Heterogeneous Information Network Embedding with Convolutional Graph Attention Networks

Meng Cao, Xiying Ma, Kai Zhu, Ming Xu, and Chongjun Wang
National Key Laboratory for Novel Software Technology
Department of Computer Science and Technology
Nanjing University
Nanjing 210023, China
{caomeng,mf1833043,zhukai}@smail.nju.edu.cn, xuming0830@gmail.com, chjwang@nju.edu.cn

***Abstract*—Heterogeneous Information Networks (HINs) are prevalent in our daily life, such as social networks and bibliography networks, which contain multiple types of nodes and links. Heterogeneous information network embedding is an effective HIN analysis method, it aims at projecting network elements into a lower-dimensional vector space for further machine learning related evaluations, such as node classification, node clustering, and so on. However, existing HIN embedding methods mainly focus on extracting the semantic-related information or close neighboring relations, while the high-level proximity of the network is also important but not preserved. To address the problem, in this paper we propose CGAT, a semi-supervised heterogeneous information network embedding method. We optimize the graph attention network by adding additional convolution layers, thereby we can extract multiple types of semantics and preserve high-level information in HIN embedding at the same time. Also, we utilize label information in HINs for semi-supervised training to better obtain the model parameters and HIN embeddings. Experimental results on real-world datasets demonstrate the effectiveness and efficiency of the proposed model.**

***Index Terms*—Social Networks, Network Embedding, Graph Neural Networks, Graph Analysis**

I. Introduction

In our daily life, most information networks contain multiple types of nodes and links, these networks are defined as Heterogeneous Information Networks (HINs). For example, the bibliography network can be modeled as a HIN containing four types of nodes: paper, author, venue, term, and three types of relations including *write*(author-paper), *publish*(paper-venue), and *contain*(paper-term). Compared to homogeneous networks with single-typed nodes and links, HINs contain richer semantic information and can better describe real-world systems. Therefore, it is of great significance to effectively extract and represent the information in HINs.

Network embedding, also known as network representation learning, is an emerging network analysis method which aims at projecting network elements into a lower-dimensional vector space while preserving the properties of the original network. The embedded vectors can be further utilized for machine-learning-related tasks, such as node classification [1], [2], community detection [3], [4], link prediction [5], [6], and so on. Existing network embedding methods including DeepWalk [7], LINE [8], and node2vec [9], are mostly designed for homogeneous networks, which will lose information if directly applied on HINs. The problem of HIN embedding is challenging due to the following two reasons: 1) the embedding of network elements should preserve not only the topological information but also the heterogeneity of nodes and links; 2) the various latent semantics and high-level proximity in HINs should also be considered.

Recently some HIN embedding methods are proposed, including metapath2vec [10] and HIN2VEC [11] which utilize shallow neural networks, as well as deep learning-based methods such as BL-MNE [12]. These methods either only focus on preserving certain semantics in HINs or concentrate on local network structures within limited hops of neighbors. In other words, the goal of the above-mentioned methods is to embed nodes with similar semantics or close neighboring relations (i.e. first-order and second-order neighbors) into nearby vectors. However, nodes in distant neighborhoods can also have similar embeddings, which indicates that high-level proximity should also be considered in the HIN embedding learning process. Besides, these methods are unsupervised, while nodes labels are usually available as support information in HINs, hence it is beneficial to include label information in the learning process. Therefore, efficient semi-supervised methods for analyzing and extracting the latent knowledge in HINs are highly desired.

Graph Neural Networks (GNNs) is a powerful tool for representation learning on graphs, which employ deep neural networks to aggregate feature information of neighboring nodes, and are shown to be effective in various network analysis tasks, such as network embedding [13], [13], graph classification [14], [15], and so on. Current models include GCN [16] and GAT [17], which utilized node attributes and can well preserve local structures. However, there are two limitations of existing GNN models. First, most GNN models are designed for homogeneous graphs which include only one type of node and edge, thus they cannot be directly applied to HINs. Second, although some GNN models for HINs are proposed lately, such as HAN [18] and HetGNN [19]. HAN converts the HIN into meta-path-based homogeneous networks with same-typed nodes, thus has limitations when embeddings of different types of nodes in HINs are required. HetGNN can learn embeddings of different types of nodes, however, it adopted a sampling-based and fixed-length neighborhood se-

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

lection mechanism, which may cause information loss among neighbors.

In this paper, we propose a novel heterogeneous information network embedding model named CGAT. To effectively preserve the structural topology and semantic properties of HINs, we adopt multiple meta-paths based information sampling and pre-training process. Besides, to effectively assemble multiple semantic information from different meta-paths, we employ a 1d-convolution layer. After that, an optimized graph attention network is proposed which can preserve the high-level proximity of HINs with the feature aggregation mechanism of graph attention networks.

