2019 IEEE International Conference on Big Data (Big Data)

Metapath Enhanced Graph Attention Encoder for HINs Representation Learning

Yuwei Fu
Shanghai Key Laboratory of Data Science, School of Computer Science Fudan University
ywfu17@fudan.edu.cn

Yun Xiong
Shanghai Institute for Advanced Communication and Data Science Shanghai Key Laboratory of Data Science, School of Computer Science Fudan University
yunx@fudan.edu.cn

Philip S. Yu
Department of Computer Science University of Illinois at Chicago, USA
psyu@uic.edu

Tianyi Tao
Shanghai Key Laboratory of Data Science, School of Computer Science Fudan University
tytao17@fudan.edu.cn

Yangyong Zhu
Shanghai Institute for Advanced Communication and Data Science Shanghai Key Laboratory of Data Science, School of Computer Science Fudan University
yyzhu@fudan.edu.cn

***Abstract*—In this paper, we propose a novel representation learning framework, named MEGAE, for heterogeneous information networks. To investigate the rich semantic information in heterogeneous information networks, we use metapaths to complete implicit links between nodes. A graph attention encoder is further used to learn graph structural information with shared weight parameters. The attention mechanism, on the other hand, provides us an intuition of how the representation is learned and improves the interpretability of our model. Furthermore, a multitask learning of node classification and link prediction is trained to achieve more robust generalization ability. To validate our ideas, extensive experiments on three real-world datasets show that our model achieves state-of-the-art results on node classification and link prediction tasks in HINs.**

***Index Terms*—graph neural networks, heterogeneous information networks, representation learning**

I. Introduction

Graph-structured data are omnipresent in real-world applications, *e.g.*, online social networks, bibliographic networks, and knowledge graphs. Network representation learning aims to learn low-dimensional node embeddings which encode high-dimensional graph structural information. Researchers have found that such low-dimensional embeddings are extremely useful as input features for downstream graph-based learning tasks, such as node classification [1], [2], node clustering [3], [4] and link prediction [5], [6].

To capture graph structural information, classic methods often depend on graph statistics (*i.e.* degree and centrality), graph kernels, or handcrafted features [7]. Many early works on network representation learning problem resort to matrix factorization algorithms [8]–[10] for dimension reduction. Inspired by the successful language model word2vec [11], many methods learn node embeddings through random walks on graphs with Skip-Gram models, such as DeepWalk [12], node2vec [3], and LINE [5]. Recent years have seen a continuing surge of interest in generalizing deep neural networks to graph-structured data [2], [13]–[15].

However, these works mainly focus on representation learning for homogeneous information networks which contain a single type of entities and relations. Yet most real-world data can be naturally represented as heterogeneous information networks (HINs) [16] with various types of entities and relations. To employ the rich semantic information embedded in HINs, one popular way is to use the metapaths [17]. A metapath is a sequence of relational links which represents a specific semantic relation, and many researchers have found it beneficial to incorporate metapaths in different data mining tasks on HINs, like similarity search [18], collective classification [19] and recommendation [20]. Recently, many metapath-guided random walk methods are introduced for learning node embeddings on HINs, such as metapath2vec [21], HIN2vec [22] and metagraph2vec [23].

In this paper, we focus on representation learning task on heterogeneous information networks, which is a challenging problem due to the following reasons:

Shallow Models: Prevalent random walk based approaches [21], [24] are shallow embedding models that directly map nodes to node vectors which are equivalent to embedding lookup tables. In shallow models, each node embedding vectors is trained independently and there is no parameter sharing between nodes. This can be statistically inefficient for large graph because its space complexity is $\mathcal{O}(|\mathcal{V}|)$.

Multiple Training Steps: The training procedure of existing models for HINs representation learning is usually divided

978-1-7281-0858-2/19/$31.00 © 2019 IEEE

into two stages [21], [22]. The first stage is to learn node embeddings through a metapath-guided random walk, and the second stage is to train another model for different downstream tasks individually. However, this kind of multi-step training method can bring more noise and lead to poor training stability and lower model performance. Further, even though some methods [2] are trained in an end-to-end way, they are usually only designed for a single task. We have to modify the model so as to tackle different tasks.

Lack of Interpretability: Interpretability is crucial for us to understand how and why a model works. Unfortunately, a few number of methods try to open the black box behind their models. In terms of representation learning on heterogeneous information networks, existing methods can not tell us what's the distinction between each metapaths, or what's the difference between different kinds of nodes and links.

To overcome the above-mentioned challenges, we propose a metapath enhanced graph attention encoder model for representation learning on heterogeneous information networks. In our model, we first add some virtual edges according to a set of predefined metapaths to capture hidden semantic information in HINs. Secondly, we use a novel graph attention encoder to learn graph structural information by focusing on more relevant nodes in the first-order neighborhood. Lastly, we use a scoring function as graph decoder for link prediction and train two different tasks end-to-end simultaneously in a multitask learning setting. We summary our main contributions as follows:

1) We propose a unified framework for representation learning on HINs which uses extensible graph encoders for learning graph structure information and uses metapaths to learn different semantic information.
2) We train our model in a multitask learning procedure which is suitable for (semi-)supervised learning and unsupervised learning.
3) For model interpretability, we utilize a graph attention encoder to visualize the relative importance of different neighboring nodes.
4) Extensive experiments show that our proposed model can achieve the state-of-the-art results in node classification task and link prediction task.

