAE studies the distribution of a high-dimensional variable conditioned on additional information. That is $P(\mathbf{s}^*|U_i)$ in our problem. CVAE first defines a prior distribution $p(\mathbf{z})$ and ensures that $q_\phi(z|\mathbf{s}^*, U_i)$ is similar to $p(\mathbf{z})$ by a learnable model $f_e(\phi)$ (encoder). Then it maximizes the likelihood of obeservational data by another learnable model $f_d(\psi)$ (decoder) to recover $P(\mathbf{s}^*|U_i)$. The whole process could be formulated in Equation (6).

$$\min\ KL[q_\phi(\mathbf{z}|\mathbf{s}^*, U_i)||p(z)] - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{s}^*,U_i)}[\log(p_\psi(\mathbf{s}^*|U_i, \mathbf{z}))], \tag{6}$$

To specify CVAE to learn $P(\mathbf{s}^*|U_i)$, we design a model in realization for personalized behavior generation. In our scenario, the latent variable $\mathbf{z}$ can be considered as the intention of making decision, and $\mathbf{z} \sim$ spherical Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. For each optimization step, we sample a real set $\mathbf{s}_j^*$ of user U_i from behavior log and take the mean pooling over the embeddings of items in the set, that is $\frac{\mathbf{s}_j^* \cdot \mathbf{X}^{I^*}}{\mathbf{s}_j^* \cdot \mathbf{1}}$, as representation of $\mathbf{s}_j^*$. Then we combine it with U_i's feature and input $\left(\frac{\mathbf{s}_j^* \cdot \mathbf{X}^{I^*}}{\mathbf{s}_j^* \cdot \mathbf{1}}, \mathbf{X}_i^U\right)$ into an encoder $f(\phi)$ composed of MLP. The encoder finally outputs a mean value vector $\boldsymbol{\mu}$ and a variance vector $\boldsymbol{\varphi}$ to represent $q(\mathbf{z}|\mathbf{s}_j^*, U_i)$ as a spherical Gaussian $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\varphi} \cdot \boldsymbol{I})$.

On the stage of decoding, that is the reconstruction process, we have to consider the non-sequential characteristics of set generation. Hence, we sample a $\mathbf{z}_j$ from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\varphi} \cdot \boldsymbol{I})$ first and combine it with

Algorithm 1 PurifiedGraph Generation (PGG)

Input: user-set mapping matrix $\mathbf{A}$, set matrix $\mathbf{S}$, user raw feature matrix $\mathbf{X}^U$ and item raw feature matrix$\mathbf{X}^I$
Output: Faithful Item2Item graph structure $\mathbf{G}$ over D items
Initial $\mathbf{G}$, parameters θ, χ, ϕ and ψ of confidence network, GCN, encoder and decoder respectively
while not converged **do**
  for $t = 1$ to $\tau^{(1)}$ **do**
    Sample positive batch $\{(\mathbf{x}_i^U, \mathbf{x}_d^I)_r, 1)\}_{r=1}^L$ ($[\mathbf{A} \cdot \mathbf{S}]_{i,d} >= 1$)
    Produce negative batch $\{(\mathbf{x}_i^U, \mathbf{x}_{d'}^I)_r, 1)\}_{r=1}^L$
    Calculate $\mathcal{L}_c$ according to Equation (3)
    Optimize θ to minimize $\mathcal{L}_c$
  end for
  for $t = 1$ to $\tau^{(2)}$ **do**
    Sample batch of data $\{(\mathbf{x}_i^U, \mathbf{s}_j)_r)\}_{r=1}^L$ ($\mathbf{A}_{i,j} >= 1$)
    Re-sample $\{(\mathbf{x}_i^U, \mathbf{s}_j^*)_r)\}_{r=1}^L$ according to confidence value
    Compute new item embedding matrix $\mathbf{X}^{I^*}$ with GCN
    Calculate $\mathcal{L}_s$ according to Equation (10)
    Optimize $\mathbf{G}$, χ, ϕ, ψ to minimize $\mathcal{L}_s$
    Update $\mathbf{X}^I$ by $\mathbf{X}^{I^*}$
  end for
end while
return: $\mathbf{G}$