In conclusion, the main contributions of this paper can be summarized as follows:

- We propose an effective HIN embedding method which utilizes multiple meta-paths based sampling processes to preserve different semantics.
- We propose an optimized graph attention network model that adopts a 1d-convolution layer for multi-semantic assembling and employs the feature aggregation mechanism for high-level proximity learning.
- We conduct experiments on two real-world datasets and test the model with three network mining tasks to evaluate model performance.

The remainder of this paper is organized as follows. We first provide the preliminary concepts in Sec. II. In Sec. III, we introduce the proposed model in detail. Experimental results and analysis are presented in Sec. IV. Then we review the related works in Sec. V. Finally, we conclude the paper and vision the future work in Sec. VI.

II. Problem Definition

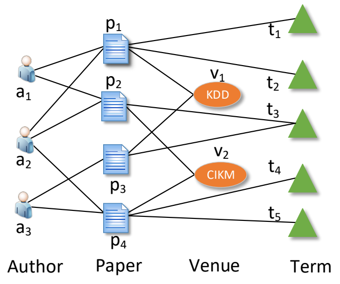
A heterogeneous information network is defined as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a node mapping function $\phi : \mathcal{V} \to \mathcal{A}$, and a link mapping function $\psi : \mathcal{E} \to \mathcal{R}$, where $\mathcal{V}$ and $\mathcal{E}$ denote nodes and links in $\mathcal{G}$, $\mathcal{A}$ and $\mathcal{R}$ denote the sets of node types and link types, and $|\mathcal{A}| + |\mathcal{R}| > 2$.

In a HIN, the semantic information shows the composite relationships between given nodes, which is usually revealed by meta-paths [20].A meta-path ρ is defined as a specific sequence of node types $a_1, a_2, \ldots, a_n$ and/or edge types $r_1, r_2, \ldots, r_{n-1}$:

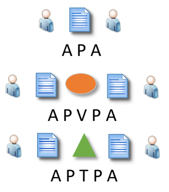
$$\rho = a_1 \overset{r_1}{\to} \ldots a_i \overset{r_i}{\to} \ldots \overset{r_{n-1}}{\to} a_n$$

In Fig. 1(a) we present an example of a bibliography network which consists of four node types as Author(A), Paper(P), Venue(V), and Term(T), and three link types as *write* (A-P), *publish* (P-V), and *contain* (P-T). Take the meta-path Author-Paper-Author(APA) for example, it denotes two authors collaborating on the same paper, and $a_1 - p_1 - a_2$ is a *meta-path instance* of the meta-path APA. Fig. 1(b) shows a list of possible meta-paths.

The goal of HIN embedding in this paper is to learn a mapping function $f : \mathcal{V} \to \mathbb{R}^d$ that projects each node $v \in \mathcal{V}$



(a) An example of HIN



(b) Meta-paths examples

Fig. 1. An illustrative example of a bibliography network and meta-paths.

to a vector in a d-dimensional space $\mathbb{R}^d$, where $d \ll |\mathcal{V}|$, meanwhile preserving the structural property as well as semantics and high-level proximity in the original HIN. The learned representation can benefit downstream network data mining tasks, such as node classification, network visualization, etc. To solve the above-mentioned problem, a task-specific learning framework is proposed, which we will introduce in detail in the next section.

III. The Proposed Method.

To address the problem of HIN embedding, we propose a model named Convolutional Graph Attention Networks (CGAT). The overall framework of the proposed model is shown in Fig. 2. CGAT includes three components: (i) meta-path-based sampling and pre-training, (ii) multi-semantic assembling with 1d-convolution, and (iii) semi-supervised learning with graph attention network. We will dilate upon the details of each component as follows.

A. Meta-path-based Sampling and Pre-training

To extract the semantic-related information in HINs, we employ multiple meta-paths based random walks to capture rich semantic information and structural correlations between different types of nodes.

Specifically, given a HIN $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a meta-path scheme set $\mathcal{P}$, we can obtain a set of meta-path constrained random walk sequences $\{v_i^t\}_{\mathcal{P}_l}$, where $v_i \in \mathcal{V}$, $t \in \mathcal{A}$ denotes node type, and $l \in [0, |\mathcal{P}|)$. In practice, the meta-paths are usually set to be symmetric so that the walk can be recursively guided for sufficient sampling in long distances.

After the meta-path based walk sequences are obtained, for each type of meta-path scheme $\mathcal{P}_l$, a heterogeneous Skip-Gram model is adopted to learn the node embeddings by maximizing the overall conditional probabilities of the context appearance with given central nodes, that is:

$$\arg\max_{\theta} \prod_{v \in \mathcal{V}} \prod_{c \in \mathcal{N}(v)} p\left(c|v;\theta\right) \tag{1}$$

where $\mathcal{N}(v)$ is the neighborhood of node v, $p\left(c|v;\theta\right)$ represents the conditional probability, and θ is the model parameter.