II. PRELIMINARIES

In this section, we provide some backgrounds and formally define the problem of presentation learning on HINs.

Definition 1: A **heterogeneous information network (HIN)** [17], [18] is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. The number of node types $|A| > 1$ or the number of edge types $|R| > 1$.

As shown in Figure 1, the citation network involves three types of nodes, *i.e.*, paper, author and conference, and three types of links: $Author\ F \xrightarrow{write} Paper\ C$, $Paper\ B \xrightarrow{publish} Conference\ E$ and $Paper\ C \xrightarrow{cite} Paper\ A$.

Definition 2: A **metapath** [17], [19] $\mathcal{P}$ is a sequence of objects linked by relations in the form of $\mathcal{V}_1 \xrightarrow{\pi_1} \mathcal{V}_2 \xrightarrow{\pi_2} \cdots \mathcal{V}_t \xrightarrow{\pi_t} \pi_{t+1} \cdots \xrightarrow{\pi_{l-1}} \mathcal{V}_l$, where $\pi = \pi_1 \circ \pi_2 \circ \cdots \circ \pi_{l-1}$ denotes the composite relations between node types $\mathcal{V}_1$ and $\mathcal{V}_l, \mathcal{V}_i \in \mathcal{V}$.

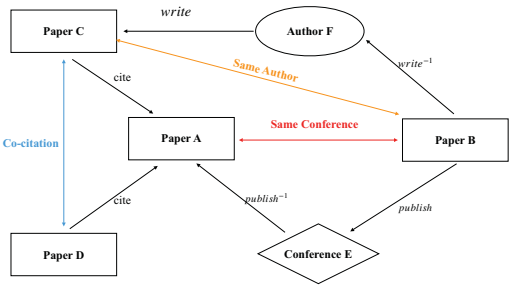


Fig. 1. A citation network with multiple types of nodes and edges.

Metapaths can be viewed as implicit links in HINs which represent specific semantic meanings between two nodes. For example, in Figure 1, a metapath $P \xrightarrow{publish} C \xleftarrow{publish^{-1}} P$ denotes a composite relationship between *paper* and *conference*, and the semantic meaning is that two *papers* are published in a same conference.

Definition 3: **Heterogeneous Information Network Embedding.** The goal of representation learning on HINs $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is to learn a mapping funtion $f_v : \mathcal{V} \rightarrow \mathbb{R}^{d_v}$ or $f_e : \mathcal{E} \rightarrow \mathbb{R}^{d_e}$ which maps node features or edge features to a dense low dimensional vector $(d_v \ll |\mathcal{V}|)$ *or* $(d_e \ll |\mathcal{E}|)$. These learned low dimension embeddings are then inputted to various downstream data mining tasks in HINs, like classification, clustering or link prediction [21], [22].

III. METHODOLOGY

We introduce a general framework, named MEGAE (**M**etapath **E**nhanced **G**raph **A**ttention **E**ncoder), for heterogeneous information networks representation learning.

A. Metapath-based Virtual Edges

Semantic meaning relations are implicit and mingled in HINs, and how to exactly capture disparate semantic meaning relations between nodes is a challenging task. Unlike some previous methods [25] which convert HINs to a *multi-channel network*, MEGAE adds metapath-based virtual edges directly to the original graph to capture the hidden semantic information. Further, our method explicitly uses an edge feature to express the semantic information of the metapath. A toy HIN with eight nodes and three predefined metapaths (MP1, MP2, MP3) is illustrated in Figure 2. For example, node 1 and node 2 are connected with MP1 and MP3, so that we add two virtual edges between node 1 and node 2. Note that as we allow two nodes to be connected with multiple types of edges, we can express each edge as a one-hot vector, like edge feature between node 1 and node 2 can be expressed as $e_{12} = (0, 0, 0, 1, 0, 1)$.

By adding virtual edges between nodes, we can use edge features to depict semantic meaning relations straightforwardly. Suppose we have T types of edges, then each edge

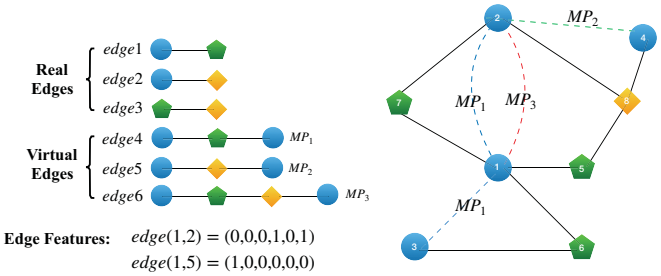


Fig. 2. A toy HIN with three different types of nodes and three different types of edges. We first define three different metapaths, then we look for nodes connected with each metapath and add a virtual edge between them. At last, we will have six different types of edges, and the edge feature between two nodes can be expressed by a six-dimension vector.