U_i's feature and the embedding of each item I_d to input $(\mathbf{z}_j, \mathbf{X}_d^I, \mathbf{X}_i^U)$ into a decoder $f(\psi)$ composed of MLP respectively. Passing the concatenation of them through a softmax-layer, we obtain,

$$\mathbf{v} = \left(\frac{e^{f(\mathbf{z}_j, \mathbf{X}_d^I, \mathbf{X}_i^U, \psi)}}{\sum_{d'=1}^D e^{f(\mathbf{z}_j, \mathbf{X}_{d'}^I, \mathbf{X}_i^U, \psi)}}\right)_{d=1}^D \tag{7}$$

where $\mathbf{v}_d$ indicates the ratio of item I_d in generated set. To maximize the likelihood of observed data, we minimize the reconstruction loss between $\mathbf{v}$ and $\mathbf{s}_j^*$ with objective function $\mathcal{L}_{rc}$.

$$\min \mathcal{L}_{rc} = -\frac{\mathbf{s}_j^* \cdot \mathbf{v}}{||\mathbf{s}_j^*||_2 \cdot ||\mathbf{v}||_2} \tag{8}$$

And we take $\mathcal{L}_{kl}$ to minimize the KL divergence between $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\varphi} \cdot \boldsymbol{I})$ and $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ as widely used in VAE models.

$$\min \mathcal{L}_{kl} = ||-0.5 * (\mathbf{1} + \log(\boldsymbol{\varphi}) - \boldsymbol{\mu}^2 - \boldsymbol{\varphi})||_2. \tag{9}$$

By jointly optimizing I2I graph $\mathbf{G}$ and parameters χ, ϕ, ψ with $\mathcal{L}_s$ in graph based personalized behavior generation task, we eventually learn the mechanism h that makes the connection from I2I graph to set data, and hence can embed the complicated relationships of items from behavior log into the graph structure in an unsupervised way.

$$\min \mathcal{L}_s = \mathcal{L}_{rc} + \alpha\mathcal{L}_{kl} \tag{10}$$

3.5 Iterative Optimization of Two Module

Actually, the module of data re-sampling and graph learning benefit each other owing to their close interaction:

settings	1-2-2			1-2-3			1-3-2			1-3-3		
HR@ (%)	top10	top20	top30	top10	top20	top30	top10	top20	top30	top10	top20	top30
COOR	17.30	32.72	46.56	16.52	32.41	47.98	17.11	32.31	48.54	20.60	37.44	52.39
Session Based	17.72	32.92	47.11	17.27	33.18	48.44	18.34	34.93	50.72	19.82	38.09	53.26
GPBG	**18.75**	**34.88**	**49.29**	**20.29**	**37.27**	**49.95**	**19.22**	**37.15**	**52.08**	**21.77**	**39.71**	**54.83**
settings	3-2-2			3-2-3			3-3-2			3-3-3		
HR@(%)	top10	top20	top30	top10	top20	top30	top10	top20	top30	top10	top20	top30
COOR	17.55	32.25	46.26	16.37	32.49	46.65	18.42	33.97	48.87	19.63	36.32	51.98
Session Based	17.80	33.66	47.72	17.31	32.32	47.28	16.65	33.74	49.56	19.49	37.38	52.42
GPBG	**18.91**	**34.88**	**48.52**	**19.29**	**35.75**	**49.64**	**19.18**	**35.87**	**50.48**	**20.08**	**37.65**	**52.58**

Table 2: Experimental results in synthetic data. GPBG significantly outperform the others for each setting at all $\mathcal{K}$ values, because of its capability to learn high-ordered and nonlinear correlations.

- If there is less noise in observed data, we can generate a higher quality I2I graph, as well as the item embeddings;
- If more reasonable embedding space is provided, we can further learn more accurate confidence value and better purify the behavior data.