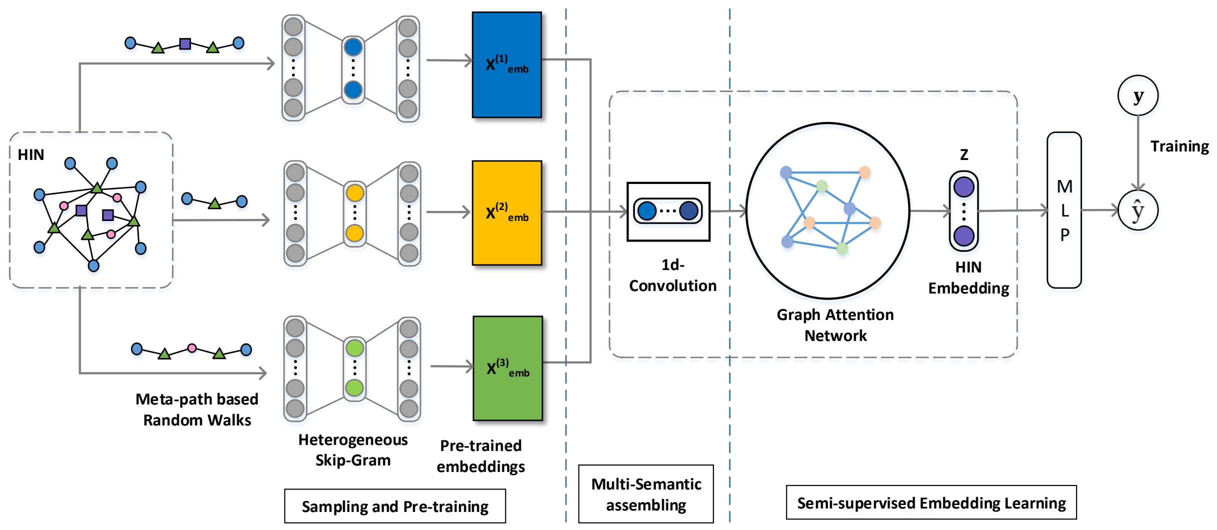


Fig. 2. The overall framework of the proposed CGAT model.

Considering different node types, we can reformulate the objective function in (1) as:

$$\arg\max_{\theta} \sum_{v\in\mathcal{V}} \sum_{t\in\mathcal{A}} \sum_{c_t\in\mathcal{N}_t(v)} \log p\left(c_t|v;\theta\right) \tag{2}$$

where $\mathcal{N}_t\left(v\right)$ represents the neighborhood of node v with type t. According to literature [21], $p\left(c_t|v;\theta\right)$ is usually defined as a softmax function:

$$p\left(c_t|v;\theta\right) = \frac{e^{X_{c_t}\cdot X_v}}{\sum_{u_t\in\mathcal{V}_t} e^{X_{u_t}\cdot X_v}} \tag{3}$$

where X_v is the embedding vector of node v, $\mathcal{V}_t$ is the node set of type t .

In order to calculate (3) efficiently, negative sampling [21] is introduced. Given negative sample size M, the objective function in (2) is reformulated as:

$$\mathcal{O}\left(X\right) = \log\sigma\left(X_{c_t}\cdot X_v\right) + \sum_{m=1}^{M} \mathbb{E}_{u_t^m\sim P_t(u_t)}\left[\log\sigma\left(-X_{u_t^m}\cdot X_v\right)\right] \tag{4}$$

where $\sigma\left(x\right) = \frac{1}{1+e^{-x}}$, and $P_t(u_t)$ is the negative sample distribution, from which a negative node u_t^m of type t is sampled for M times.The network embedding $X^{(l)}$ of meta-path $\mathcal{P}_l$ is then optimized by using stochastic gradient descent algorithm.

B. Multi-Semantic Assembling with 1d-Convolution

For meta-path set $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_l, ..., \mathcal{P}_L\}$, we repeat the heterogeneous embedding training process to obtain the pre-trained network embeddings with different meta-path based semantics, i.e. $\left\{X^{(1)}, X^{(2)}, ..., X^{(l)}, ..., X^{(L)}\right\}$, where $X^{(l)} \in \mathbb{R}^{N\times d}$, N is the number of nodes, and d is the embedding dimension. We then concatenate the embeddings as follows:

$$\mathbf{X} = concat(X^{(1)}, X^{(2)}, ..., X^{(l)}, ..., X^{(L)}) \in \mathbb{R}^{N\times d\times L} \tag{5}$$

In order to assemble the semantic information of each meta-path, we designed a 1d-convolution kernel $\tau \in \mathbb{R}^L$, each dimension in τ can be seen as the weight of the corresponding semantic embedding in $\mathbf{X}$.

