Bipartite Graph Embedding via Mutual Information Maximization

Jiangxia Cao*, Xixun Lin*
Institute of Information Engineering, Chinese Academy of Sciences & School of Cyber Security, University of Chinese Academy of Sciences
{caojiangxia, linxixun}@iie.ac.cn

Shu Guo
National Computer Network Emergency Response Technical Team/Coordination Center of China
guoshu@cert.org.cn

Luchen Liu, Tingwen Liu
Institute of Information Engineering, Chinese Academy of Sciences & School of Cyber Security, University of Chinese Academy of Sciences
{liuluchen, liutingwen}@iie.ac.cn

Bin Wang
Xiaomi AI Lab, Xiaomi Inc.
wangbin11@xiaomi.com

ABSTRACT

Bipartite graph embedding has recently attracted much attention due to the fact that bipartite graphs are widely used in various application domains. Most previous methods, which adopt random walk-based or reconstruction-based objectives, are typically effective to learn local graph structures. However, the global properties of bipartite graph, including community structures of homogeneous nodes and long-range dependencies of heterogeneous nodes, are not well preserved. In this paper, we propose a bipartite graph embedding called **BiGI** to capture such global properties by introducing a novel local-global infomax objective. Specifically, BiGI first generates a global representation which is composed of two prototype representations. BiGI then encodes sampled edges as local representations via the proposed subgraph-level attention mechanism. Through maximizing the mutual information between local and global representations, BiGI enables nodes in bipartite graph to be globally relevant. Our model is evaluated on various benchmark datasets for the tasks of top-K recommendation and link prediction. Extensive experiments demonstrate that BiGI achieves consistent and significant improvements over state-of-the-art baselines. Detailed analyses verify the high effectiveness of modeling the global properties of bipartite graph.

CCS CONCEPTS

• **Information systems → Data mining**; • **Computing methodologies → Neural networks**.

KEYWORDS

Bipartite Graph Embedding; Global Properties; Mutual Information Maximization; Recommender System

ACM Reference Format:
Jiangxia Cao, Xixun Lin, Shu Guo, Luchen Liu, Tingwen Liu, Bin Wang. 2021. Bipartite Graph Embedding via Mutual Information Maximization. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441783

1 INTRODUCTION

Bipartite graph is a general structure to model the relationship between two node types. It has been widely adopted in many real-world applications, arranging from recommender system [34], drug discovery [39] to information retrieval [43]. For instance, in recommender systems, user and item represent two node types. The interactions between users and items are formed as a bipartite graph, where observed edges record previous purchasing behaviours of users. Furthermore, different from heterogeneous graphs, bipartite graph has its own structural characteristics, e.g., there are no direct links between nodes of the same type.

Learning meaningful node representations for bipartite graphs is a long-standing challenge. Recently, a significant amount of progresses have been made toward the graph embedding paradigm [2, 5, 14]. Although they work pretty well in the settings of homogeneous and heterogeneous graphs, most of them are not tailored for modeling bipartite graphs. As a result, they are sub-optimal to learn bipartite graph embedding [7, 8]. To remedy such a problem, several studies have been specifically proposed for modeling bipartite graphs. They can be roughly divided into two branches: random walk-based and reconstruction-based methods. The former [7, 8, 44] relies on designing the heuristics of random walks to generate different node sequences. Afterwards, they learn node representations via predicting context nodes within a sliding window [45]. The reconstruction-based works[15, 32, 34, 37, 40, 42] are closely related with collaborative filtering [28]. They attempt to reconstruct the adjacency matrix by learning different encoders. In particular, some works [34, 37, 40, 42] train graph neural networks (GNNs) [9, 20, 22, 38] to learn node representations via aggregating features of neighborhood nodes recursively.

*Both authors contributed equally and are listed in alphabetical order.

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441783

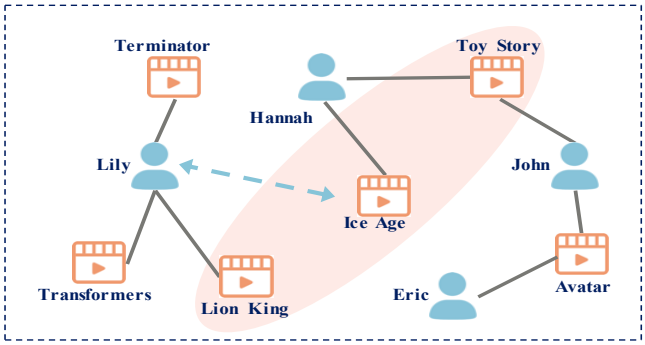

Figure 1: An example of user-movie bipartite graph. The orange shaded area represents a underlying community structure where three movies may share similar genres. The blue dotted lines denote the long-range dependency between "Lily" and "Ice Age". However, these global properties are hard to be well learned from local graph structures.

Above methods achieve promising results to some extent, but they mainly focus on learning local graph structures with the assumption that nodes within the sliding window or neighborhoods are closely relevant [7, 37, 41]. We argue that they lack the capability of better modeling the global properties of bipartite graph including community structures of homogeneous nodes and long-range dependencies of heterogeneous nodes. A concrete example is shown in Figure 1. In the user-movie bipartite graph, the movies "Lion King", "Ice Age" and "Toy Story" can be regarded as belonging to the same group since they have similar genres, but the community structure of these three homogeneous nodes is not well preserved by previous methods, due to the fact that "Lion King" is unreachable to "Ice Age" and "Toy Story". In addition, because "Lily" and "Ice Age" are distant from each other, the long-range dependency between these two heterogeneous nodes is also hard to be revealed from the local graph structures of them, even "Lily" is likely to be interested with "Ice Age".

To recognize the global properties of bipartite graph, we propose a novel Bipartite Graph embedding called **BiGI** via mutual Information maximization. Specifically, BiGI first introduces a global representation which is composed of two prototype representations, and each prototype representation is generated by aggregating the corresponding homogeneous nodes. BiGI then encodes sampled edges as local representations via the proposed subgraph-level attention mechanism. On top of that, we develop a novel local-global infomax objective to maximize the mutual information (MI) between local and global representations. In this way, our infomax objective can preserve community structures of homogeneous nodes via maximizing the MI between each node and its homogeneous prototype. Simultaneously, long-range dependencies of heterogeneous nodes are also captured by maximizing the MI between each node and its heterogeneous prototype. The main contributions of our work are as follows,

- We propose a novel bipartite graph embedding called BiGI to capture the global properties of bipartite graph including community structures of homogeneous nodes and long-range dependencies of heterogeneous nodes.
- A novel local-global infomax objective is developed via integrating the information of two node types into local and global representations. The global representation is composed of two prototype representations, and the local representation is further armed with an h-hop enclosing subgraph to preserve the rich interaction information of sampled edge.
- Our model is evaluated on multiple benchmark datasets for the tasks of top-K recommendation and link prediction. Experimental results demonstrate that our method yields consistent and significant improvements over state-of-the-art baselines[1].

