A Pre-training Strategy for Recommendation

Susen Yang∗
University of Science and Technology of China

Yong Liu†
Nanyang Technological University

Chenyi Lei
Alibaba Group

Guoxin Wang
Alibaba Group

Haihong Tang
Alibaba Group

Juyong Zhang
University of Science and Technology of China

Chunyan Miao
Nanyang Technological University

ABSTRACT

The side information of items has been shown to be effective in building the recommendation systems. Various methods have been developed to exploit the item side information for learning users’ preferences on items. Differing from previous work, this paper focuses on developing an unsupervised pre-training strategy, which can exploit the items’ multimodality side information (e.g., text and images) to learn the item representations that may benefit downstream applications, such as personalized item recommendation and click-through ratio prediction. Firstly, we employ a multimodal graph to describe the relationships between items and their multimodal feature information. Then, we propose a novel graph neural network, named Multimodal Graph-BERT (MG-BERT), to learn the item representations based on the item multimodal graph. Specifically, MG-BERT is trained by solving the following two graph reconstruction problems, i.e., graph structure reconstruction and masked node feature reconstruction. Experimental results on real datasets demonstrate that the proposed MG-BERT can effectively exploit the multimodality information of items to help downstream applications.

ACM Reference Format:
Susen Yang, Yong Liu, Chenyi Lei, Guoxin Wang, Haihong Tang, Juyong Zhang, and Chunyan Miao. 2020. A Pre-training Strategy for Recommendation. In Proceedings of ACM Conference (Conference’17). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Personalized recommendation systems have attracted lots of research attentions in recent years. One group of recommendation methods are based on the collaborative filtering techniques, e.g., matrix factorization [26]. The other group of methods are based on the deep learning techniques [38], e.g., convolutional neural networks [30] and recurrent neural networks [14], and graph neural networks [34]. To improve the recommendation performances, the multimodality side information (e.g., the textual descriptions, images, and videos) of items has been exploited [27]. How to effectively exploit the multimodality information of items becomes the key to success in building recommendation systems.

Traditional methods exploit the item side information by manual feature engineering [20] and then employ factorization machine [25] or gradient boosting machine [4] to predict users’ preferences on items. The deep learning-based methods enable the development of end-to-end recommendation systems. They leverage the strong representation learning ability of neural networks to exploit item side information for learning the user and item representations. However, for different applications, similar operations are repeatedly performed on items’ side information. Due to the limitation of manpower and computation resources, each application would only consider a specific type of side information of items, instead of all the multimodality side information.

Inspired by the successes of unsupervised pre-training strategies in computer vision (CV) and natural language processing (NLP) tasks [7, 8, 10], we propose to develop an unsupervised pre-training strategy that can aggregate the multimodality information of items to learn the item representations, which may benefit downstream tasks, e.g., item recommendation and click-through ratio (CTR) prediction. Specifically, we use a homogeneous graph to describe the relationships between items, where the features of each node (i.e., item) include its multimodality information. Then, we propose to pre-train the graph neural network (GNN) on the item multimodal graph to enable the GNN model to capture both the item relationships and the multimodality information of items.

The major contributions made in this paper are as follows:

- We propose a novel GNN pre-training framework, namely Multimodal Graph BERT (MG-BERT), to exploit the items’ multimodality information under the unsupervised learning paradigm. To the best of our knowledge, this is the first deep pre-training method developed to exploit the multimodality side information of items for recommendation.
- We decompose the learning objective of MG-BERT into two subproblems, i.e., graph structure reconstruction, and masked node feature reconstruction. To handle large-scale graph data, we develop the Mini-batch Contextual Neighbors
Figure 1: (a) The framework of the proposed MG-BERT model. From left to right, it contains four components: contextual neighbors sampling, node embedding initialization, transformer-based graph encoder, and graph reconstruction. GSR and NFR denote the graph structure reconstruction task and masked node feature reconstruction task, respectively. (b) The node embedding is initialized by considering the node’s multimodal features, position-id embedding, and role-label embedding.