For each node $v_i \in \mathcal{V}$, the assembled embedding $\mathbf{h}_i$ is defined as:

$$\mathbf{h}_i = \tau * \mathbf{x}_i \in \mathbb{R}^d \tag{6}$$

where the $*$ operator denotes convolution operation, $\mathbf{x}_i \in \mathbf{X}$ is the concatenated embedding of node v_i, and the k-th dimension in $\mathbf{h}_i$ can be calculated as follows:

$$\mathbf{h}_i(k) = \sum_{j=1}^{L} \tau(j)\mathbf{x}_i(k,j) \tag{7}$$

The convolution process is illustrated in Fig. 3, where three meta-path-related embeddings are concatenated and assembled via the convolution kernel τ. In practice, the values in τ can be seen as the shared model parameters in a CNN layer, hence they can be learned by jointly training a CNN layer and succeeding graph attention network, which will be elaborated in the next subsection.

C. Semi-supervised Learning with Graph Attention Network

After obtaining the multi-semantic assembled embedding $\mathbf{h}_i$ for node v_i, we employ graph attention network to perform feature transmission and aggregation among neighboring nodes.

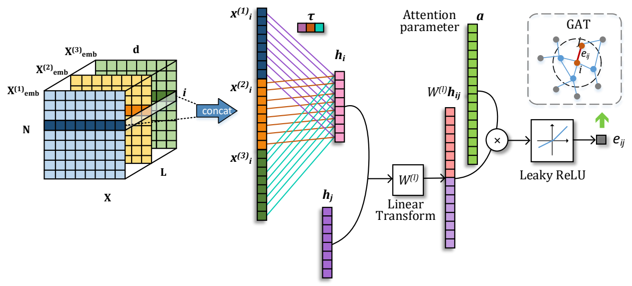


Fig. 3. The illustration of multi-semantic assembling of node v_i via 1d-convolution and succeeding graph attention layer. Same colors in the convolution layer indicate shared weights.

The graph attention network consists of two graph attention layers, and each graph attention layer contains an input layer, a hidden layer, and an output layer.

Firstly, the input layer performs a linear transformation on the assembled embedding $\mathbf{h}_i$, i.e.:

$$\mathbf{z}_i = W^{(l)}\mathbf{h}_i \tag{8}$$

where $W^{(l)}$ is the weight matrix of the l-th input layer.

After the linear transformation, a pair-wise attention score between v_i and its neighbor v_j is calculated:

$$e_{ij} = \text{LeakyReLU}\left(\vec{a}^T\left(\mathbf{z}_i || \mathbf{z}_j\right)\right) \tag{9}$$

where $||$ denotes concatenation, and $\vec{a}$ is a learnable weight vector.

To normalize (9), a softmax function is applied on each node's in-coming edges:

$$\alpha_{ij} = softmax_j\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(e_{ik}\right)} \tag{10}$$

where $\mathcal{N}_i$ is the set of node v_i's one-hop neighbors, including itself.

The above calculation process is shown in Fig. 3, where the 1d-convolution is applied before the linear operation with a learnable kernel τ, and the attention parameters are then calculated among all neighbors of the central node and for each node in the network.

Then, in the output layer, a non-linear function σ is performed to generate the embedding of node v_i:

$$\mathbf{h}'_i = \sigma\left(\sum_{j\in\mathcal{N}(i)}\alpha_{ij}\mathbf{z}_j\right) \tag{11}$$

For model stabilization, multi-head attention mechanism is adopted by concatenating the embeddings from K independent self-attention processes:

$$\mathbf{h}'_i = ||_{k=1}^{K}\sigma\left(\sum_{j\in\mathcal{N}(i)}\alpha_{ij}^{k}\mathbf{z}_j^{k}\right) \tag{12}$$

In this way, the node features are transmitted and aggregated on the central node. After several iterations, the node features will be transmitted to farther neighborhoods.

Finally, with node labels as support information, the model is trained in a semi-supervised way to obtain hyper-parameters. For example, in node classification task, we try to minimize the cross entropy loss between the predictions and the ground truth, the loss function is:

$$L = -\sum_{i=1}^{N} y_i \log\left(\frac{e^{\mathbf{h}'_i}}{\sum_j e^{\mathbf{h}'_j}}\right) \tag{13}$$

where y_i is the label of node i. The network embeddings can then be extracted from the last output layer and used for succeeding network analytical tasks.

The algorithm of CGAT is presented in Algorithm 1 below.