2 RELATED WORK

2.1 Bipartite Graph Embedding

Homogeneous and heterogeneous graph embeddings are usually used for modeling bipartite graphs. The pioneering homogeneous graph methods include DeepWalk [27], LINE [33], Node2vec [11] and VGAE [19]. Some representative heterogeneous graph methods are Metapath2vec [6] and DMGI [41]. But they are not tailored for bipartite graphs, and the structural characteristics of bipartite graph are hard to be preserved by them. IGE [44], PinSage [40], BiNE [7] and FOBE [32] are specially designed for bipartite graphs. However, as mentioned in the *introduction*, they mainly focus on how to model local graph structures in the latent space.

Matrix completion [34, 42] and collaborative filtering [15, 37] are also connected with modeling bipartite graphs closely. They propose various DNNs to solve recommendation tasks. For example, GC-MC [34] uses one relation-aware graph convolution layer to learn node embeddings, thus only the direct links in user-item bipartite graphs are exploited. NGCF [37] incorporates collaborative signals into the embedding process by aggregating features of neighborhood nodes. However, it still overlooks the importance of modeling the global properties of bipartite graph.

2.2 Mutual Information Maximization

Maximizing the MI between inputs and corresponding latent embeddings provides a desirable paradigm for the unsupervised learning [35]. However, estimating MI is generally intractable in high-dimensional continuous settings [25]. MINE [1] derives a lower bound of MI and works by training a discriminator to distinguish samples coming from the joint distribution of two random variables or the product of their marginals. DIM [16] introduces the structural information into input patches and adopts different infomax objectives.

DGI [36] is the first work that applies the infomax objective to homogeneous graphs. It provides a new approach for the task of unsupervised node classification. Based on DIM, InfoGraph [31] tries to learn unsupervised graph representations via maximizing the MI between the graph-level representation and the representations of substructures. DMGI [41] extends DGI into heterogeneous graphs. It splits the original graph into multiple homogeneous ones and adopts the infomax objective used in DGI for modeling split graphs. So DMGI still puts more emphasis on learning the correlation of homogeneous nodes. GMI [26] proposes a new approach

[1]The source code is available from https://github.com/caojiangxia/BiGI.

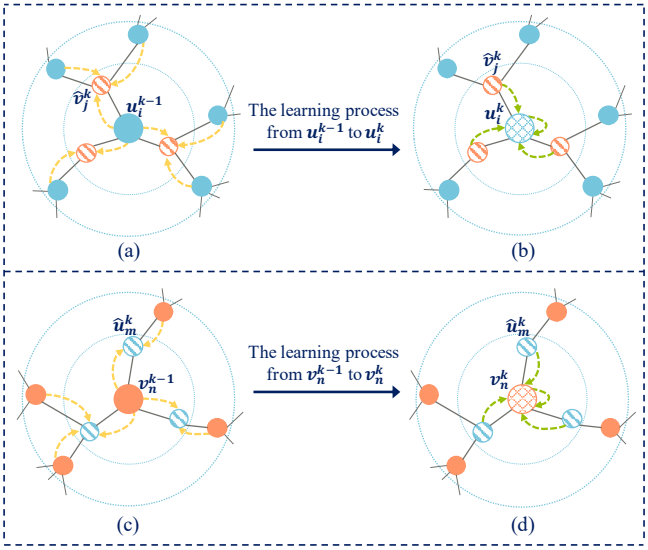

Figure 2: A simple illustration of the proposed encoder. In k-th layer, (a) and (b) show the learning process of u_i^{k-1}. (c) and (d) show the learning process of v_n^{k-1} in a similar way. The yellow dotted lines (Eq.(1) and Eq.(3)) and the green dotted lines (Eq.(2) and Eq.(4)) demonstrate how to derive node embeddings u_i^k and v_n^k.

to measure the MI between input homogeneous graphs and node embeddings directly. Compared with them, we combine two types of node information for generating local and global representations and develop a novel infomax objective that is more suitable for bipartite graphs.

3 BACKGROUND

We begin by providing the background of our work. Let $G = (U, V, E)$ be a bipartite graph, where U and V are two disjoint node sets, and $E \subseteq U \times V$ denotes the edge set. It is obvious that G has two node types. The nodes that fall into the same node set are homogeneous, and the nodes belonging to different node sets are heterogeneous. $A \in \{0,1\}^{|U|\times|V|}$ is a binary adjacency matrix, where each element $A_{i,j}$ describes whether node $u_i \in U$ has interacted with node $v_j \in V$. Given a bipartite graph $G = (U, V, E)$ with the adjacency matrix A, the goal of bipartite graph embedding is to map each node in G to a d-dimensional vector. To keep notations simple, we use $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ to represent the embedding vectors of u_i and v_j, respectively.

4 PROPOSED MODEL

A novel bipartite graph embedding termed as **BiGI** is proposed from the perspective of mutual information maximization. We first describe a basic bipartite graph encoder to generate the initial node representations. Taking these node representations as the inputs of our framework, we then demonstrate how to construct the global representation, local representations and the local-global infomax objective. The detailed model analysis is provided in the end.

4.1 Bipartite Graph Encoder

In this section, we introduce a basic bipartite graph encoder following the principle of GNN to learn the initial node representations. The proposed encoder is well compatible with our infomax objective. Compared with other GNN encoders [37, 40] for bipartite graphs, it achieves promising performances empirically.