2 RELATED WORK

Pre-training is widely applied in CV tasks [10]. A general pretraining paradigm for CV tasks is firstly training a model on the ImageNet dataset [6] and then fine-tuning the pre-trained model in a specific downstream task. NLP is the other domain where pre-training is usually adopted. The shallow pre-training methods, e.g., word2vec [21] and GloVe [22], learn word representations based on the word co-occurrence patterns in a corpus of documents. Recently, significant progress has been made in developing deep pre-training models for NLP tasks. Representative methods include BERT [7] and XLNET [36] that use attention mechanisms to learn word representations.

For graph data, the graph embedding techniques have attracted lots of research attentions [3]. DeepWalk [23] and LINE [29] are the representative shallow graph embedding methods. The recent popularity of GNNs motivates the development of pre-training strategies for GNN models. Generally, these methods pre-train GNNs by solving the graph reconstruction problems [19]. For example, GraphSAGE [9] is a general inductive framework that exploits node features to generate the node embeddings by sampling and aggregating features from a node’s local neighborhood. The Deep Graph Infomax model [32] aims to maximize the mutual information between the node representations and the representation of the graph. Recently, self-supervised learning [17] is employed to simultaneously pre-train GNNs at both the node and graph levels [15]. Similarly, the generative framework GPT-GNN [16] employs a self-supervised attributed graph generation task to pre-train GNNs, by effectively capturing both the semantic and structural properties of the graph. Moreover, the recent work [24] proposes a self-supervised graph neural network to capture the universal network topological properties across multiple networks.

3 THE PROPOSED MODEL

This section firstly introduces some preliminary background and then describes the details of the MG-BERT model.

3.1 Preliminaries

In this work, we use a homogeneous graph \( G = (V, E) \) to describe the relations between items, as well as their multimodality side information (e.g., text descriptions and images). Here, \( V \) denotes the set of items, and \( E \) denotes the set of edges between items. In \( G \), each node \( h \) has multiple types of modality information. We denote the \( i \)-th modality feature of \( h \) by \( x^i_h \), and the number of modality type by \( m \). Moreover, we denote the nearest neighbors of \( h \) by \( N_h \), and use \( \omega_{ht} \) to denote the weight of the edge between two nodes \( h \) and \( t \), where \( \omega_{ht} > 0 \). For a node \( h \), we use \( C_h \) to denote its contextual neighbors selected by the MCNSampling algorithm. Given the item graph \( G \) and the contextual neighbors of each node, MG-BERT aims to obtain the node representations that can capture the multimodality information of nodes and the graph structure information. Then, the learned node representations can be applied in downstream tasks directly or with necessary adjustments, e.g., fine-tuning.

3.2 MG-BERT

Figure 1 shows the framework of MG-BERT. We can note that MG-BERT contains four main components: 1) contextual neighbors sampling, 2) node embedding initialization, 3) transformer-based encoder, and 4) graph reconstruction. Next, we will introduce the details of each component.
3.3 Contextual Neighbors Sampling

For each node $h$, there exist some most relevant nodes in the graph that may help enrich the representation of $h$. These relevant nodes are referred as the contextual neighbors of $h$. To efficiently select the contextual neighbors for a batch of nodes during the training of MG-BERT, we develop the MCNSampling Algorithm, which iteratively samples a list of neighboring nodes for a target node $h$ with a sampling depth $K$. Let $S_h^{K-1}$ denote the list of nodes sampled at the $(K-1)$-th step. For each node $t$ in $S_h^{K-1}$, we randomly sample $n_k$ nodes with replacement from $N_t$ at the $k$-th step. The probability that a node $t' \in N_t$ would be sampled is proportional to the weight $\omega_{tt'}$ of the edge between $t$ and $t'$. Note that a node may appear multiple times in the list $S_h^{K-1}$. In the MCNSampling Algorithm, we treat the nodes in $S_h^{K-1}$ as “different nodes” and perform the sampling process.