Hence, we optimize I2I graph structure $\mathbf{G}$, the confidence network $f_c(\theta)$ and GPBG $\{f_g(\chi), f_e(\phi), f_d(\psi)\}$ to achieve their best optimal solution as iterative process:
If all the parameters $\{\mathbf{G}(t), \theta(t), \chi(t), \phi(t), \psi(t)\}$ have been trained after t epochs; at epoch $(t+1)$, fixing $\{\mathbf{G}(t), \chi(t), \phi(t), \psi(t)\}$, we compute $\mathcal{L}_c$ based on user features $\mathbf{X}^U$ and item features $\mathbf{X}^I(t) = f_c(G(t), \chi(t))$ first and update $\theta(t+1)$ for $\tau^{(1)}$ steps to minimize $\mathcal{L}_c$; then fixing $\theta^{(t+1)}$, we re-sample new behaviors $\mathbf{S}^*(t)$ using $f_c(\theta(t+1))$ first, and update $\{\mathbf{G}(t+1), \chi(t+1), \phi(t+1), \psi(t+1)\}$ for $\tau^{(2)}$ steps to minimize $\mathcal{L}_s$ in turn. Finally, we summary our algorithm of learning faithful Item2Item graph in Algorithm 1.

4 EXPERIMENT

In this section, we evaluate the effectiveness of proposed **PGG** to generate a faithful I2I graph from decision-making behavior. We carry out primary studies on both synthetic data and real datasets to demonstrate the advantage of learnable graph structure and the harm of noisy sparse data respectively. And we conduct extensive experiments on two large scale real recommendation scenarios to show the superiority of our algorithm.

Baselines. To explore the impact of each component in our model, we use the following models as baselines,

- **Co-occurrence Graph (COOR)**: A handy constructed graph according to the rule that connects edges linking items with highly co-occurrence frequency;
- **SWING**: SWING filters the data noise with cooperation ability of human by counting the "SWING" structures in the user-item graph as the relatedness between items pair.
- **LightGCN[7]**: A powerful GCN model that is more appropriate for collaborative filtering in recommender system. It takes (local) user-item graph, i.e. user-set, as supervised information and ablate noise in heterogeneous structure. After learning item embeddings, we construct the graph based on embedding similarity ;
- **Session Based Embedding Learning**: This method learns the item embeddings on fixed graph structure by supervision of session data, and then construct the graph based on the similarity of embedding pair;
- **GPBG**: The graph structure learning method designed in this work (without data re-sampling process);
- **PGG**: Our proposed complete framework.

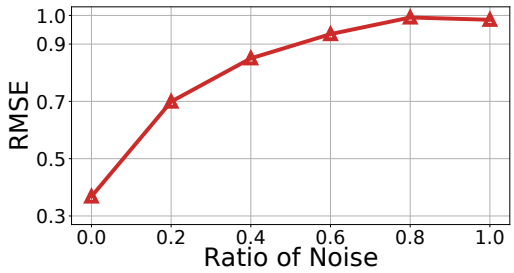

Figure 2: Noise's impact on indentifying use behaviors.

For clear discussion in later sections, we divide all the methods into three types: co-occurred (COOR, SWING), embedding based (Session Based, LightGCN), structure learning (GPBG, PGG). In the same type, SWING/LightGCN/PGG has a respective mechanism to resist the impact of noise on the learned graph. That is saying these baselines cover all typical methods (containing the novel learnable framework we proposed) for generating the I2I graph. All the implementations of PGG and baselines are realised by Pytorch[13] in this work.

Metric (Item based Top-$\mathcal{K}$ Matching). To evaluate the performance of the I2I graph, we introduce the $HitRate(HR)$ index that is commonly accepted in recommender system. Classical HR measures the recall ratio of the item list recommended to a specific user. But in our scenario, we want to evaluate the quality of graph structure, regardless of user feature. Hence, we redefine the HR index in the item based top-$\mathcal{K}$ matching task.