Algorithm 1 CGAT

Input: Heterogeneous information network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, number of nodes N, number of walks w, walk length l, meta-path set $\mathcal{P}$, number of negative samples m, window size q, label set Y, number of attention heads a, number of training epochs k, convolution kernel τ, train data proportion p

Output: Network embedding $\mathbf{X}$

```
1:  Initialize $\mathbf{X}, \tau$
2:  for $i \leftarrow 1$ to $|\mathcal{P}|$ do
3:      for $j \leftarrow 1$ to $N$ do
4:          $\mathcal{M}_j \leftarrow$ MetaPathRandomWalk$(\mathcal{G}, \mathcal{P}_i, v_j, l, w)$
5:      end for
6:      $\mathbf{M}_i \leftarrow$ concat$(\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_N)$
7:      $X_i \leftarrow$EmbeddingPreTraining$(\mathbf{X}, \mathbf{M}_i, m, q)$(Eq.(4))
8:  end for
9:  $\mathbf{X} \leftarrow$ concat$(X_1, X_2, ..., X_{|\mathcal{P}|})$
10: for epoch$\leftarrow 0$ to $k$ do
11:     $\mathbf{H} \leftarrow$ conv1d$(\mathbf{X}, \tau)$ (Eq.(6))
12:     $\mathbf{X} \leftarrow$ GAT$(\mathcal{G}, \mathbf{H}, Y, a, \tau, p)$ (Eq.(8) $\sim$ (13))
13: end for
14: return $\mathbf{X}$
```

IV. Experiments.

A. Datasets

We conduct experiments on two publicly-available HIN datasets: DBLP and ACM. The statistics of the two datasets are summarized in Table I, and the details are listed as follows.

- **DBLP**[1]. A computer science bibliography network which contains 14375 papers (P), 14475 authors (A), 20 venues (V), and 8811 terms (T). The authors, papers, conferences are divided into four areas: *database, data mining, machine learning, and information retrieval*. Three meta-paths, i.e. APA, APVPA, and APTPA are employed for the experiments.
- **ACM**[2]. A bibliography network consists of 12239 papers (P), 17423 authors (A), 13 venues (V), and 767 terms

[1]https://dblp.uni-trier.de/
[2]http://dl.acm.org/

TABLE I
STATISTICS OF THE DATASETS.

Dataset	Relations(A-B)	Number of A	Number of B	Number of A-B	Ave.Degrees of A	Ave.Degrees of B	Meta-paths
DBLP	Paper-Author	14375	14475	41794	2.91	2.89	APA
	Paper-Venue	14375	20	14375	1.00	718.75	APVPA
	Paper-Term	14375	8811	88683	6.17	10.07	APTPA
ACM	Paper-Author	12239	17423	37038	3.03	2.13	APA
	Paper-Conf	12239	13	12239	1.00	941.46	APCPA
	Paper-Term	12239	767	182094	14.88	237.41	APTPA

(T) and the papers are matched to one of eleven labels according to its research area. Three meta-paths, i.e. APA, APVPA, and APTPA are employed for the experiments.

The network schema and meta-path selections for DBLP and ACM are the same as the HIN example shown in Fig. 1 [3].

B. Baseline Methods

We compare CGAT against the following network embedding methods which can be divided into three groups:

a) Homogeneous Network Embedding Methods::

- **DeepWalk** [7]: A classic random walk based homogeneous network embedding method. Here we run DeepWalk on the whole network ignoring the different node types.
- **LINE** [8]: An edge modeling based homogeneous network embedding method which directly learn vertex representations from vertex-vertex connections. We also run LINE on the whole network ignoring the different node types.

b) HIN Embedding Methods::

- **metapath2vec** [10]: A heterogeneous network embedding method which adopts meta-paths and heterogeneous Skip-Gram to learn network semantic features. It can only learn one type of semantics at a time, and only produce node embeddings of certain node types which are included in the meta-paths.
- **HIN2VEC** [11]: A shallow neural network-based embedding method for HINs. It consists of a single-layer neural network to learn network embedding via binary classification on link types.

c) Semi-supervised Learning Method::

- **GCN** [16]: Graph convolutional network, a semi-supervised deep neural network model for graphs which requires node features as input. Here we initialize node features with attribute information of the network. For node type ”P”, we use paper abstract as node feature, for node type ”A”, we use the abstracts of papers linked to the author as node feature, and for node type ”C”, we use the conference descriptions from the website as node feature. We run GCN on the whole network ignoring the different node types.
- **GAT** [17]: Graph attention network, a semi-supervised deep neural network model for graphs which requires node features as input. we initialize node features the same way as described in GCN. We also run GAT on the whole network ignoring the different node types.

  The experiments of GCN and GAT are conducted based on the Deep Graph Library (DGL)[4].

C. Parameter Settings

For methods containing random walks and negative sampling, including DeepWalk, metapath2vec, and HIN2VEC, the walk length is set to 50, and the walks are repeated 50 times per node, the window size and number of negative samples per node are set to 7 and 5, respectively. The embedding size for all baselines is set to 128. For LINE, first-order and second-order proximity is used, and the starting value of learning rate is 0.025; for metapath2vec, the meta-path schema is set as ”APVPA” for DBLP, and ”APCPA” for ACM. The parameters in GAT are set the same as the proposed model below.