In this work, we select the contextual neighbors by considering 1) the sampled frequency of a node, and 2) the distance between a sampled node and the target node in the sampling process. For every node $t \in \mathcal{V} \setminus h$, we empirically define its importance to the target node $h$ at the $k$-th sampling step as follows,

$$s_{t}^{k} = f_{t}^{k} \cdot (K - k + 1),$$

where $f_{t}^{k}$ denotes the appearing times of $t$ in the list $S_h^{k}$. The motivation is that a node is more relevant to the target node if it has a higher sampling frequency and a shorter distance to the target node in the sampling process. The final importance score of a node $t$ is defined as $s_{t} = \sum_{k=1}^{K} f_{t}^{k}$. Then, we sort the nodes in $\mathcal{V} \setminus h$ according to their importance scores in descending order and choose $S$ top-ranked nodes as the contextual neighbors of $h$. The details of the MCNSampling algorithm are summarized in Algorithm 1.

3.4 Node Embedding Initialization

The input of MG-BERT is the concatenation of the target node $h$ and its ordered contextual neighbors $C_h$, which is denoted by $I_h = [h_0, h_1, h_2, \ldots, h_S]$. Here, $h_0$ is the target node $h$, and $h_j$ is the $j$-th node in $C_h$ when $j > 0$. For each node $t \in I_h$, we apply the attention mechanism to obtain its multimodal representation $M_t$ as follows,

$$X_t = x_t W_M + b_M,$n
$$

$$X_t = X_t^{1} \odot X_t^{2} \odot \cdots \odot X_t^{m},$$

$$\alpha_t = \text{softmax}(\text{tan}(X_t W_s + b_s),$$

$$M_t = \sum_{i}^{m} \alpha_t^i X_t^i,$$

where $W_M \in \mathbb{R}^{d_f \times d_0}$ and $b_M \in \mathbb{R}^{1 \times d_0}$ denote weight matrix and bias term for the $t$-th modality, $W_s \in \mathbb{R}^{(md_f) \times m}$ and $b_s \in \mathbb{R}^{1 \times m}$ denote weight matrix and bias term for attention mechanism. $\odot$ is the concatenation operation. $\alpha_t^i$ denotes the $i$-th element of $\alpha_t$.

The position of a node in the input list $I_h$ reflects its importance to the target node $h$. Thus, we argue that the order of nodes in an input list is important in learning the node representations. The following position-id embedding is used to identify the node order information of an input list,

$$P_t = \text{P-Embedding}(p(t)),$$

where $p(t)$ denote the position id of the node $t$ in $I_h$. $P_t \in \mathbb{R}^{1 \times d_0}$ denotes the position-based embedding for $t$.

For the input list $I_h$, the main objective is to obtain the representation of the target node $h$. Intuitively, the target node and its contextual neighbors should play different roles in the model. To identify the role differences, we add the following role-based embedding to each node $t \in I_h$,

$$R_t = \text{R-Embedding}(r(t)),$$

where $r(t)$ and $R_t \in \mathbb{R}^{1 \times d_0}$ denote the role label and role-based embedding of the node $t$ respectively. In practice, we set the role label of the target node as “Target” and the role labels of the contextual neighbors as “Context”.

Based on the embeddings stated above, we aggregate them together to define the initial input embedding for a node $t \in I_h$ as follows,

$$H_t^0 = \text{Aggregate}(M_t, P_t, R_t).$$

In this work, we simply define the $\text{Aggregate}(\cdot)$ function as the vector summation. The initial input embeddings for the nodes in the input list $I_h$ can be stacked into a matrix $H^0 = [H_t^0, H_t^1, \ldots, H_t^S] \in \mathbb{R}^{(S+1) \times d_0}$.