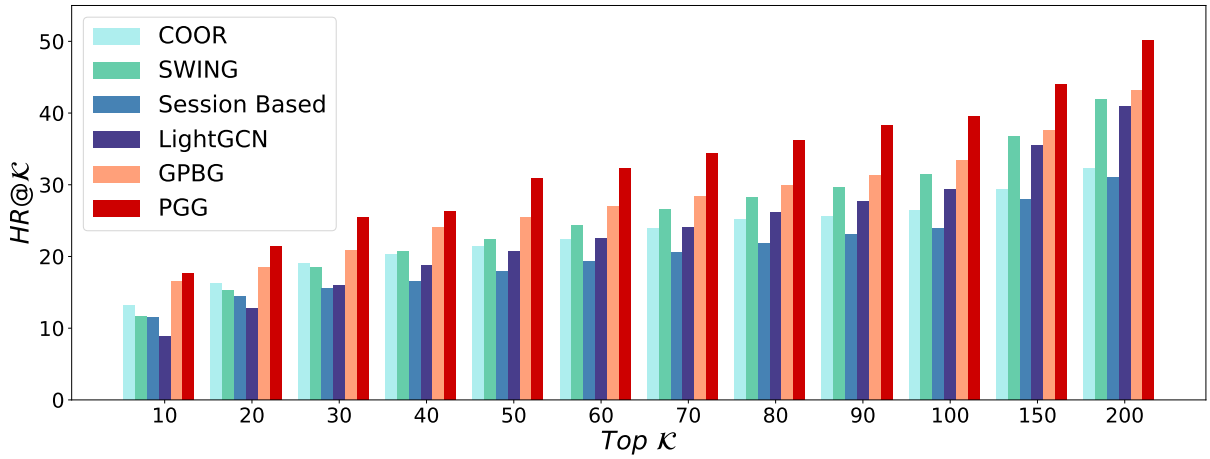

Figure 3: Performance of methods in Kwai dataset. Our PGG has gained a huge margin on the basis of competitive GPBG owing to data re-sampling process for cleaning up noise.

For set $\mathbf{s}_j$, we randomly split it into a ground truth $\mathcal{I}_j$ and other trigger items $\mathbf{s}_j - \{\mathcal{I}_j\}$. Item based top-$\mathcal{K}$ matching is to recall $\mathcal{I}_j$ from the top $\mathcal{K}$ nearest items to $\mathbf{s}_j - \{\mathcal{I}_j\}$ (except for $I_{d'} \in \mathbf{s}_j - \{\mathcal{I}_j\}$). Then we can define $HR@\mathcal{K}$ as follows:

$$HR@\mathcal{K} = \frac{\sum_{j=1}^{N_{test}} |\{I_d | I_d \in top\mathcal{K}_{small} Dis(I_d, \mathbf{s}_j - \{\mathcal{I}_j\})\} \cap \{\mathcal{I}_j\}|}{N_{test}} \tag{11}$$

$$Dis(I_d, \mathbf{s}_j - \{\mathcal{I}_j\}) = \frac{\sum_{I_{d'} \in \mathbf{s}_j - \{\mathcal{I}_j\}} Cosine(\mathbf{x}_d^I, \mathbf{x}_{d'}^I)}{|\mathbf{s}_j - \{\mathcal{I}_j\}|}. \tag{12}$$

We take the **Cosine Similarity** between item embeddings as distance measure. The performance of graph structure is actually reflected in the rationality of the whole embedding space. So we enumerate different values of $\mathcal{K}$ to make the evaluation. And for a fair comparison, we randomly select a fixed dataset to learn embeddings of different graphs after obtaining them by baselines.

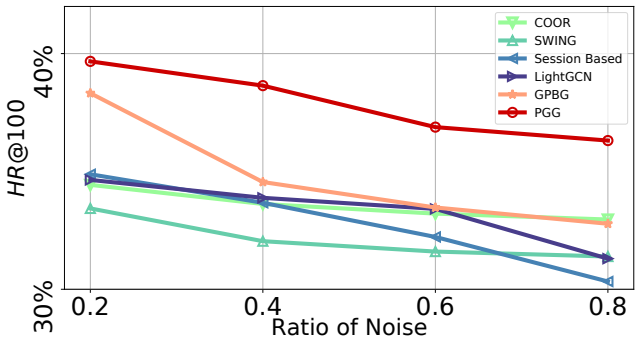

Figure 4: The robustness of different methods to noise ratio.