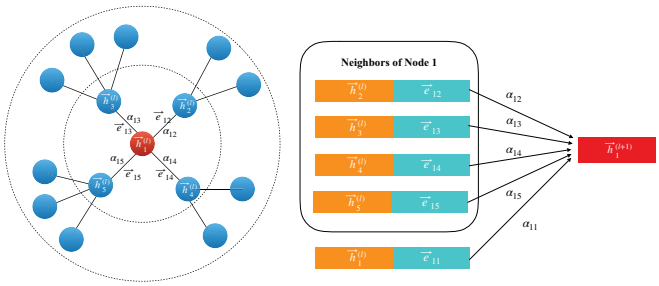


Fig. 3. A variant of the graph attention network is used as the graph encoder. An attention mechanism which leverages node features and edge features helps to focus on the most relevant nodes in the neighborhood.

can be defined as an one-hot vector $e_{ij} \in \mathbb{R}^T$. With the edge embedding matrix $\mathbf{T} \in \mathbb{R}^{E \times d_e}$ where d_e is the dimension of each edge embedding, we can express the edge embedding $\vec{e}_{ij}$ between node i and node j as $\vec{e}_{ij} = \mathbf{E}e_{ij}$. The edge embedding matrix $\mathbf{E}$ is a parameter which is trained along with the model.

B. Graph Attention Encoder

We denote a HIN as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $v_i \in \mathcal{V}$ and edges $(v_i, e_{ij}, v_j) \in \mathcal{E}$, where e_{ij} is the edge type. Our graph encoder is motivated by *graph attention network* (GAT) [15] which uses attention mechanism to learn node features from local neighborhoods and focus on the most relevant ones. It can be understood as a special case of the differentiable message passing neural networks framework [13]. In the message passing neural networks framework, the node features are updated iteratively by two phases, a message aggregating phase, and a feature update phase.

We denote $\vec{h}_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ as the hidden state of node v_i in the l-th layer of the neural network, with $d^{(l)}$ being the dimensionality of this layer's representations. For each node i, we first use the message aggregation function g_m to learn a message based on its neighbors' features. The aggregated message is then passed to an update function with the previous hidden state to get an updated hidden state:

$$\vec{m}_i^{(l+1)} = \sum_{j \in \mathcal{N}_i} g_m \left(\vec{h}_i^{(l)}, \vec{h}_j^{(l)}, \vec{e}_{ij} \right) \tag{1}$$

$$\vec{h}_i^{(l+1)} = U_t \left(\vec{h}_i^{(l)}, \vec{m}_i^{(l+1)} \right) \tag{2}$$

where $\mathcal{N}_i$ denotes the neighbors of node i and $\vec{e}_{ij}$ is the edge embedding of the edge between node i and node j.

In our model, we use a variant of *graph attention network* (GAT) [15] as the message aggregation function. Figure 4 presents the detail of the graph attention encoder. To learn graph structural information, we use an attention mechanism based on node i and its first-order neighbors $\mathcal{N}_i$. For node i and one of its neighbor node j, we use $\vec{x}_j^{(l)}$ to represent the neighbor feature of node j in the l-th layer in the neural network: $\vec{x}_j^{(l)} = \mathbf{W}_e \vec{e}_{ij} + \mathbf{W}_n \vec{h}_j^{(l)}$, where $\mathbf{W}_e$ is the linear transformation for the edge embedding and $\mathbf{W}_n$ is the linear transformation for the node embedding. The attention coefficients α_{ij} between node i and j is calculated by:

$$\alpha_{ij}^{(l)} = \frac{\exp(\sigma(a([\vec{x}_i^{(l)} || \vec{x}_j^{(l)}])))}{\sum_{k \in \mathcal{N}_i} \exp(\sigma(a([\vec{x}_i^{(l)} || \vec{x}_k^{(l)}])))} \tag{3}$$

where the $\sigma(\cdot)$ is a nonlinear activation like ReLU function, $a(\cdot, \cdot)$ denotes a fully connected layer mapping feature vectors to a scalar and $||$ is the concatenation operation. For the target node i, we add a self-loop edge $\vec{e}_{ii}$ to calculate $\vec{x}_i^{(l)}$. Once obtained the attention coefficients, a linear combination of neighbor features is calculated as the message $\vec{m}_i^{(l+1)}$ for node i:

$$\vec{m}_i^{(l+1)} = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{(l)} \vec{x}_j \right) \tag{4}$$

Further, we use an update transformation $\mathbf{W}_u$ to calculate new node embedding for node i with a nonlinear activation (e.g., ReLU), and use a linear activation at the output layer:

$$\vec{h}_i^{(l+1)} = \sigma \left(\mathbf{W}_u \cdot [\vec{h}_i^{(l)} \| \vec{m}_i^{(l+1)}] \right) \tag{5}$$

We define a graph attention encoder GAE to represent the combination of message aggregating phase and message updating phase. To calculate the updated node embedding $\vec{h}_i^{(l+1)}$ for node i, we input the hidden state $\vec{h}_i^{(l)}$ and edge embeddings $\vec{e}_{i,j}$ to the encoder: $\vec{h}_i^{(l+1)} = \text{GAE}_l \left(\vec{h}_i^{(l)}, \vec{h}_j^{(l)}, \vec{e}_{ij} \right), j \in \mathcal{N}_i$, where $\mathcal{N}_i$ denotes the neighbors of node i. We use raw node features as the initial input $\vec{h}_i^{(0)}$.