3.5 Transformer-based Graph Encoder

The Transformer framework [31] is then to model the mutual influences between a node and its contextual neighbors. Given the node representations $H^{(t-1)}$ at the $(t-1)$-th layer, the output at the $t$-th
layer of original Transformer model is defined as follows,

\[
H^t = \text{FFN} \left( \text{softmax} \left( \frac{QK^T}{\sqrt{d_h}} \right) V \right),
\]

\[
Q = H^{t-1}W^t_Q, \quad K = H^{t-1}W^t_K, \quad V = H^{t-1}W^t_V,
\]

where \( W^t_Q, W^t_K, W^t_V \in \mathbb{R}^{d_h \times d_h} \) denote the weight matrices, \( \text{FFN}(\cdot) \) is the feed forward network. Here, we omit the residual network in the formula for convenience.

For the target node \( h \), there may exist some sampled nodes in \( C_h \), whose representations are similar with the representation of \( h \). Assume that all the sampled contextual neighbors are relevant to the target node. We hope the proposed model can capture the diversity of the sampled contextual neighbors, by concentrating on the nodes that are relevant but not very similar with the target node. To achieve this objective, we design a diversity-promoting attention mechanism and include it into the attention network of Transformer,

\[
S = H^{t-1}\mathbf{W}^t_s,
\]

\[
U_1 = \text{softmax}(E - \frac{SS^T}{||S||_2 ||S||_2} + I),
\]

\[
U_2 = \text{softmax}(\frac{QK^T}{\sqrt{d_h}}),
\]

\[
H^t = \text{FFN} \left( (\beta U_1 + (1 - \beta)U_2)V \right).
\]

where \( W^t_s \in \mathbb{R}^{d_h \times d_h} \) is the weight matrix, \( E \in \mathbb{R}^{(S+1) \times (S+1)} \) is a matrix where all the elements are 1, \( ||S||_2 \in \mathbb{R}^{(S+1) \times 1} \) denotes the \( \ell_2 \) row norm of \( S \), and \( I \in \mathbb{R}^{(S+1) \times (S+1)} \) denotes the identity matrix. Note that the larger the similarity of two nodes, the smaller the attention weight between them in \( U_1 \). The objective of adding \( I \) in the definition of \( U_1 \) is to include the node’s self information. \( \beta \) is a constant controlling the contributions of the two attention weights.

After obtaining the output \( H^t \) at the last layer of the encoder, we obtain \( H_0^t \) as the representation of target node \( h \) which is denoted as \( h \) for simplicity. Then, \( H^t \) will be used in the following pre-training tasks.

### 3.6 Graph Reconstruction

The proposed MG-BERT model is pre-trained with the following two tasks: 1) graph structure reconstruction, and 2) masked node feature reconstruction. To ensure the learned node representations can capture the graph structure information, we define the following loss function,

\[
\ell_{\text{edge}} = \frac{1}{|V|} \sum_{h \in V} \frac{1}{|N_h|} \sum_{t \in N_h} \left[ -\log(\sigma(\frac{h^t}{||h||_2}) \right] - Q \cdot E_{\pi^t} \cdot p_{n}(t) \log(\sigma(\frac{h^t}{||h||_2})) \right],
\]

where \( \sigma(\cdot) \) is the sigmoid function, \( p_{n} \) and \( Q \) denote the negative sampling distribution and the number of negative samples, respectively.