For the proposed model, the embedding size of the heterogeneity embedding training module is set to 128. For the graph attention network module, the number of attention heads is set to 16, and the number of hidden units is set to 8. For the output layer, the network embedding is set as the concatenation of all attention heads. Leaky ReLU is employed as the non-linear activation function across all layers with a negative slope of 0.25. For each dataset, we split the set of nodes with ground-truth labels into train set (60%), validation set (10%), and test set (30%). The learning rate is 0.005 and we train both datasets for 200 epochs. To test model generalization capability, the nodes in test set are used for succeeding evaluations.

D. Node Classification

We first evaluate the effectiveness of our proposed method on multi-class node classification task. For a fair comparison, we adopt a logistic regression classifier with stochastic average gradient descent for all methods. We train the classifier with different ratios of labeled data ranging from 20% to 80%. For each training ratio, we randomly split train and test set and repeat the training process 10 times. The average F1-macro and F1-micro scores on two datasets are displayed in Table II.

[3]the term ”Venue” in DBLP and ”Conf” in ACM refer to the same concept in which papers are published, hence we think the two networks share the same network schema.

[4]https://github.com/dmlc/dgl

TABLE II
NODE CLASSIFICATION RESULT ON DBLP AND ACM DATASETS

Dataset	Metric	Train ratio	DeepWalk	LINE	metapath2vec	HIN2VEC	GCN	GAT	CGAT
DBLP	F1-macro	20%	0.9123	0.8894	0.9317	0.9196	0.7930	0.8539	**0.9352**
		40%	0.9215	0.8952	0.9333	0.9286	0.8649	0.8471	**0.9378**
		60%	0.9243	0.8989	0.9345	0.9320	0.8300	0.8604	**0.9384**
		80%	0.9273	0.9010	0.9322	0.9364	0.8618	0.7042	**0.9405**
	F1-micro	20%	0.9179	0.8975	0.9358	0.9246	0.8125	0.8593	**0.9389**
		40%	0.9266	0.9027	0.9370	0.9330	0.8737	0.8650	**0.9415**
		60%	0.9294	0.9061	0.9382	0.9361	0.8918	0.9201	**0.9415**
		80%	0.9322	0.9091	0.9363	0.9406	0.8710	0.9412	**0.9438**
ACM	F1-macro	20%	0.3160	0.3001	0.3342	0.3179	0.3266	0.3205	**0.3422**
		40%	0.3303	0.3151	0.3397	0.3331	0.3254	0.3348	**0.3468**
		60%	0.3345	0.3174	0.3424	0.3422	0.3311	0.3463	**0.3480**
		80%	0.3285	0.3284	0.3400	0.3516	0.3181	0.3136	**0.3525**
	F1-micro	20%	0.7095	0.7045	0.7408	0.7127	0.7243	0.7279	**0.7517**
		40%	0.7221	0.7148	0.7492	0.7285	0.7287	0.7367	**0.7567**
		60%	0.7284	0.7203	0.7497	0.7335	0.7392	0.7503	**0.7583**
		80%	0.7236	0.7322	0.7504	0.7393	0.7163	0.7123	**0.7642**

From the results, we can observe that the proposed CGAT model exhibits the best performance among the baselines. For DBLP, heterogeneous network embedding methods, i.e., metapath2vec and HIN2VEC, are superior to homogeneous network embedding methods, i.e., DeepWalk, LINE, GCN and GAT. CGAT performs better than all the other baselines. For ACM, CGAT still achieves the best results. HIN2VEC and metapath2vec are more effective compared to other baselines but still outperformed by CGAT. For both datasets, the performance of GCN and GAT are the worst among all methods, which shows that the two models can not perform well on HINs even with appropriate initial node features. It is notable that for ACM, the F1-macro values are much lower than F1-micro values, which is possible because ACM dataset has a larger label space than DBLP, and some labels didn't appear in prediction during testing, which results in zeroes when calculating F1-macro with the same averaging number, hence the F1-macro values become lower for all methods.

E. Node Clustering

We then perform node clustering task on the network embeddings, where KMeans is employed as the clustering algorithm. The quality of clustering is evaluated with Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI), and the clustering quality is better when both metrics are higher. The results are displayed in Table III.

The results indicate that the proposed CGAT model yields the best clustering results on both datasets. Also, We can observe that DeepWalk and metapath2vec perform relatively better than other baselines. While HIN2VEC, GCN, and GAT produce the worst clustering performance.