Intuitively, our graph attention encoder accumulates transformed feature vectors of neighboring nodes through a weighted sum. We introduce a neighbor feature $\vec{x}_j^{(l)}$ which depends on both of the node embedding and edge embedding. We utilize the node neighbors to learn graph structural information and use edge embeddings to learn semantic meaning relations. In our framework, we stack two layers of the graph attention encoder to map each node $v_i \in \mathcal{V}$ to a low-dimensional node embedding $z_i \in \mathbb{R}^d$. The learned low dimensional latent node embeddings can then be efficiently applied to different downstream tasks.

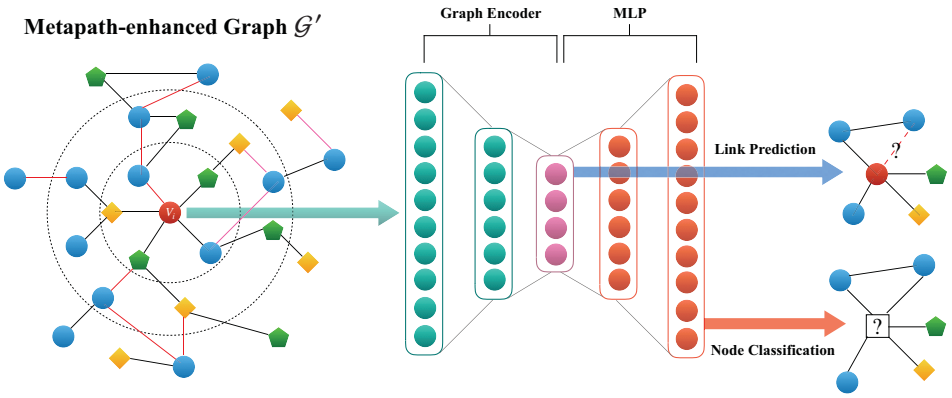


Fig. 4. A visual illustration of the MEGAE model.

C. Multitask Learning for Node Classification and Link Prediction

Inspired by recent promising results achieved by multitask learning [26], [27]. We apply a multitask learning setting for represent learning on HINs, we use node classification task and link prediction task in our multitask learning framework.

Node classification task aims at predicting labels of nodes, such as the type of a movie or the category of a book. We can simply stack two graph attention encoders with a softmax$(\cdot)$ activation for multi-class classification or a sigmoid$(\cdot)$ activation for multi-label classification. We then evaluate the cross-entropy loss over all labeled nodes in the network: $\mathcal{L}_{node} = -\sum_{i\in\mathcal{Y}}\sum_{k=1}^{K} y_{ik}\ln z_{ik}$, where z_{ik} denotes the k-th entry of the encoder output for the i-th labeled nodes, and $\mathcal{Y}$ is the set of nodes with labels. y_{ik} is its respective ground truth label.

Link prediction task attempts to answer the question of how likely are two nodes being connected. In this paper, we mainly focus on the structural link prediction problem, where the task is to predict the likelihood of a missing or unobserved edge exists between two nodes. To solve the link prediction problem, we introduce a graph auto-encoder model which is comprised of above-mentioned graph attention encoder and a scoring function as a decoder. The decoder uses the output node embeddings to reconstruct network edges through a scoring function $f : \mathbb{R}^d \times \mathbb{R}^e \times \mathbb{R}^d \to \mathbb{R}$. Unlike most previous link prediction methods that use a real-valued vector for each node v_i as parameters and optimize directly during training process [28], [29], we predict links based on the output from the graph attention encoder so that our model can leverage graph structural information to make more precise predictions. We choose the DistMult factorization [29] as the scoring function which is known to perform well on link prediction tasks. For each type of edges, a diagonal matrix R_e is used in DistMult factorization: $f(v_i, e_{ij}, v_j) = z_i^\top R_e z_j$.

In the experiments, we select one specific type of original edges to predict, and we train the link prediction model with negative sampling. For each observed edge we sample w negative edges by randomly replace v_i or v_j in the (v_i, e_{ij}, v_j), and we use cross-entropy loss to push the model give higher scores for the positive edges than negative edges:

$$\mathcal{L}_{edge} = -\sum_{(v_i,e_{ij},v_j)\in\mathcal{T}} y\log\sigma\left(f(v_i,e_{ij},v_j)\right)\eta + (1-y)\log\left(1-\sigma(f(v_i,e_{ij},v_j))\right) \tag{6}$$

where $\mathcal{T}$ is the set of positive triplets $(y = 1)$ and negative triplets $(y = 0)$, and $\sigma(\cdot)$ is the sigmoid activation. We define a weighting factor $\eta = \frac{\#\text{absent links}}{\#\text{present links}}$ in the edge prediction loss to alleviate the class imbalance problem.