The node feature reconstruction task focuses on capturing the multimodal features in the learned node representations. Previous methods, e.g., GRAPH-BERT [37], design an attribute reconstruction task without masking operations. Thus, the models’ abilities in aggregating the features of different nodes may be limited. In this work, we design a masked node feature reconstruction task, which aims to reconstruct the features of masked nodes by other non-masked nodes in \( I_h \). As the representation of the target node \( h \) is needed to reconstruct the graph structure in Eq. (8), we do not apply the masking operation to \( h \) (i.e., \( h_0 \) in \( I_h \)). Following [7], we randomly choose 20% of nodes in the list \( I_h \) for masking. If the node \( t \) is chosen, we replace \( t \) with: 1) the [Mask] node 80% of the time, (2) a random node 10% of the time, and (3) the unchanged node \( 10\% \) of the time. Then, the masked item list will be input to the model, and the output \( H^t \) will be used to reconstruct the multimodal features of the masked nodes. We set the input features of the [Mask] node to 0, and define the feature reconstruction loss as follows,

\[
\ell_{\text{feature}} = \frac{1}{|V|} \sum_{h \in V} \frac{1}{|M_h|} \sum_{t \in M_h} \| H^t - W^t_r \|^2_2,
\]

where \( M_h \) denotes the set of masked nodes in \( I_h \), \( H^t \) denotes the representation of \( t \) in \( H^t \), and \( W^t_r \) is the weight matrix for the \( t \)-th modality information reconstruction.

The model parameters of MG-BERT can be learned by minimizing the following objective function,

\[
\ell_{\text{edge}} + \lambda \ell_{\text{feature}},
\]

where \( \lambda \) is empirically set to 1 in the experiments. The entire framework can be effectively trained by the end-to-end backpropagation algorithm. To make the training of the model more stable, a mini-batch of nodes are randomly sampled to update the model. When applying the pre-trained MG-BERT model in downstream tasks, the learned node representations can be either fed into the new tasks directly or with necessary adjustment (e.g., fine-tuning).

### 4 EXPERIMENTS

In this section, we present the details of the experimental results.

#### 4.1 Experiment Settings

4.1.1 Datasets. The experiments are performed on the Amazon review dataset [11] and Movielens-20M dataset. Two downstream tasks, i.e., item recommendation, and CTR prediction, are studied to demonstrate the effectiveness of the proposed MG-BERT model.

We choose the following 5-score review subsets for experiments, i.e., “Video Games”, “Toys and Games”, and “Tools and Home Improvement” (respectively denoted by VG, TG, and THI). The rating data generated before 2015-01-01 are used for building the product
Then, the averaged visual features of these keyframes extracted
by the pre-trained Inception-v4 network are used as the visual
modality feature from its images. In downstream tasks, we convert all
the observed review ratings to be positive interactions and filter out
the products that are not included in the product graph.

For Movielens-20M dataset (denoted by ML), we construct
the movie graph based on the tags of movies. Two movies will be
connected in the graph if they have the same tag. The weight of the
dge between two movies is defined based on the number of tags they both have. The rating data generated since 2008-01-01 are used to
evaluate the performances of downstream tasks, where we keep
ratings larger than 3 as positive interactions. We collect the movie
trailers from Youtube\(^1\) and extract keyframes for each movie trailer.
Then, the averaged visual features of these keyframes extracted
by the pre-trained Inception-v4 network are used as the visual
modality of the movie. Moreover, we separate the audio track from the
movie trailer with FFmpeg\(^2\) and adopt VGGish [13] to obtain the
acoustic modality of the movie. The textual modality information of
a movie is derived by applying the pre-trained BERT model on the
movie text description collected from TMDB\(^3\). Table 1 summarizes the
statistics of these experimental datasets. More details of the
experimental settings can be found in the Appendix.