Different from homogeneous graphs, each node in bipartite graph is not the same type as its adjacent nodes. Therefore, directly updating the node embedding via aggregating features of its one-hop neighbors is ill-posed. To alleviate such an issue, our encoder attempts to learn each node embedding from its two-hop neighbors in each layer. As shown in Figure 2, both of the learning processes of $\boldsymbol{u}_i^{k-1}$ and $\boldsymbol{v}_n^{k-1}$ have two operations in the k-th layer. Taking $\boldsymbol{u}_i^{k-1}$ for example ((a) and (b) in Figure 2), we first generate temporary neighborhood representations, e.g., $\widehat{\boldsymbol{v}}_j^k$ via a mean operation (MEAN) with a non-linear transformation:

$$\widehat{\boldsymbol{v}}_j^k = \delta\Big(\widehat{W}_v^k \cdot \mathrm{MEAN}\big(\{\boldsymbol{u}_i^{k-1} : u_i \in \mathcal{N}(v_j)\}\big)\Big), \tag{1}$$

where δ denotes the LeakyReLU activation function, $\widehat{W}_v^k$ is a weight matrix and $\mathcal{N}(v_j)$ denotes one-hop neighbors of v_j. In contrast with common graph convolutional operators [9, 13, 20], we only aggregate neighborhood features, and the own feature $\boldsymbol{v}_j^{k-1}$ is not involved in Eq.(1). Hence, $\widehat{\boldsymbol{v}}_j^k$ can be approximately regarded as a u-type node embedding. Afterwards, we use homogeneous graph convolution to obtain $\boldsymbol{u}_i^k$:

$$\begin{aligned} \overline{\boldsymbol{u}}_i^k &= \delta\Big(\overline{W}_u^k \cdot \mathrm{MEAN}\big(\{\widehat{\boldsymbol{v}}_j^k : v_j \in \mathcal{N}(u_i)\}\big)\Big), \\ \boldsymbol{u}_i^k &= W_u^k \cdot \big[\overline{\boldsymbol{u}}_i^k \| \boldsymbol{u}_i^{k-1}\big], \end{aligned} \tag{2}$$

where $\overline{W}_u^k$ and W_u^k are two weight matrices and $[\cdot\|\cdot]$ is a concatenation operation. The similar procedures are also employed to update $\boldsymbol{v}_n^{k-1}$. Sub-figure (c) illustrates the neighborhood aggregation of $\widehat{\boldsymbol{u}}_m^k$:

$$\widehat{\boldsymbol{u}}_m^k = \delta\Big(\widehat{W}_u^k \cdot \mathrm{MEAN}\big(\{\boldsymbol{v}_n^{k-1} : v_n \in \mathcal{N}(u_m)\}\big)\Big). \tag{3}$$

The final node embedding $\boldsymbol{v}_n^k$ is defined as:

$$\begin{aligned} \overline{\boldsymbol{v}}_n^k &= \delta\Big(\overline{W}_v^k \cdot \mathrm{MEAN}\big(\{\widehat{\boldsymbol{u}}_m^k : u_m \in \mathcal{N}(v_n)\}\big)\Big), \\ \boldsymbol{v}_n^k &= W_v^k \cdot \big[\overline{\boldsymbol{v}}_n^k \| \boldsymbol{v}_n^{k-1}\big]. \end{aligned} \tag{4}$$

$\widehat{W}_u^k$, $\overline{W}_v^k$ and W_v^k in Eq.(3) and Eq.(4) are also weight matrices. Dropout [30] is applied to each layer of our encoder to regularize model parameters.

4.2 Local-Global Infomax

Building upon the generated node representations, in this section, we first present the calculations of global and local representations. A novel local-global infomax objective is then developed to capture the global properties of bipartite graph.

4.2.1 Global Representation. The global representation is a holistic representation of bipartite graph, which is generated via a simple composition function (COM) that combines two prototype representations. Specifically, for each node type, we introduce a prototype representation to aggregate all homogeneous node information.

Our insight is similar to the classic few-shot learning [29] which would generate a prototype representation of each class. There are many choices to induce the prototype representation. In our work, we also adopt the mean operation which averages the information of all homogeneous nodes to obtain the corresponding prototype representation. The concrete procedures can be formulated as follows,

$$\boldsymbol{p}_u = \mathrm{MEAN}\big(\{\boldsymbol{u}_i : u_i \in U\}\big), \quad \boldsymbol{p}_v = \mathrm{MEAN}\big(\{\boldsymbol{v}_i : v_i \in V\}\big),$$
$$\boldsymbol{g} = \mathrm{COM}(\boldsymbol{p}_u, \boldsymbol{p}_v) = \big[\sigma(\boldsymbol{p}_u) \| \sigma(\boldsymbol{p}_v)\big], \tag{5}$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ denote the outputs of our encoder. $\boldsymbol{g}$ is the global representation composed of two prototype representations $\boldsymbol{p}_u$ and $\boldsymbol{p}_v$. For efficiency, we select the simple concatenation operation with the sigmoid activation function σ as our composition function.

4.2.2 Local Representation. Each input of local representation is a bipartite edge i.e., (u, v), and we further arm it with an h-hop enclosing subgraph [42] to describe the surrounding environment of (u, v). The concrete definition of h-hop enclosing subgraph is given below.

Definition 4.1. **(H-hop Enclosing Subgraph)** Given a bipartite graph $G = (U, V, E)$, two nodes $u \in U$ and $v \in V$, the h-hop enclosing subgraph for (u, v) is the subgraph $G^h_{(u,v)}$ induced from G by the union of two node sets, i.e., $G^h(u) \cup G^h(v)$. Here, $G^h(u) = \{v_i | dis(v_i, u) \le h\}$, $G^h(v) = \{u_i | dis(u_i, v) \le h\}$ and dis is a distance function. Due to the particular structure of bipartite graph, h is set as an odd number strictly.

For a specific edge $(u, v) \in E$ with the corresponding h-hop enclosing subgraph $G^h_{(u,v)}$ (The subscripts of u and v are omitted for simplicity), we use an attention mechanism (ATT) to calculate the local representation. Given node u and node $v_i \in G^h(u)$, the attention weight $\alpha_{u,i}$ can be expressed as:

$$\alpha_{u,i} = \frac{\exp\Big\{\big(W_a \cdot \boldsymbol{v}_i\big)^T \cdot \big(W'_a \cdot \boldsymbol{u}\big)\Big\}}{\sum_{v_j \in G^h(u)} \exp\Big\{\big(W_a \cdot \boldsymbol{v}_j\big)^T \cdot \big(W'_a \cdot \boldsymbol{u}\big)\Big\}}, \tag{6}$$

where T denotes the transpose operation, W_a and W'_a are two shared trainable matrices. The similar calculation procedure for node v and node $u_i \in G^h(v)$ can be defined as:

$$\alpha_{v,i} = \frac{\exp\Big\{\big(W'_a \cdot \boldsymbol{u}_i\big)^T \cdot \big(W_a \cdot \boldsymbol{v}\big)\Big\}}{\sum_{u_j \in G^h(v)} \exp\Big\{\big(W'_a \cdot \boldsymbol{u}_j\big)^T \cdot \big(W_a \cdot \boldsymbol{v}\big)\Big\}}. \tag{7}$$

The final representation of local input $\boldsymbol{g}^h_{(u,v)}$ is formulated as:

$$\boldsymbol{g}^h_{(u,v)} = \Big[\sigma\Big(\sum_{v_i \in G^h(u)} \alpha_{u,i}\boldsymbol{v}_i + \boldsymbol{u}\Big) \Big\| \sigma\Big(\sum_{u_i \in G^h(v)} \alpha_{v,i}\boldsymbol{u}_i + \boldsymbol{v}\Big)\Big]. \tag{8}$$

The local attentive representation also combines different local environments together via the same composition function used in Eq.(5). It not only highlights the central role of (u, v), but also adaptively assigns different importance factors to neighboring nodes by the subgraph-level attention mechanism.