The semi-supervised node classification task and link prediction task are trained simultaneously by sharing the same encoder output z_i. Then the total loss for the multitask learning problem can be written as the sum of node classification loss, edge loss, and regularization loss: $\mathcal{L} = \mathcal{L}_{node} + \mathcal{L}_{edge} + \mathcal{L}_{reg}$.

In addition, we use a similar loss function as in [26] which can learn different loss weights automatically by considering the homoscedastic uncertainty of each task. Then the overall loss can be written as:

$$\mathcal{L} = \frac{1}{2\sigma_1^2}\mathcal{L}_{node} + \frac{1}{2\sigma_2^2}\mathcal{L}_{edge} + \log\sigma_1 + \log\sigma_2 + \mathcal{L}_{reg} \tag{7}$$

where σ_1 and σ_2 are two uncertainty coefficients.

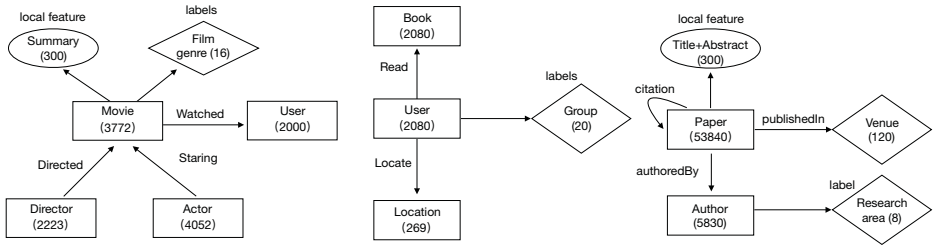
Under this multitask learning setting, our model can perform two tasks at the same time in an end-to-end method. The losses from different tasks also play a role as an implicit regularization, which makes the multitask model less likely to overfit and improve generalization performance.

IV. Experiments

A. Datasets

We validate the performance of our model on multi-label classification task and link prediction task in three real-world datasets (Table I).

• **Movie Dataset:** The first dataset is a movie dataset which contains four different types of nodes and three types of connections/links as shown in Figure 5(a). For each movie, we extract a bag-of-words representation of the plot summary as



(a) IMDB Dataset (b) Douban Dataset (c) Aminer Dataset

Fig. 5. Schema of three datasets used in the experiments.

local features. Our target instance type is the movie instance, which is assigned with multiple genres as labels. In the experiments, we use three different metapaths: M-A-M, M-D-M, and U-M-U.

• **Douban Dataset:** Our second dataset is a book review dataset which has three types of nodes: book, user and location, and two types of connections/links: reading link and locating link. Our target instance type is the user instance, which is tagged with one or more group labels. The related network schema is shown in Figure 5(b). In the experiments, we use two metapaths: U-B-U and U-L-U.

• **Aminer Dataset:** Our last dataset is a bibliographic information dataset and the corresponding network schema is displayed in Figure 5(c). This network includes three types of nodes and three types of connections/links. We extract a bag-of-words of the paper title and abstract as local features, which include 300 words. Our target instance type is the author instance, which is assigned with one or more research areas labels. In the experiments, we use two different metapaths: A-P-A and A-P-V-P-A.

TABLE I
SUMMARY OF EXPERIMENTAL DATASETS

	Movie	Douban	Aminer
# Labels	16	20	8
# Feature	300	-	300
# Node Type	4	3	3
# Link Type	3	2	3
# Instance	12047	5709	58790

B. Compared Algorithms

• metapath2vec [21]: A heterogeneous networks representation learning method based on metapath-guided random walks. And metapath2vec++ is a variant of the model based on a heterogeneous negative sampling technique.

• DeepWalk [12]: A homogeneous networks embedding method which learns node vectors by capturing node pairs within w-hop neighborhood via *uniform* random walks.

• node2vec [3]: A generalized model of DeepWalk which learns node vectors through *parameterized* random walks.

• LINE [5]: A homogeneous networks embedding method which considers both the first(Line1) and second(Line2) order proximities of nodes in the network separately.

• ESim [24]: A general similarity search framework for heterogeneous information networks. ESim learns node vectors by paths between nodes along a given metapaths.

• Hin2vec [22]: A neural network model for HINs representation learning which not only learns node embeddings but also learns representations of metapaths.

• MEGAE: This is our model proposed in Section 3. In the experiments, we use the pretrained metapath2vec results as input node embeddings for the nodes without features.

• MAGAE(single): This is a variant version of our model which do each task individually.

C. Multi-label classification of Nodes

In this section, we evaluate the models by multi-label classification of nodes. We first introduce the experimental setup, including the preparation of labeled datasets, default parameters of the compared model and the process of classification. Then we perform sensitivity experiments on parameters of MEGAE to determine our default settings. We finally show the experimental results over three real-world datasets.