TABLE III
NODE CLUSTERING RESULTS ON TWO DATASETS

Dataset	Method	ARI	NMI
DBLP	DeepWalk	0.7335	0.6998
	LINE	0.1631	0.2131
	metapath2vec	0.7497	0.7049
	HIN2VEC	0.0054	0.0083
	GCN	0.0164	0.1263
	GAT	-0.0002	0.0336
	CGAT	**0.7618**	**0.7208**
ACM	DeepWalk	0.1724	0.2618
	LINE	0.1335	0.2247
	metapath2vec	0.1804	0.2791
	HIN2VEC	0.1166	0.2063
	GCN	0.0162	0.0202
	GAT	-0.0455	0.0391
	CGAT	**0.1882**	**0.2841**

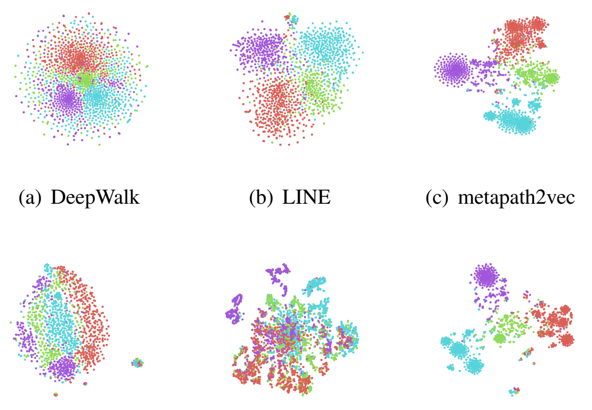
F. Network Visualization

To evaluate the ability of CGAT in generating comprehensible results, we conduct network visualization experiments using the t-SNE [22] model to map high-dimensional node representations into a 2D space. The 2D representation results on DBLP are shown in Fig. 4, where colors indicate different classes.

From the results, we can observe that GAT performs the worst which cannot separate nodes with different labels well. GCN also has similar visualization performance as GAT, hence we do not include it here. DeepWalk, LINE, and HIN2VEC perform better than GAT, yet they still have unclear bounds for the four classes. Metapath2vec can differentiate the classes well, but still, some nodes are mixing in the center. It is shown that our proposed CGAT model can distinguish different nodes with minimum overlay in boundaries, hence our model can be applied to visualization-related tasks with satisfying performance.

G. Parameter Sensitivity

We further analyze the parameter sensitivity of CGAT on the following parameters: (1) the embedding dimension; (2) the


(a) DeepWalk (b) LINE (c) metapath2vec

(d) HIN2VEC (e) GAT (f) CGAT

Fig. 4. The t-SNE 2D representations on DBLP.

number of graph attention heads; (3) the number of training epochs; (4) the labeled data proportion for semi-supervised training. The results are depicted in Fig. 5.

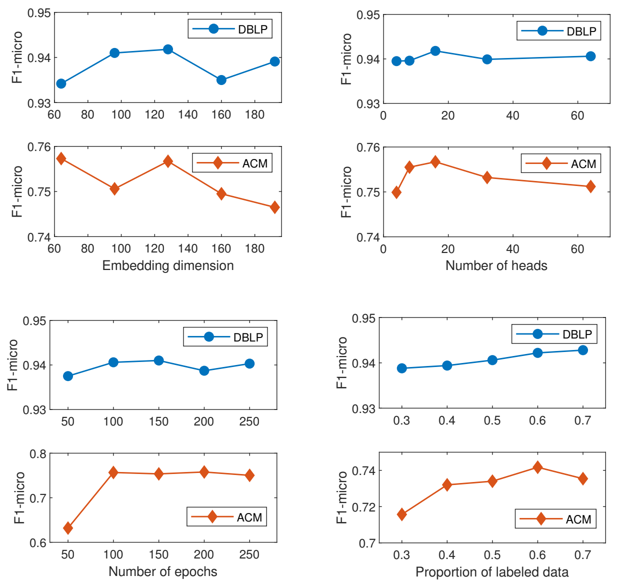


Fig. 5. Parameter sensitivity analysis of CGAT model.

It can be observed that, with the embedding dimension changes from 64 to 192, the performance fluctuates for both datasets, and the embedding dimension of 128 yields the highest F1-micro values. Besides, when the number of attention heads increases from 4 to 64, the F1-micro values in both datasets first increase then decrease, indicating that the model with 16 attention heads produces the best performance. For the number of training epochs, it shows that for DBLP, when the training epoch increases, the F1-micro values also increase a little, but without too much improvement. For ACM, at first, when the number of training epochs increases from 50 to 100, the performance drastically improves, and when the number of training epochs increases from 100 to 250, the performance stays nearly unchanged. This implies that for different datasets, the number of epochs for the model to convergence is different. Lastly, we evaluate the influence of the amount of labeled data from 30% to 70%. It can be observed that with more data used for training, the model performance improves for both datasets.

V. Related Works.

A. *Homogeneous network embedding*

Over the past few years, a significant amount of progress has been made towards homogeneous network embedding [23]. For instance, DeepWalk [7] considers nodes as words and uses truncated random walk sequences to learn network embeddings. Similarly, node2vec [9] extends DeepWalk by adopting biased random walk. LINE [8] proposes an edge modeling based method and directly learns node representations from linking relations. To learn highly non-linear network properties, SDNE [24] and DNGR [25] utilize deep auto-encoders to learn latent node representations.