4.1 Simulation Experiment

4.1.1 High-ordered weighted random walk (HWRW). Traditional weighed random walk generates a path via walking among the nodes with the Markov transition probability calculated by edge weights. Further, we introduce high-ordered weighed random walk to simulate more complex correlations of nodes through: 1)assigning all nodes into 2 types; 2) defining a moving probability function $\pi(u, v)$ from node u to node v as,

$$\pi(u, v) = \frac{C(pa(u), u, v) * w_{u,v}}{\sum_{v' \in N(u)} C(pa(u), u, v') * w_{u,v'}}. \tag{13}$$

$$C(pa(u), u, v) = \begin{cases} w_{u,v} * q_2 & If\ d(pa(u), v) <= 1 \quad and \\ & pa(u),\ u,\ v\ are\ the\ same\ type \\ w_{u,v} * q_1 & If\ u,\ v\ are\ the\ same\ type \\ w_{u,v} & Otherwise \end{cases} \tag{14}$$

where $w_{u,v}$ is the weight of edge (u, v), $N(u)$ is the neighbour set of node u, $pa(u)$ is the parent node of u if u is not the first node in path, $C(pa(u), u, v)$ is a term depending on node u, v and $pa(u)$ according to the rule in Equation (14) (coefficients $q_2 > q_1 > 1$, $d(pa(u), v)$ is the shortest path from $pa(u)$ to u).

Obviously, moving probability $\pi(u, v)$ is a high-ordered and non-linear function. And nodes with the same type are more likely to gather in one path with the increased value of q_1 and q_2.

To generate synthetic data, we randomly sample a 100 node undirected graph and connect each node pair with 50% probability. The edge weight is assigned from 1 to 100. The total 100 nodes are equally divided into 2 types. For realistic simulation, we also define a coefficient q_0 controlling the prior distribution discrepancy of 2 type nodes in synthetic data (type one being a start node of the random walk has q_0 times probability than type two). By adjusting coefficients q_0, q_1 and q_2, we finally generate PGG synthetic datasets underlying different correlations of nodes.

4.1.2 Experimental result. ha Because there is no user feature in synthetic data, we compare our GPBG with co-occurrence graph

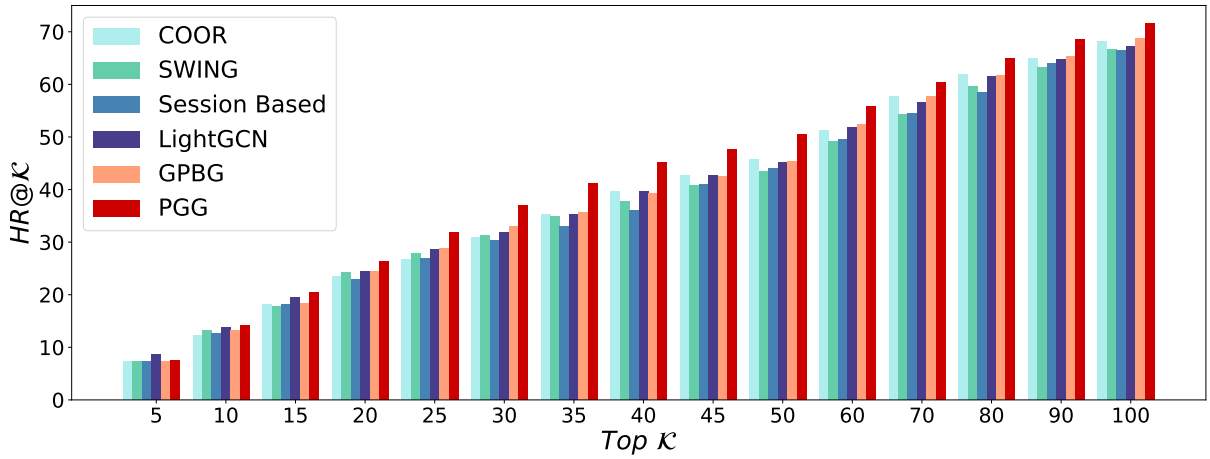

Figure 5: Performance of methods in CloudTheme Click Dataset. Iterative optimization of data purifying and graph learning helps to detect subtle noise signal and achieve best solution of Item2Item graph.