1) Experimental Setup: Firstly, given a HIN $\mathcal{G}$ we first add virtual edges to the original network with a set of predefined metapaths as described in section 3 to get a metapath enhanced HIN $\mathcal{G}'$. Secondly, we select one of the original edge type and remove parts of the edges from $\mathcal{G}'$ to keep them unobserved during training for the link prediction task. For node classification experiments, we split 80% of the labeled nodes as a training set, 10% of the labeled nodes as the validation set and 10% of the rest as the test set. We use *micro-f1* score and *macro-f1* score as metrics for evaluation. We apply early stopping with patience of 25, i.e. we stop training if the evaluation metrics on validation set do not increase for 25 consecutive epochs. In practice, we train the model using (full-batch) gradient descent techniques with Adam [30] using a learning rate of 0.01 and weight decay as 1e-5.

In Movie dataset, each movie is labeled by 16 movie genres. We use all 3772 movies as our target instances. In Douban dataset, we use 20 groups as user labels to perform user classification. In Aminer dataset, our target instance is the author nodes. We use 20 research fields as labels, and each of the 120 venues is connected with one of the research filed. The author instance is labeled by the research filed if more than a quarter of his/her publications are related to the research

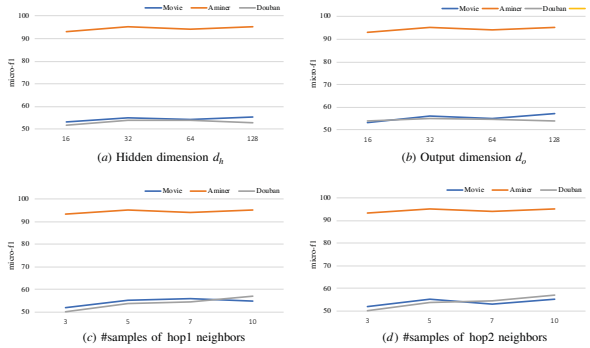


Fig. 6. Parameter Tuning.

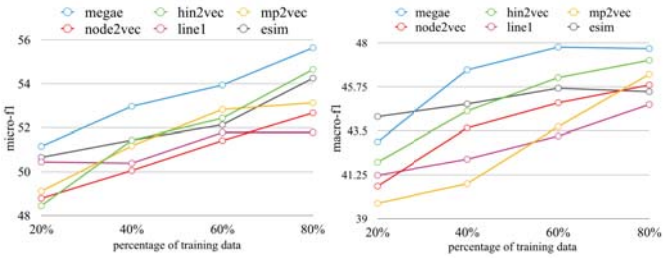


Fig. 7. Node classification performancewith on Douban dataset only part of the training data. Compared with other baselines, MEGAE can learn faster with fewer samples.

fields. We select 5830 authors with at least 10 publications as our target instances.

Regarding default parameters, the dimensionality of node vectors, d, is set to 100 for all approaches. The negative sampling rate w is set to 5. For random walk based methods include DeepWalk, node2vec, ESim, metapath2vec, we set the window size to 5, walk length to 100, walks per node to 50.

2) Parameter Tuning in MEGAE: Parameters settings in MEGAE affect the learned representation and application performance. To decide the default settings, we vary the values of important parameters to observe how the micro-$f1$ changes in node classification in the networks. The results are shown in Figure 6.

Hidden dimension. Firstly, Figure 6(a) shows that setting the number of hidden dimension d_h to be 64 is reasonable. Generally, a small d is not sufficient to capture the information embedded in relationships between nodes, and a large d tends to incur overfitting problem.

Output dimension. Figure 6(b) shows the setting of output dimension d_o is suitable to choose as 32. We follow a similar reason for not choosing a large one as for the size of hidden dimension to avoid overfitting.

Number of sample size. In the experiments, we sample a certain number of nodes for the graph attention encoder instead of passing its whole first-order neighborhood. For the sack of the computational efficiency and performance, we choose the sample size of the first order neighbors to be 10 and the size of second order neighbors to be 5.

Based on these results, We use two layer graph attention encoder in MEGAE, and set the hidden dimension to be 64. The size of the output dimension is chosen as 32.

3) Evaluation of models: We report the performance of average Micro-f1 score and Macro-f1 score for node classification task after 10 runs in Table 2. We can observe from the result that metapath based methods usually perform better than homogeneous network representation learning methods, which proves that it is useful and advantageous to leverage metapath to capture semantic information in HINs to learn better representations. In addition, we can find that compared with the random walk-based models, like metapath2vec and hin2vec, our model MEGAE achieves better results in all three datasets. We argue this this evidence proves the effectiveness of graph encoder to learn better graph structural information.

Further, we can find that MEGAE always achieves better results compared with MEGAE(single), which demonstrates that our multitask learning setting is able to get better generalizatoin ability than single task learning. Figure 7 shows the result of node classification in Douban dataset with only part of the training data. The leading performance of MEGAE shows that our multitask learning method can learn faster with fewer samples, e.g., MEGAE with 40% training data outperforms all other baselines with 60% training data on both micro-f1 and macro-f1. Similar trends holds for the other datasets which is not shown due to space limitation.