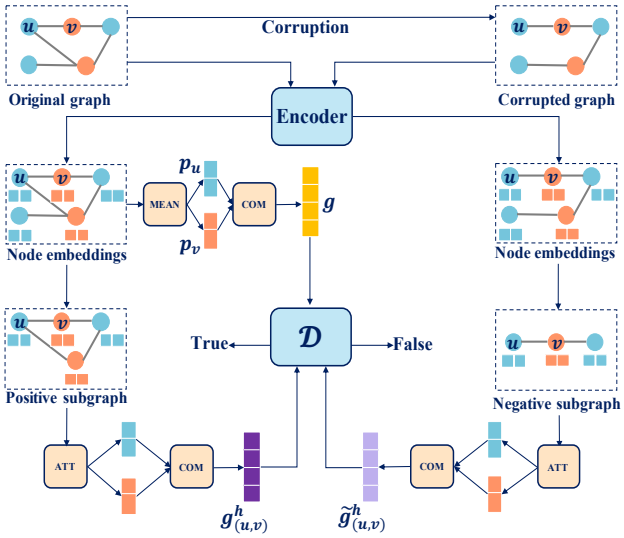

Figure 3: An overview of BiGI. "ATT", "MEAN" and "COM" denote the subgraph-level attention mechanism, mean operation and composition function, respectively. $\boldsymbol{p}_u$ and $\boldsymbol{p}_v$ are two prototype representations. $\boldsymbol{g}$ is the global representation. $\boldsymbol{g}^h_{(u,v)}$ and $\widetilde{\boldsymbol{g}}^h_{(u,v)}$ are local representations.

4.2.3 Infomax Objective. After obtaining local and global representations, our local-global infomax objective is reformulated as a noise-contrastive loss, where positive samples come from the joint distribution and negative samples come from the product of marginals. A corruption function C is required to generate the negative samples, and BiGI uses a general trick that corrupts the graph structure A to define C. The switch parameter $S_{i,j}$ determines whether to corrupt the entry of adjacency matrix $A_{i,j}$. Above operations are performed as follows,

$$S_{i,j} = \mathrm{Bernoulli}(\beta),$$
$$\widetilde{G} = (U, V, \widetilde{E}) = C(G, \beta) = A \oplus S, \tag{9}$$

where β is the corruption rate, $\oplus$ denotes the XOR (exclusive-OR) operation, $\widetilde{G}$ is the corrupted graph and $\widetilde{E}$ is the corresponding set of corrupted edges. The concrete loss function is defined as:

$$\mathcal{L}_m = -\frac{1}{|E| + |\widetilde{E}|}\Big(\sum_{i=1}^{|E|} \mathbb{E}_G\big[\log \mathcal{D}(\boldsymbol{g}^h_{(u,v)_i}, \boldsymbol{g})\big] + \sum_{i=1}^{|\widetilde{E}|} \mathbb{E}_{\widetilde{G}}\big[\log\big(1 - \mathcal{D}(\widetilde{\boldsymbol{g}}^h_{(u,v)_i}, \boldsymbol{g})\big)\big]\Big). \tag{10}$$

Here, $\mathcal{D}$ is a discriminator to score local-global representations via a bilinear mapping function:

$$\mathcal{D}(\boldsymbol{g}^h_{(u,v)_i}, \boldsymbol{g}) = \sigma\big((\boldsymbol{g}^h_{(u,v)_i})^T W_b \boldsymbol{g}\big), \tag{11}$$

where W_b is a weight matrix. The binary cross-entropy loss in Eq.(10) is an effective MI estimator. It can maximize the MI between $\boldsymbol{g}^h_{(u,v)_i}$ and $\boldsymbol{g}$, based on Jensen–Shannon divergence between the joint distribution and the product of marginals. Because it follows a standard minmax game originated from the generative adversarial

network (GAN) [10], and the "GAN" distance and Jensen-Shannon divergence are closely related [24].

From Eq.(5), we can observe that the information of two node types is integrated into the global representation via the generated prototype representations, and these two prototypes are not entangled together. Through Eq.(10) and Eq.(11), each node has access to the homogeneous prototype and to the heterogeneous prototype simultaneously, which enables our model to break the limit of local graph topology. Therefore, the global properties can be naturally captured even the correlated nodes in bipartite graph are distant from each other.

4.3 Model Training

The total loss function $\mathcal{L}$ contains two terms:

$$\mathcal{L} = \lambda \mathcal{L}_m + (1-\lambda)\mathcal{L}_r, \tag{12}$$

where λ is the harmonic factor. $\mathcal{L}_r$ is a margin-based ranking loss over observed edges for our encoder, which is formulated as follows,

$$\mathcal{L}_r = \sum_{(u,v)\in E} \sum_{(u',v')\in E'_{(u,v)}} \Big[\gamma + \phi([\boldsymbol{u}'|\boldsymbol{v}']) - \phi([\boldsymbol{u}|\boldsymbol{v}])\Big]_+, \tag{13}$$

where ϕ is a ranking function parameterized by a two-layer multi-layer perceptron (MLP), $[x]_+$ denotes the positive part of x and γ is a margin. $E'_{(u,v)}$ is the set of negative node pairs, which can be defined as:

$$E'_{(u,v)} = \{(u',v)|u' \in U\} \cup \{(u,v')|v' \in V\}. \tag{14}$$

The negative sampling used in Eq.(14) is similar to [12, 21]: $E'_{(u,v)}$ is composed of real interactions with either the head or tail replaced by a random node from the same node set. BiGI is an end-to-end model which is optimized by Adam [17]. The whole architecture of BiGI is shown in Figure 3.