and session based embedding learning to verify the availability of unsupervised graph structure from observed set data. We evaluate the methods in the item based Top-$\mathcal{K}$ matching task and report the *HR* performance of them in 8 different correlation strength settings (12800 samples in each setting and 8:2 for training and test). For the results in Table 2, we can see that:

- Session based methods have better performance than COOR graph because it can learn the high-ordered and nonlinear correlations in data;
- GPBG further outperform session based with the advantage of achieving best solution by optimizing learnable graph structure;
- Although the task becomes easier as the strength of the relationship increases due to the aggregation of the same type nodes, GPBG gains a significant improvement margin in $\mathcal{K} = 10, 20, 30$, implying ours can really encode the complex correlation into the graph structure.

4.2 Videos Watching on Short Video Platform

4.2.1 Dataset. In this section, our experiments are conducted on a large-scale fine-grained dataset from Kwai. Kwai is one of the most popular short video platforms in China. Our edge data is collected from the user-video interaction log of one scene in Kwai, which contains nearly 400 million actions of different types in 7 days. The action types reflect deeper and deeper user-video relationship from just showing to playing the whole video. Besides, attributes of 2.1 million users and 1.7 million videos are also included. It is worth noting that, there are many promotion strategies can infect the action of users, for example, a user may be informed to watch some specific videos to earn some bonus.

In a large-scale real-world dataset, noise is the major obstacle preventing us from capturing sound relationships. Partial user-video edges with types: 'show', 'effect play', 'deep play', 'full play', have reliability score 0, 1, 2, 3 in our data. Considering the 'show' as noise, we combine it in different proportions with the other types (clean data) composing 6 datasets with different noise levels. Then we conduct SVD method to acquire user and item representations, and evaluate by RMSE metric of reconstructing 'effect play', 'deep play', 'full play' edges in each dataset. The results in Figure 2 say the user behaviors become harder to identify as the increase of noise data, leading it harder to understand the item-item relationships from data.

4.2.2 Experimental Result. Taking use of fine-grained information in Kwai data, we design two experimental settings to compare the performance of different methods (co-occurred/embedding based/structure learning, with/without denoise mechanism) to generate the I2I graph. As the first, we select training data containing 662 users, 5900 videos and average set length $\sim$ 11 with a noise ratio 0.57. After generating graphs by baselines and ours, we report their test performance in item-based Top-$\mathcal{K}$ matching task defined ahead. As shown in Figure 3,

- All the methods with denoise mechanism clearly outperform the other (of the same type) without denoise;
- The co-occurred methods even have higher *HR*@$\mathcal{K}$ than embedding based methods, because of the failure of function calculating edge weights by item embeddings;
- GPBG can fully encode complicated relationships of items into the graph structure, and hence obtain a significant improvement;
- Moreover, our PGG further achieves a remarkable performance for all $\mathcal{K}$, through sufficiently getting rid of the noise signal (**+18%** relative to GPBG).

As the second, we evaluate the impact of noise ratio on the performance of baselines and ours. We sample 3213 user sets and rank them by the noise ratio in each set at first; then we select 662 sets with ratios closest to 0.2/0.4/0.6/0.8 and form 4 different training data. Through similar graph generation and test process, we report the curves of *HR*@100 for all the methods to noise ratio in Figure 4. Compared to other lines, either in a lower position or sharply slide in Figure 4, our red line keeps the highest position

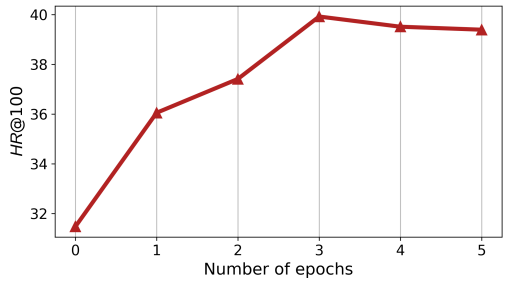

Figure 6: Iterative Optimization of data re-sampling and graph learning (each epoch is an iterative loop).

with quite a smooth trend, presenting the strong robustness of PGG to the agnostic noise environment.