TABLE II
SUMMARY OF EXPERIMENTAL DATASETS

Algorithms	Movie		Douban		Aminer	
	mi-f1	ma-f1	mi-f1	ma-f1	mi-f1	ma-f1
MEGAE	**54.67**	**43.08**	**55.16**	**48.72**	**95.56**	**95.36**
MEGAE(single)	51.25	41.54	54.34	45.86	95.12	94.82
Hin2vec	54.11	40.12	52.44	45.29	94.16	93.96
metapath2vec	52.09	39.16	52.18	44.80	95.13	94.51
metapath2vec++	51.56	40.19	51.37	43.39	95.05	94.88
DeepWalk	50.31	38.13	50.93	43.96	92.35	91.74
node2vec	51.92	39.44	51.72	44.53	94.03	93.91
Line1	51.63	39.04	51.44	43.76	91.26	89.42
Line2	51.97	40.35	52.89	42.45	91.34	90.51
ESim	50.15	39.17	52.02	41.51	89.93	89.39

TABLE III
SUMMARY OF EXPERIMENTAL DATASETS

Algorithms	Movie		Douban		Aminer	
	AUC	AP	AUC	AP	AUC	AP
MEGAE	**74.38**	**53.24**	**87.66**	**76.17**	**88.49**	**70.35**
MEGAE(single)	74.08	52.65	86.82	74.74	87.79	69.32
Hin2vec	73.47	51.30	84.45	69.38	89.46	69.71
metapath2vec	73.71	50.75	82.16	65.88	85.67	64.57
metapath2vec++	70.57	51.84	78.65	62.08	84.23	62.02
DeepWalk	70.16	49.20	78.31	62.93	86.83	64.72
node2vec	71.89	51.03	78.12	62.85	80.90	64.56
Line1	70.11	48.99	78.64	61.89	82.02	61.85
Line2	70.22	49.53	79.81	63.08	80.64	56.49
ESim	72.56	51.24	82.57	71.87	80.87	59.56

D. Link prediction

In this section, we describe the details of our experiments on link prediction task. We first introduce the experimental setup including the data preprocessing stage and the metrics for evaluation. Then we show the experimental results of MEGAE in link prediction task compared with other models.

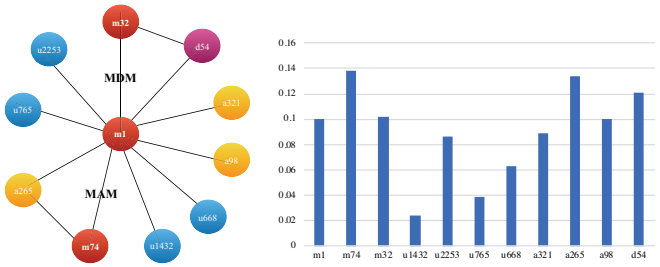
1) Experimental Setup: Since our model employs a multitask learning method to train node classification task and link prediction task at the same time, link prediction task shares the same input data and output representations as in the node classification task. To simulate the unobserved edges, we first select one type of the original edges as the target link and then generate a sub-network by randomly removing 20% of the target edges from the network, and then drop the isolate nodes. In order to save time in the training and inference procedure, we sample w negative edges for each node in the data preprocessing stage. We choose AUC and AP as the evaluation metrics. In experiments, we use the output embeddings of the first encoder layer as input features for link prediction task.

In movie dataset, our target link is the user-movie link and predicts which movie is watched by the user. In Douban dataset, our target link is the user-book link and predicts which book is read by the user. In Aminer dataset, our target link is the paper-paper link and predicts which paper is cited by another paper.

2) Evaluation of models: We report the performance of average AUC and AP for link prediction task after 10 runs in Table 3. MEGAE is able to improve performance than the state-of-the-art models. Compared the results of homogeneous networks embeddings models with heterogeneous networks embeddings models, the homogeneous networks embeddings models sometimes show competitive performance. Which implies in the link prediction task, sometimes the structural proximity information is strong enough to make a pretty good prediction. And we can find that our model can always achieve stable performance in all three different datasets.

E. Visualization

We try to understand how does our model work by visualizing the attention weights of the graph attention encoder. Here, we take movie node m_1 in Movie dataset as an illustrative example (Figure 8). For movie node m_1, we collect the attention weight α_{ij} from the first layer graph attention encoder. As we can see in Figure 8, movie $m74$, actor $a265$ and director $d54$ contribute most to the representation of movie $m1$. This observation obeys the intuition that movie $m1$ is a comedy movie, actor $a265$ is a popular comedy actor and director $d54$ is a director who directs many comedy movies. Compared with the actor nodes and movie nodes, we can find that the user nodes have fewer contribution to the movie representation. This can be explained as these users watched a broad range of movies, so that they don't provide enough meaningful information for learning representation for our target node. Thanks to the attention mechanism, we can easily comprehend how our model learn the node embeddings, and we can locate abnormality more easily.