4.4 Model Analysis

4.4.1 Time Complexity. The main operations of BiGI are learning the initial node representations and calculating the total loss. To avoid parameter overhead, we use the shared encoder to learn node representations of G and $\widetilde{G}$. The computational complexity of BiGI is approximated as $O(k(|E| + |\widetilde{E}|)d^2)$, where k is the number of layers in our encoder and d is the embedding size. In addition, we also provide an experimental comparison to verify that our model can be deployed to large-scale bipartite graphs in Section 5.3.

4.4.2 Relation with DGI. Our model is closely related to DGI, since they use a local-global infomax objective on graphs. However, there are important design differences between them. 1) BiGI focuses on modeling bipartite graphs, which integrates the information of two node types into local and global representations. By contrast, DGI is designed for homogeneous node embeddings. 2) DGI tries to maximize the MI between node-level and graph-level representations, while we actually maximize the MI between subgraph-level and graph-level representations. The subgraph-level representations are capable of effectively preserving rich interactions of sampled edges. 3) The choice of encoders is different. Considering the structural characteristics of bipartite graphs, we design a novel basic encoder to learn initial node representations.

Table 1: Statistics of datasets.

Datasets	$\|U\|$	$\|V\|$	$\|E\|$	Density
DBLP	6,001	1,308	29,256	0.4%
ML-100K	943	1,682	100,000	6.3%
ML-10M	69,878	10,677	10,000,054	1.3%
Wikipedia	15,000	3,214	64,095	0.1%

5 EXPERIMENTS

5.1 Datasets

Four benchmark datasets, i.e., DBLP [2], MovieLens-100K (ML-100K) [3], MovieLens-10M (ML-10M) [4] and Wikipedia [5] are used in experiments. DBLP, ML-100K and ML-10M are adopted for top-K recommendation. Wikipedia is used for link prediction. We convert their user-item interaction matrices into the implicit data. The concrete statistics of them are listed in Table 1. From it, we can observe that ML-10M is much larger than other datasets, since it is used to test whether our model can be deployed to large-scale bipartite graphs.

5.1.1 Data Preprocessing. As used in BiNE [7], we select 60% edges for training and remaining edges for test in both of DBLP and ML-10M. We use the same division in IGMC [42] for ML-100K. Following experimental settings in the previous work [8], we split Wikipedia into two datasets, i.e., Wiki (5:5) and Wiki (4:6). The training/test ratios of these two datasets are 5:5 and 4:6, respectively.

5.2 Experimental Setting

5.2.1 Evaluation Metrics. In top-K recommendation, for each user, we first filter out some items that the user has already interacted with in training process. Then, we rank remaining items and evaluate ranking results with the following evaluation metrics: $F1$ score, $NDCG$ (Normalized Discounted Cumulative Gain), MAP (Mean Average Precision) and MRR (Mean Reciprocal Rank). All of these metrics are widely used in recommendation tasks. Two common metrics are used to evaluate the results of link prediction: AUC-ROC (area under the ROC curve) and AUC-PR (area under the Precison-Recall curve).

5.2.2 Compared Baselines. We compare our model with the following strong baselines which can be divided into:

- **Homogeneous graph embedding**: DeepWalk [27], LINE [33], Node2vec [11] and VGAE [19]. DeepWalk and Node2vec are typically random-walk based. LINE learns a joint probability distribution of connected nodes, and LINE (2nd) is exploited here due to its expressive performances. Based on variational auto-encoder [18], VGAE adopts the graph convolutional network (GCN) [20] as the basic encoder to learn graph-structured data.
- **Heterogeneous graph embedding**: Metapath2vec [6] and DMGI [41]. Metapath2vec first designs the metapath-based random walks to construct heterogeneous node neighborhoods. It then leverages a heterogeneous skip-gram model to

[2] https://github.com/clhchtcjj/BiNE/tree/master/data/dblp
[3] https://grouplens.org/datasets/movielens/100k/
[4] https://grouplens.org/datasets/movielens/10m/
[5] https://github.com/clhchtcjj/BiNE/tree/master/data/wiki

Table 2: Performance (%) comparison of top-K recommendation on DBLP.

Model	F1@10	NDCG@3	NDCG@5	NDCG@10	MAP@3	MAP@5	MAP@10	MRR@3	MRR@5	MRR@10
DeepWalk	6.93	4.91	6.60	9.12	3.37	4.23	5.29	9.04	10.44	11.70
LINE	8.45	16.31	19.03	20.32	14.25	15.56	16.07	22.58	25.28	26.08
Node2vec	7.66	20.33	22.09	23.00	17.76	18.61	18.90	28.00	29.21	29.85
VGAE	10.16	15.71	16.57	18.75	11.08	11.38	12.17	16.93	18.30	19.64
Metapath2vec	8.16	19.81	21.89	22.70	17.24	18.15	18.46	27.23	29.25	29.68
DMGI	9.16	19.71	22.01	23.65	17.09	18.27	18.87	26.69	28.27	30.13
PinSage	12.55	18.62	21.17	23.97	14.71	16.04	17.30	27.75	29.68	30.84
BiNE	11.36	19.85	21.95	25.15	17.12	18.05	19.34	27.14	29.40	31.33
GC-MC	12.02	19.87	22.18	24.62	16.75	17.98	19.12	28.91	30.65	31.70
IGMC	12.18	20.35	22.65	25.17	17.21	18.43	19.61	29.56	31.30	32.36
NeuMF	11.14	19.59	21.08	24.31	16.46	17.19	18.44	27.01	28.23	30.32
NGCF	12.38	21.29	23.38	25.58	17.36	18.50	19.51	30.96	32.48	33.44
BiGI	**14.27***	**23.56***	**25.39***	**28.28***	**19.10***	**20.15***	**21.49***	**33.19***	**35.35***	**36.51***

* indicates that the improvements are statistically significant for p < 0.05 judged by paired t-test.

Table 3: Performance (%) comparison of top-K recommendation on ML-100K.