### 4.1.2 Setup and Metrics.

| Datasets | Metrics  | Random | DeepWalk | LINE | TransAE | GraphSAGE | GRAPH-BERT | GPT-GNN | MG-BERT |
|----------|----------|--------|----------|------|---------|-----------|------------|---------|---------|
| VG       | REC-R@20 | 0.2742 | 0.3179   | 0.3169 | 0.2903  | 0.2821    | 0.3330     | 0.3269  | 0.3405  |
|          | REC-N@20 | 0.1494 | 0.1711   | 0.1690 | 0.1480  | 0.1522    | 0.1767     | 0.1636  | 0.1890  |
|          | CTR-AUC  | 0.7311 | 0.768    | 0.7762 | 0.7675  | 0.7674    | 0.7746     | 0.7746  | 0.7990  |
| TG       | REC-R@20 | 0.3068 | 0.3807   | 0.3873 | 0.3051  | 0.3352    | 0.3808     | 0.3608  | 0.4030  |
|          | REC-N@20 | 0.1644 | 0.2141   | 0.2162 | 0.1620  | 0.1790    | 0.2215     | 0.1962  | 0.2342  |
|          | CTR-AUC  | 0.8047 | 0.8289   | 0.8326 | 0.8214  | 0.8266    | 0.8322     | 0.8328  | 0.8370  |
| THI      | REC-R@20 | 0.2555 | 0.2756   | 0.2403 | 0.2191  | 0.2577    | 0.2627     | 0.2451  | 0.3025  |
|          | REC-N@20 | 0.1529 | 0.1598   | 0.1214 | 0.1077  | 0.1505    | 0.1632     | 0.1264  | 0.1895  |
|          | CTR-AUC  | 0.7652 | 0.7850   | 0.7896 | 0.7815  | 0.7643    | 0.7787     | 0.7817  | 0.7923  |
| ML       | REC-R@20 | 0.4556 | 0.4618   | 0.4589 | 0.4549  | 0.4606    | 0.4581     | 0.4628  | 0.4640  |
|          | REC-N@20 | 0.4829 | 0.4911   | 0.4881 | 0.4822  | 0.4888    | 0.4883     | 0.4928  | 0.4928  |
|          | CTR-AUC  | 0.9109 | 0.9218   | 0.9200 | 0.9194  | 0.9145    | 0.9184     | 0.9191  | 0.9205  |

\(^1\)https://www.youtube.com/  
\(^2\)http://ffmpeg.org/  
\(^3\)https://www.themoviedb.org/

We randomly sample one item that has no interactions with the
user as negative feedback to update the model. The item recommend-
ation performances are evaluated by Recall@20 and NDCG@20
(respectively denote by REC-R@20 and REC-N@20). To improve
the item recommendation evaluation efficiency, we randomly sample
1000 items that the testing user has not interacted with to compute
REC-R@20 and REC-N@20. In the CTR prediction task, for each
testing positive interaction, we randomly sample 5 unobserved in-
teractions as negative feedbacks to construct the testing dataset.
The CTR prediction performances are evaluated by AUC (denoted by
CTR-AUC).

#### 4.1.3 Baseline Methods.

We compare MG-BERT with the follow-
ing pre-training methods: (1) DeepWalk [23]: This method learns
node representations by sampling a large number of paths in the
graph and maximizing the average logarithmic probability of all vertex context pairs in sampled paths; (2) LINE [29]: This graph
embedding method is trained to preserve the first- and second-order
proximities of nodes in the graph; (3) GraphSAGE [9]: This GNN
model forces connected nodes to have similar embeddings by ag-
gregating the information of neighboring nodes; (4) TransAE [35]: This method combines a multimodal encoder and the TransE [2]
model to learn the node representations; (5) GRAPH-BERT [37]: This method applies Transformer to aggregate neighbors’
information without masking operations on the nodes; (6) GPT-GNN [16]: This method employs the attribute generation and edge generation
tasks to pre-train the GNN model. For a fair comparison, we use
the same multimodal representation in Eq. (2) as the inputs for all
pre-training methods.