Further, we confirm whether the two modules of PGG architecture, data re-sampling and graph learning, promote each other empirically. The curve in Figure 6 refers to the optimization process of our approach, and can support our claim well.

4.3 Product Click on E-commerce Platform

In this section, we consider studying our problem in electronic commerce, another typical recommender application. The public "Cloud Theme Click Dataset"[3] collected in Taobao app of Alibaba represents an important recommendation procedure in the mobile terminal of electronic business, including more than 4 million purchase histories of users for one month before the promotion started. Usually, a user tends to make decisions much more cautiously before promotion, waiting for discount activity. So compared to the Kwai dataset, the released shopping data carries less noise information.

We fetch the data from one day's record, containing 568 users, 684 items and each set length exceeds 5. According to the results reported in Figure 5,

- For co-occurred method, SWING loss its effectiveness as the increase of $\mathcal{K}$ value and even hurts the item relationships in clean data, implying its rule-based strategy is not absolutely reasonable;
- For embedding based method, LightGCN shows few advantages compared to session based method, not able to bring any more improvetment;
- However, our PGG still detects the subtle noise signal and has further outperformed the competitive GPBG, indicating ours can achieve the best solution via iterative optimization of data purifying and graph learning.

5 CONCLUSION

In this paper, we investigate how to generate faithful Item2Item graph, a powerful tool for highly matching in the first step of recommender system, from user-item interaction behavior. To accomplish this goal, we propose a novel framework **PGG** with iterative optimization of two cooperative modules: a confidence network for noisy data purification and a graph-based personalized behavior generation model (GPBG) for learning graph structure. Our framework shows much more salient performance in real-world applications, compared to competitive baselines. In a large-scale short video dataset from Kwai, ours significantly improve the quality of the I2I graph with strong robustness to agnostic environment.

ACKNOWLEDGMENTS

This work was supported in part by National Key R&D Program of China (No. 2018AAA0102004, No. 2020AAA0106300), National Natural Science Foundation of China (No. U1936219, 61521002, 61772304), Beijing Academy of Artificial Intelligence (BAAI), and a grant from the Institute for Guo Qiang, Tsinghua University.

REFERENCES

[1] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. 2018. Netgan: Generating graphs via random walks. In *International Conference on Machine Learning*. PMLR, 610–619.
[2] Nicola De Cao and Thomas Kipf. 2018. MolGAN: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973* (2018).
[3] Zhengxiao Du, Xiaowei Wang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Sequential scenario-specific meta learner for online recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2895–2904.
[4] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. 2019. Learning discrete structures for graph neural networks. In *International conference on machine learning*. PMLR, 1972–1982.
[5] Matt W Gardner and SR Dorling. 1998. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment* 32, 14-15 (1998), 2627–2636.
[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.
[7] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 639–648.
[8] Yue He, Peng Cui, Jianxin Ma, Hao Zou, and Philip S. Yu. 2020. Learning Stable Graphs from Multiple Environments with Selection Bias. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.
[9] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).
[10] Thomas N Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. (2016).
[11] Tengfei Ma, Jie Chen, and Cao Xiao. 2018. Constrained generation of semantically valid graphs via regularizing variational autoencoders. *arXiv preprint arXiv:1809.02630* (2018).
[12] Yutaka Matsuo and Mitsuru Ishizuka. 2004. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools* 13, 01 (2004), 157–169.
[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703* (2019).
[14] Darwin Saire Pilco and Adín Ramírez Rivera. 2019. Graph learning network: A structure learning algorithm. *arXiv preprint arXiv:1905.12665* (2019).
[15] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems* 28 (2015), 3483–3491.
[16] Rakshit Trivedi, Jiachen Yang, and Hongyuan Zha. 2020. Graphopt: Learning optimization models of graph formation. In *International Conference on Machine Learning*. PMLR, 9603–9613.
[17] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 839–848.
[18] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 169–178.
[19] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International Conference on Machine Learning*. PMLR, 5708–5717.