Model	F1@10	NDCG@3	NDCG@5	NDCG@10	MAP@3	MAP@5	MAP@10	MRR@3	MRR@5	MRR@10
DeepWalk	14.20	7.17	9.32	13.13	2.72	3.54	4.92	43.86	46.83	48.75
LINE	13.71	6.52	8.57	12.37	2.45	3.26	4.67	44.16	44.37	46.30
Node2vec	14.13	7.69	9.91	13.41	3.07	3.90	5.19	44.80	48.02	49.78
VGAE	11.38	6.43	8.18	10.93	2.35	2.95	3.94	39.39	42.32	43.68
Metapath2vec	14.11	7.88	9.87	13.35	2.85	3.71	5.08	45.49	48.74	49.83
DMGI	19.58	10.16	13.13	18.31	3.98	5.33	7.82	59.33	61.37	62.71
PinSage	21.68	10.95	14.51	20.27	4.52	6.18	9.13	62.56	64.77	65.76
BiNE	14.83	7.69	9.96	13.79	2.87	3.80	5.24	48.14	50.94	52.51
GC-MC	20.65	10.88	13.87	19.21	4.41	5.84	8.43	60.60	62.21	63.53
IGMC	18.81	9.21	12.20	17.27	3.50	4.82	7.18	56.89	59.13	60.46
NeuMF	17.03	8.87	11.38	15.89	3.46	4.54	6.45	54.42	56.39	57.79
NGCF	21.64	11.03	14.49	20.29	4.49	6.15	9.11	62.56	64.62	65.55
BiGI	**23.36***	**12.50***	**15.92***	**22.14***	**5.41***	**7.15***	**10.50***	**66.01***	**67.70***	**68.78***

* indicates that the improvements are statistically significant for p < 0.05 judged by paired t-test.

Table 4: Performance (%) comparison of top-K recommendation on ML-10M.

Model	F1@10	NDCG@3	NDCG@5	NDCG@10	MAP@3	MAP@5	MAP@10	MRR@3	MRR@5	MRR@10
DeepWalk	7.25	3.12	4.39	6.50	1.12	1.65	2.55	19.14	20.97	22.45
LINE	6.93	3.07	4.21	6.24	1.09	1.55	2.37	19.69	21.54	23.08
Node2vec	6.36	2.82	3.84	5.71	1.00	1.40	2.14	18.10	19.83	21.32
VGAE	11.82	5.00	6.97	10.61	1.88	2.79	4.65	34.75	37.13	39.00
Metapath2vec	8.28	3.26	4.66	7.21	1.18	1.79	2.98	19.99	21.92	23.50
DMGI	12.52	6.03	8.09	11.69	2.15	3.04	4.77	42.78	44.86	46.08
PinSage	14.93	7.53	10.07	14.14	2.70	3.81	5.85	45.72	47.58	48.96
GC-MC	14.74	7.05	9.42	13.73	2.58	3.68	5.88	48.07	49.95	51.18
IGMC	13.68	6.58	8.70	12.78	2.41	3.32	5.22	45.57	47.82	49.29
NeuMF	13.91	6.58	8.92	12.93	2.38	3.41	5.34	45.82	48.14	49.57
NGCF	15.11	7.21	9.67	14.01	2.67	3.84	6.16	48.19	50.15	51.33
BiGI	**16.12***	**7.96***	**10.41***	**15.25***	**3.02***	**4.31***	**6.77***	**49.86***	**50.66***	**51.70***

* indicates that the improvements are statistically significant for p < 0.05 judged by paired t-test.

Table 5: Performance comparison (%) of link prediction.

Model	Wiki (5:5)		Wiki (4:6)	
	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR
DeepWalk	87.19	85.30	81.60	80.29
LINE	66.69	71.49	64.28	69.89
Node2vec	89.37	88.12	88.41	87.55
VGAE	87.81	86.93	86.32	85.74
Metapath2vec	87.20	84.94	86.75	84.63
DMGI	93.02	93.11	92.01	92.14
PinSage	94.27	93.95	92.79	92.56
BiNE	94.33	93.93	93.15	93.34
GC-MC	91.90	92.19	91.40	91.74
IGMC	92.85	93.10	91.90	92.19
NeuMF	92.62	93.38	91.47	92.63
NGCF	94.26	94.07	93.06	93.37
BiGI	**94.91***	**94.75***	**94.08***	**94.02***

* indicates that the improvements are statistically significant for p < 0.05 judged by paired t-test.

learn node embeddings. DMGI [41] also follows the principle of MI maximization, and it uses the same infomax objective in DGI [36].

- **Bipartite graph embedding**: PinSage [40] and BiNE [7]. PinSage integrates random walk into GNN architectures for high-scalable performances. BiNE jointly optimizes explicit and implicit relations in a unified framework.
- **Matrix completion**: GC-MC [34] and IGMC [42]. GC-MC introduces a relation-aware graph auto-encoder to learn embeddings of users and items. These representations are then used to reconstruct the rating links through a bilinear decoder. IGMC proposes a novel GNN based on local subgraphs for the task of inductive matrix completion.
- **Collaborative filtering**: NeuMF [15] and NGCF [37]. NeuMF uses MLP to learn the nonlinear interactions between user and item embeddings. NGCF considers the high-order connectivity via the proposed embedding propagation layer.

5.2.3 Implementation Details. PinSage is implemented by ourselves. Except from it, we use official implementations of other methods. To make a fair comparison, the side information of nodes is not exploited in all experiments. The embedding size d is fixed as 128, the learning rate is 0.001 and all models are iterated with 100 epochs for convergence. For making a good trade-off between effectiveness and efficiency, we use 1-hop enclosing subgraphs as suggested by IGMC [42]. The depth of our encoder k (the number of stacked layers) is 2. The margin γ used in Eq.(13) is 0.3. the corruption rate β is selected from {1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1}, and the harmonic factor λ is selected from 0.1 to 0.9 with step length 0.2.

To verify whether the results of our model are statistically significant, we perform paired t-test for each dataset. In addition, the results of BiNE on ML-10M are not provided. Although we use the official implementation of BiNE [6] for large-scale bipartite graphs, the concrete results of it are hard to be well reproduced (The generation of node sequences has not finished within 72 hours).

[6] https://github.com/clhchtcjj/BiNE

Table 6: Performance (%) comparison of model variants.