(*a*) neighbors of node m1 (*b*) attention weights of neighbors

Fig. 8. Visulization of the attention weights.

V. RELATED WORK

Network Representation Learning. Network representation learning aims to learn dense low-dimensional vectors for each node while preserving the original graph structure information. There are two kinds of node embedding approaches: one is the *shallow embedding approach* like *factorization-based algorithm* or *random-walk based algorithm* which maps nodes to vector embeddings directly just as a lookup tabel [10], [12]; the other is the *deep embedding approach* like *graph neural networks* which rely on neural networks to share weights between nodes and leverage node features [2]. Hamilton et al [7] introduced a general encoder-decoder framework for network representation which unifies a number of existing methods like matrix factorization-based methods, random-walk based algorithms and graph neural networks.

Graph Neural Networks. Gori et al [31] first introduced Graph Neural Networks (GNNs) as recursive neural networks. A few work [2], [32], [33] then began to explore the generalizations of CNNs to graphs. Hamilton et al [14] introduced GraphSAGE, a framework which allows learning node embeddings effectively for the inductive learning task. Inspired by the success of attention mechanism applied in NLP tasks, [15], [34] proposed network representation learning models based on attention mechanisms.

Multitask Learning. Multitask learning aims to leverage useful information contained in multiple tasks by sharing representations to improve generalization performance. Common multitask models can be divided into two categories: feature-based approach and parameter-based approach [35]. In a feature-based approach, the same input features are used to learn a shared feature representation among different tasks. While in a parameter-based approach, different tasks are linked by sharing the same model parameters.

VI. CONCLUSION

This paper presents a metapath enhanced graph attention encoder model for heterogeneous information networks representation learning. Our proposed model, MEGAE, combines metapath and graph encoder to learn both semantic information and graph structural information in HINs. A variant

of the graph attention network is employed to learn the graph structural information by focusing on the most relevant neighboring nodes. Further, we train MEGAE in a multitask learning setting which is capable of doing different tasks simultaneously while achieving better generalization performance. Our approach uses predefined metapaths to capture semantic meaning relations in HINs. To select meaningful metapaths requires a lot of expert knowledge, so how to automatically generate meaningful metapaths for different HINs is an interesting problem to solve. In addition, our model applies a uniform sampling method, other nonuniform sampling methods could be a direction for future study.

VII. Acknowledgment

This work is supported in part by the National Natural Science Foundation of China Projects No.U1636207, 91546105, the NSF under grants III-1526499, III-1763325, III-1909323, CNS-1930941, CNS-1626432, and the Shanghai Science and Technology Development Fund No.19DZ1200802, 16JC1400801.

References

[1] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, p. 93, 2008.

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

[3] 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.

[4] Y. Sun, B. Norick, J. Han, X. Yan, P. S. Yu, and X. Yu, "Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 7, no. 3, p. 11, 2013.

[5] 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*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.

[6] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," *arXiv preprint arXiv:1802.09691*, 2018.

[7] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[8] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[9] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 891–900.

[10] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 1105–1114.

[11] 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.

[12] 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*, ser. KDD '14. New York, NY, USA: ACM, 2014, pp. 701–710. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623732

[13] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," *arXiv preprint arXiv:1704.01212*, 2017.

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

[15] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, vol. 1, no. 2, 2017.

[16] Y. Sun and J. Han, "Mining heterogeneous information networks: principles and methodologies," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp. 1–159, 2012.

[17] Y. Sun, Y. Yu, and J. Han, "Ranking-based clustering of heterogeneous information networks with star network schema," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 797–806.

[18] 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.

[19] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild, "Meta path-based collective classification in heterogeneous information networks," in *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2012, pp. 1567–1571.

[20] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1531–1540.

[21] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *KDD '17*. ACM, 2017, pp. 135–144.

[22] T.-y. Fu, W.-C. 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.

[23] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Metagraph2vec: Complex semantic path augmented heterogeneous network embedding," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2018, pp. 196–208.

[24] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *arXiv preprint arXiv:1610.09769*, 2016.

[25] Y. Zhang, Y. Xiong, X. Kong, S. Li, J. Mi, and Y. Zhu, "Deep collective classification in heterogeneous information networks," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2018, pp. 399–408.

[26] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7482–7491.

[27] P. V. Tran, "Learning to make predictions on graphs with autoencoders," in *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2018, pp. 237–245.

[28] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *International Conference on Machine Learning*, 2016, pp. 2071–2080.

[29] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv preprint arXiv:1412.6575*, 2014.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[31] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, vol. 2. IEEE, 2005, pp. 729–734.

[32] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[33] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems*, 2016, pp. 3844–3852.

[34] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "Gaan: Gated attention networks for learning on large and spatiotemporal graphs," *arXiv preprint arXiv:1803.07294*, 2018.

[35] Y. Zhang and Q. Yang, "A survey on multi-task learning," *arXiv preprint arXiv:1707.08114*, 2017.