B. *Heterogeneous information network embedding*

Recently, some embedding models on heterogeneous information networks are also proposed. Metapath2vec [10] extends the random walk process to meta-path based sampling and heterogeneous skip-gram model. HIN2VEC [11] utilizes shallow neural networks to capture semantic information in HINs. BL-MNE [12] adopts deep auto-encoders to learn different semantic features under different meta-paths. To sum up, these methods either only focus on specific meta-path-related semantics or neglecting the high-level proximity information in HINs. Besides, they all work in an unsupervised way without utilizing the label information in HINs.

C. *Graph neural networks*

Graph Neural Networks (GNN) are semi-supervised learning models that exhibit outstanding performance in graph-related tasks, such as graph embedding, graph classification, and so on [26], [27]. State-of-the-art GNN models include Graph Convolutional Networks (GCN) [16] which employs neighborhood aggregation and utilizes multiple depths of the model to capture high-order information. GraphSAGE [28] samples a node's local k-hop neighborhood with fixed size and derives the central node's final state by aggregating its neighbors' features. FastGCN [29] further improves the performance of GraphSAGE with importance sampling to make the model more efficient. To conclude, although the GNN models mentioned above have achieved remarkable results in network analytical tasks, they are designed for attributed homogeneous networks and cannot be directly applied to HINs. Recently some GNN models for HIN embedding are also proposed. For example, HAN [18] incorporate semantic-level attention and node-level attention mechanism to learn node embeddings in HINs. M-HIN [30] utilizes nodes and metagraphs between them to construct HIN triplets, and apply the Hadamard function to describe the relationship between nodes and metagraphs. HetGNN [19] proposes an unsupervised model that samples a fixed length of heterogeneous neighbors and performs type-based neighbors aggregation, and heterogeneous types combination to learns node embeddings.

VI. Conclusion and Future Work.

In this paper, we propose CGAT, a HIN embedding model which aims at preserving rich semantics and high-level proximity in HINs. By utilizing 1d-convolution for multi-semantic assembling and optimized graph attention network for information aggregation and semi-supervised learning, the model is capable of preserving rich semantic information as well as high-level proximity in HINs, which has not been jointly considered before. Our model achieves satisfying results in two real-world HIN datasets on three network analysis tasks, which demonstrates its effectiveness. In the future, we hope to tackle the problem of embedding learning on HINs with richer side information, such as HINs with both node attributes and link attributes, and develop effective learning methods to better analyze and discover latent knowledge in HINs.

Acknowledgment

This paper is supported by the National Key Research and Development Program of China (Grant No. 2018YFB1403400), the National Natural Science Foundation of China (Grant No. 61876080), the Key Research and Development Program of Jiangsu (Grant No. BE2019105), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

References

[1] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, "Harp: Hierarchical representation learning for networks," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[2] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017, pp. 731–739.

[3] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[4] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 377–386.

[5] F. Zhou, B. Wu, Y. Yang, G. Trajcevski, K. Zhang, and T. Zhong, "Vec2link: Unifying heterogeneous data for social link prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 1843–1846.

[6] H. Chen, H. Yin, W. Wang, H. Wang, Q. V. H. Nguyen, and X. Li, "Pme: projected metric embedding on heterogeneous networks for link prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1177–1186.

[7] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[8] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 1067–1077.

[9] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 855–864.

[10] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2017, pp. 135–144.

[11] T. Fu, W. Lee, and Z. Lei, "Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 2017, pp. 1797–1806.

[12] J. Zhang, C. Xia, C. Zhang, L. Cui, Y. Fu, and P. S. Yu, "Bl-mne: Emerging heterogeneous social network embedding through broad learning with aligned autoencoder," in *IEEE International Conference on Data Mining*, 2017.

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

[14] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] J. B. Lee, R. Rossi, and X. Kong, "Graph classification using structural attention," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1666–1674.

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

[17] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *ICLR 2018*, 2018.

[18] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference*. ACM, 2019, pp. 2022–2032.

[19] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*, 2019, pp. 793–803.

[20] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[22] V. D. M. Laurens and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2605, pp. 2579–2605, 2008.

[23] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 833–852, 2019. [Online]. Available: https://doi.org/10.1109/TKDE.2018.2849727

[24] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 1225–1234.

[25] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[26] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia, "Graph networks as learnable physics engines for inference and control," *arXiv preprint arXiv:1806.01242*, 2018.

[27] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, "Cross-lingual knowledge graph alignment via graph convolutional networks," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 349–357.

[28] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[29] J. Chen, T. Ma, and C. Xiao, "Fastgcn: fast learning with graph convolutional networks via importance sampling," *arXiv preprint arXiv:1801.10247*, 2018.

[30] Y. Fang, X. Zhao, P. Huang, W. Xiao, and M. de Rijke, "M-hin: Complex embeddings for heterogeneous information networks via metagraphs," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 913–916.