Model	F1@10	NDCG@10	MAP@10	MRR@10
Encoder	12.29	25.71	19.31	31.08
BiGI (node)	12.64	25.46	19.34	34.08
BiGI (pair)	13.23	27.57	21.20	34.80
BiGI (w/o att)	13.26	27.59	21.26	35.33
BiGI	14.27	28.28	21.49	36.51
VGAE	10.16	18.75	12.17	19.64
BiGI (VGAE)	11.15	24.36	19.05	30.68
NGCF	12.38	25.58	19.51	33.44
BiGI (NGCF)	13.03	26.25	20.34	35.23
PinSage	12.55	23.97	17.30	30.84
BiGI (PinSage)	13.26	27.61	21.48	35.11

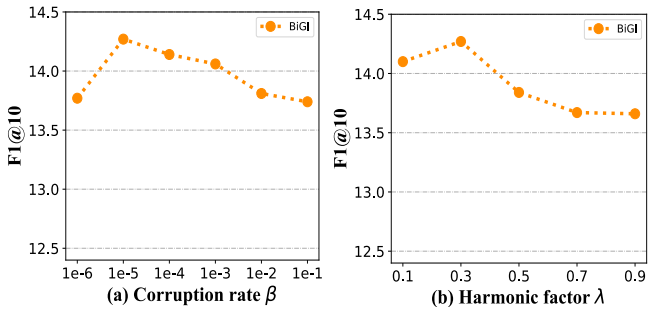

Figure 4: Results of parameter sensitivity.

5.3 Top-K Recommendation

Table 2, Table 3 and Table 4 demonstrate the performances of compared methods on DBLP, ML-100K and ML-10M. The best performance is in boldface and the second is underlined. From them, we have the following observations. 1) Our method consistently yields the best performances on these datasets for all metrics. It demonstrates the high effectiveness of learning the global properties of bipartite graph. 2) Modeling the structural characteristics of bipartite graph is very important. Homogeneous and heterogeneous graph embeddings ignore such characteristics, and they are inferior to BiGI and to other bipartite graph embeddings. 3) It should be noticed that DMGI also maximizes MI between local and global representations, but the performance of it is not satisfying. Therefore, designing a suitable infomax objective for bipartite graphs plays a central role in our work.

5.4 Link Prediction

For the task of link prediction, given a node pair (u_i, v_j), we feed the corresponding embeddings $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ into a logistic regression classifier which is trained on the observed edges of bipartite graph. Table 5 shows the performances of all models, and our method achieves higher predictive results on both datasets. It demonstrates

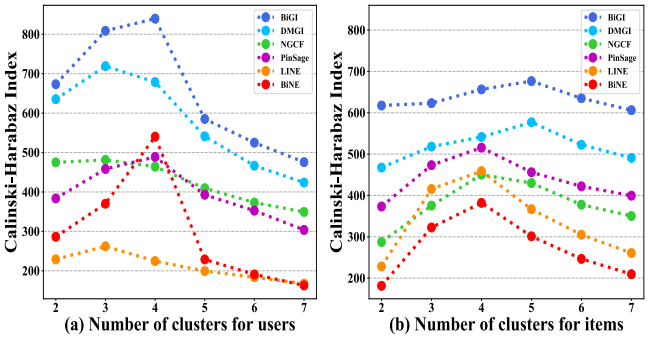

Figure 5: Results of clustering analysis. BiGI achieves the best clustering results (A higher score is preferred).

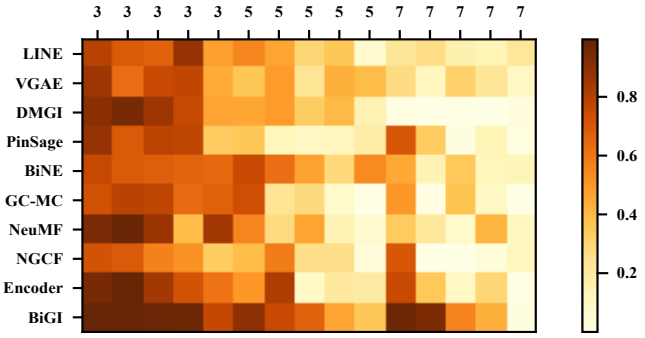

Figure 6: Visualization of prediction scores. BiGI achieves the better results compared with other baselines.

that the global properties of bipartite graph are beneficial to learn node representations. In particular, capturing long-range dependencies of heterogeneous nodes is helpful to down-stream tasks.

5.5 Discussions of Model Variants

We investigate the effects of different local representations and the extensibility of proposed infomax objective. The results of these model variants and an ablation study of the proposed encoder are provided in Table 6. The experiments are conducted on DBLP. BiGI (node) uses each node embedding as the local representation. BiGI (pair) simply concatenates the representations of node pair (u, v) as the local representation. BiGI (w/o att) calculates the representation of subgraph via the mean operation instead of the attention mechanism. BiGI (VGAE), BiGI (NGCF) and BiGI (PinSage) adopt VGAE, NGCF and PinSage as their encoders, respectively. All of them keep the same infomax objective with BiGI.

From the results in Table 6, we can draw the following conclusions. 1) The proposed encoder achieves competitive performance. Furthermore, in contrast with it, the improvements of BiGI are also significant. 2) Through the comparison of different local representations, we find that constructing a suitable local representation is crucial to BiGI. Introducing the subgraph-level attention mechanism into the calculation of local representation is a sensible choice. 3) By contrast with VGAE, NGCF and PinSage, the improvements of BiGI (VGAE), BiGI (NGCF) and BiGI (PinSage) are satisfying. It indicates that the proposed infomax objective can be seamlessly incorporated into other encoders to capture the global properties of bipartite graph.

5.6 Parameter Sensitivity

We investigate the parameter sensitivity of our model on DBLP with respect to two hyper-parameters: the corruption rate β in Eq.(9) and the harmonic factor λ in Eq.(12). As shown in Figure 4, when β=1e-5 and $\lambda = 0.3$, our model achieves the best result. Therefore, choosing relative small values of β and λ is a reasonable way. Moreover, our model is robust to the changes of β and λ. Even in the worst settings of β and λ, BiGI is still better than other baselines shown in Table 2.

5.7 Analysis of the Global Properties

In this section, to validate that our method is better to capture the global properties of bipartite graph, we conduct two detailed comparisons between BiGI and other strong baselines. In the first experiment, we provide two clustering analyses of users and items which are conducted on ML-100K. We first save all representations of users and items and then cluster them via the well-known K-Means algorithm. The clustering metric Calinski-Harabasz Index (CHI) [3] is used here. CHI measures the ratio between the within-cluster dispersion and the between-cluster dispersion. It is also commonly used to evaluate the task of community detection [4, 23]. As shown in Figure 5, compared with other graph embeddings, BiGI achieves the best clustering results with the varying number of clusters. It demonstrates that BiGI can better capture community structures of users and items simultaneously.