#### 4.1.4 Implementation Details.

For the pre-training and downstream
tasks, we set the dimensionality of latent space \(d_o\) to 128. In the
experiments, we empirically set the sampling depth \(K\) to 3, and
the sampling sizes \(n_1, n_2, n_3\) to 16, 8, 4 respectively. The number
of contextual neighbors \(S\) is selected from \([5, 10, 20, 30, 40]\). The
number of transformer layers \(L\) is chosen from \([1, 2, 3, 4, 5]\). The
weight of diversity-promoting attention \(\beta\) is selected from \([0, 0.2, 0.5, 0.8, 1.0]\). We implement MG-BERT based on TensorFlow [1].
Adam [18] is used as the optimizer for learning model parameters,
and the learning rate is chosen from \([10^{-4}, 10^{-3}, 10^{-2}]\).
4.2 Performance Comparison

After pre-training on the item graph, we use the pre-trained item representations to initialize the item embeddings in the downstream tasks. Then, we train the NCF and DCN models and fine-tune the item embeddings based on the user-item interaction data. Table 2 summarizes the performances of NCF and DCN initialized with item representations pre-trained by different methods. We make the following observations. Compared with the random initialization, initializing the base models with pre-trained item representations usually achieves better item recommendation and CTR prediction performances. This demonstrates the pre-training strategies can benefit downstream tasks in recommendation scenarios. The deep pre-training methods GRAPH-BERT, GPT-GNN, and MG-BERT usually outperform other shallow pre-training methods, by employing GNN to aggregate the neighbor information and using node feature reconstruction and graph structure reconstruction tasks to pre-train the model. Moreover, MG-BERT usually achieves the best item recommendation and CTR prediction performances on all datasets. This demonstrates the effectiveness of MG-BERT in exploiting the item graph structure and item features. In addition, we can also note that MG-BERT achieves smaller improvements on the ML dataset. One potential reason is that the interaction data of the ML dataset are denser. Thus, with the random initialization, the base models can learn sufficiently good item representations for downstream tasks.

4.3 Ablation Study

We study the effectiveness of MG-BERT in exploiting different modality information. Figure 2 summarizes the item recommendation and CTR prediction performances of MG-BERT, GRAPH-BERT, and GPT-GNN, considering different modality information on the TG and ML datasets. As shown in Figure 2, we can note that the original methods considering multimodality information usually outperform the variants that only consider single modality information. This observation is as expected. It indicates that representing items with multimodality information can achieve better performance. MG-BERT is superior to GRAPH-BERT and GPT-GNN with considering single modality information in most scenarios. This observation again demonstrates that MG-BERT is more effective in capturing different types of modality information than baseline methods.

Moreover, we also study the effectiveness of the two graph reconstruction tasks in learning node representations. Figure 3 summarizes the performances of MG-BERT variants on the TG and ML datasets. We can note that the original MG-BERT model with two tasks consistently outperforms the variants using a single task.
as the pre-training objective. This indicates that both the graph structure reconstruction task and masked node feature reconstruction task are essential for learning useful node representations for downstream tasks.

4.4 Parameter Sensitivity Study

In this section, we study the performances of MG-BERT with respect to (w.r.t.) different settings of three hyper parameters. Firstly, we vary the number of Transformer layers \( L \) from 1 to 5. As shown in Figure 4a, the best item recommendation and CTR prediction performances are achieved by setting \( L \) to 3 and 2, respectively. Further stacking more layers does not help improve the performances of downstream tasks. Moreover, we vary the weight of the diversity-promoting attention score \( \beta \) in \([0, 0.2, 0.5, 0.8, 1.0]\). As shown in Figure 4b, the recommendation accuracy can be improved by considering diversity-promoting attention in the Transformer-based encoder, when \( \beta \) is set to 0.5, 0.8, and 1.0. This indicates it is important to consider the diversity of contextual neighbors when pre-training the node representations. Figure 4c summarizes the performances of MG-BERT w.r.t. different settings of the number of contextual neighbors \( S \). We observe that MG-BERT usually achieves better performances by setting \( S \) to 5 and 10. This indicates that a small number of contextual neighbors can capture the important neighborhood information of a node. Further increase of \( S \) tends to include noise information, thus may not help improve the model performances.