Another comparison is used to test whether the long-range dependencies of heterogeneous nodes can be learned by our model. The scores are predicted by BiGI and several baselines for fifteen node pairs $\{(u_i, v_j)\}$ which are randomly picked from the test data of DBLP. These node pairs can be actually divided into three groups in terms of the distance between u_i and v_j, i.e., 3, 5 and 7. From Figure 6, we have the following conclusions. 1) When the distance of target node pair is relative short, e.g. 3, all baselines and BiGI are capable of learning the latent interaction of node pair. 2) With the increase of distance, the observable relation between u_i and v_j is gradually weakened. Compared with state-of-the-art baselines, BiGI still maintains promising results. It demonstrates that BiGI can learn the long-range dependency of u_i and v_j even though they are distant from each other.

6 CONCLUSION

In this paper, we propose a novel bipartite graph embedding named as BiGI. We first introduce a novel bipartite graph encoder to learn initial node representations. Two prototype representations are then generated via aggregating different homogeneous node information, which are further used to construct the global representation. Furthermore, we incorporate the structure prior into local representations via the designed subgraph-level attention mechanism. Through maximizing the MI between local and global representations, BiGI can recognize the global properties of bipartite graph effectively. Extensive experiments demonstrate that BiGI consistently outperforms state-of-the-art baselines on various datasets for different tasks.

ACKNOWLEDGEMENT

This research was supported by the National Key Research and Development Program of China (grant No.2016YFB0801003), the Strategic Priority Research Program of Chinese Academy of Sciences (grant No.XDC02040400) and the National Social Science Foundation of China (grant No.19BSH022). Shu Guo and Tingwen Liu are corresponding authors.

REFERENCES

[1] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, R. Devon Hjelm, and Aaron C. Courville. 2018. Mutual Information Neural Estimation. In *International Conference on Machine Learning (ICML)*.

[2] 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 (TKDE)* (2018).

[3] Tadeusz Caliński and Jerzy Harabasz. 1974. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods* (1974).

[4] Janamejaya Chowdhary, Frank E Löffler, and Jeremy C Smith. 2017. Community detection in sequence similarity networks based on attribute clustering. *PloS One* (2017).

[5] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2017. A Survey on Network Embedding. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2017).

[6] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *ACM Knowledge Discovery and Data Mining (KDD)*.

[7] Ming Gao, Leihui Chen, Xiangnan He, and Aoying Zhou. 2018. BiNE: Bipartite Network Embedding. In *International Conference on Research on Development in Information Retrieval (SIGIR)*.

[8] Ming Gao, Xiangnan He, Leihui Chen, and Aoying Zhou. 2019. Learning Vertex Representations for Bipartite Networks. *ArXiv* (2019).

[9] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning (ICML)*.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*.

[11] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *ACM Knowledge Discovery and Data Mining (KDD)*.

[12] Shu Guo, Lin Li, Zhen Hui, Lingshuai Meng, Bingnan Ma, Wei Liu, Lihong Wang, Haibin Zhai, and Hong Zhang. 2020. Knowledge Graph Embedding Preserving Soft Logical Regularity. In *ACM International Conference on Information and Knowledge Management (CIKM)*.

[13] William Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*.

[14] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. *IEEE Data Engineering Bulletin* (2017).

[15] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *International World Wide Web Conferences (WWW)*.

[16] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations (ICLR)*.

[17] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*.

[18] Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*.

[19] Thomas N Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *NeurIPS Workshop on Bayesian Deep Learning* (2016).

[20] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

[21] Xixun Lin, Hong Yang, Jia Wu, Chuan Zhou, and Bin Wang. 2019. Guiding Entity Alignment via Adversarial Knowledge Embedding. In *IEEE International Conference On Data Mining (ICDM)*.

[22] Xixun Lin, Chuan Zhou, Hong Yang, Jia Wu, Haibo Wang, Yanan Cao, and Bin Wang. 2020. Exploratory Adversarial Attacks on Graph Neural Networks. In *IEEE International Conference On Data Mining (ICDM)*.

[23] Xin Liu, Hui-Min Cheng, and Zhong-Yuan Zhang. 2019. Evaluation of community detection methods. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* (2019).

[24] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. 2016. f-gan: Training generative neural samplers using variational divergence minimization. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*.

[25] Liam Paninski. 2002. Estimation of Entropy and Mutual Information. *Neural Computation* (2002).

[26] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph Representation Learning via Graphical Mutual Information Maximization. In *International World Wide Web Conferences (WWW)*.

[27] Bryan Perozzi, Rami Al-Rfou', and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *ACM Knowledge Discovery and Data Mining (KDD)*.

[28] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *International World Wide Web Conferences (WWW)*.

[29] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)* (2014).

[31] Fan-Yun Sun, Jordan Hoffmann, and Jian Tang. 2020. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *International Conference on Learning Representations (ICLR)*.

[32] Justin Sybrandt and Ilya Safro. 2019. FOBE and HOBE: First- and High-Order Bipartite Embeddings. *ArXiv* (2019).

[33] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *International World Wide Web Conferences (WWW)*.

[34] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *ArXiv* (2017).

[35] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *ArXiv* (2018).

[36] Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. 2019. Deep Graph Infomax. In *International Conference on Learning Representations (ICLR)*.

[37] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *International Conference on Research on Development in Information Retrieval (SIGIR)*.

[38] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)* (2020).

[39] Yoshihiro Yamanishi, Masaaki Kotera, Minoru Kanehisa, and Susumu Goto. 2010. Drug-target interaction prediction from chemical, genomic and pharmacological data in an integrated framework. In *Bioinformatics*.

[40] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *ACM Knowledge Discovery and Data Mining (KDD)*.

[41] Chan young Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. 2020. Unsupervised Attributed Multiplex Network Embedding. In *AAAI Conference on Artificial Intelligence (AAAI)*.

[42] Muhan Zhang and Yixin Chen. 2020. Inductive Matrix Completion Based on Graph Neural Networks. In *International Conference on Learning Representations (ICLR)*.

[43] Yuan Zhang, Dong Wang, and Yan Zhang. 2019. Neural IR Meets Graph Embedding: A Ranking Model for Product Search. In *International World Wide Web Conferences (WWW)*.

[44] Yao Zhang, Yun Xiong, Xiangnan Kong, and Yangyong Zhu. 2017. Learning node embeddings in interaction graphs. In *ACM International Conference on Information and Knowledge Management (CIKM)*.

[45] Ziwei Zhang, Peng Cui, Xiao Wang, Jian Pei, Xuanrong Yao, and Wenwu Zhu. 2018. Arbitrary-order proximity preserved network embedding. In *ACM Knowledge Discovery and Data Mining (KDD)*.