4.5 Online Case Study

A case study is conducted in the video recommendation scenario of an E-commerce platform. The video graph is built based on users’ watching behaviors. Two videos are connected in the graph if they are watched by the same user within one hour. The edge weight is the number of users who have watched these two videos within one hour. We remove the edges with weights smaller than 10. Finally, there are about 4 million nodes and 500 millions of edges in the video graph. Given the pre-trained video representation by MG-BERT, for a user, we retrieve 50 most similar videos for each video she has watched, based on the Cosine similarity between the video representations. Then, the retrieved videos are ranked and recommended to the user. After three days of online testing, for 600 thousand users, the number of new video plays increases by 6.80%, compared with the baseline method using the video representation learned by UNITER [5]. Figure 5 shows two video retrieval examples.

We can note that the videos retrieved based on the representations pre-trained by MG-BERT are more diverse than those retrieved based on the representations pre-trained by UNITER.

5 CONCLUSION AND FUTURE WORK

This paper proposes a novel pre-training strategy, namely MG-BERT, which exploits items’ multimodal information guided by the unsupervised learning tasks on graph. Two graph reconstruction tasks, i.e., graph structure reconstruction and masked node feature reconstruction, are used as learning objectives to pre-train the model. The learned representation of an item not only integrates the multimodal information of the item itself but also aggregates the information of its contextual neighbors in the graph. The superiority of MG-BERT has been validated by two downstream tasks on four different datasets. In this work, we focus on the homogeneous graph of items. For future work, we would like to investigate how...
to extend the proposed model to process the heterogeneous item graph.

REFERENCES

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16). 265–283.

[2] Antoine Bordes, Nicolas Usunier, A. Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In {Neural Information Processing Systems}.

[3] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. {IEEE Transactions on Knowledge and Data Engineering} 30, 9 (2018), 1616–1637.

[4] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 785–794.

[5] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations. arXiv preprint arXiv:1908.11740 (2019).

[6] Ji Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In 2009 {IEEE} conference on computer vision and pattern recognition. Ieee. 248–255.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).

[8] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. 2010. Why does unsupervised pre-training help deep learning?. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. 201–208.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In Advances in neural information processing systems. 1024–1034.

[10] Kaiming He, Ross Girshick, and Pierre Dollar. 2016. Rethinking imagenet pre-training. In Proceedings of the {IEEE} international conference on computer vision. 4918–4927.

[11] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In proceedings of the 25th international conference on world wide web. 507–517.

[12] Xiaoguang He, Linxi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In Proceedings of the 26th international conference on world wide web. 173–182.

[13] Shawn Hershley, Sourish Chaudhuri, Daniel PW Ellis, Jost F Gmeckme, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. 2017. CNN architectures for large-scale audio classification. In 2017 ieee international conference on acoustics, speech and signal processing (icassp). IEEE, 131–135.

[14] Balazs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06999 (2015).

[15] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2020. Strategies for Pre-training Graph Neural Networks. In ICLR’20.

[16] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. Repeat buyer prediction for e-commerce. Conference’17, July 2017, Washington, DC, USA Susen Yang, Yong Liu, Chenyi Lei, Guoxin Wang, Hailong Tang, Juyong Zhang, and Chunyan Miao.

[17] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning with deep neural networks. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 701–710.

[18] Jiexi Zhang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. 565–573.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems. 5998–6008.

[20] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. In ICLR’19.

[21] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network with ad click predictions. In Proceedings of the ADKDD’17. 1–7.

[22] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. 165–174.

[23] Z. Wang, L. Li, Q. Li, and D. Zeng. 2019. Multimodal Data Enhanced Representation Learning for Knowledge Graphs. In 2019 International Joint Conference on Neural Networks (IJCNN). 1–8.

[24] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems. 5753–5763.

[25] Jiawei Zhang, Haopeng Zhang, Li Sun, and Congyun Xia. 2020. Graph-Bert: Only Attention is Needed for Learning Graph Representations. arXiv preprint arXiv:2001.05140 (2020).

[26] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR) 52, 1 (2019), 1–38.