Fine-Grained Semantics-Aware Heterogeneous Graph Neural Networks

Yubin Wang[1,2], Zhenyu Zhang[1,2], Tingwen Liu[1,2](✉), Hongbo Xu[1], Jingjing Wang[1], and Li Guo[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangyubin,zhangzhenyu1996,liutingwen,hbxu,wangjingjing,guoli}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. Designing a graph neural network for heterogeneous graph which contains different types of nodes and links have attracted increasing attention in recent years. Most existing methods leverage meta-paths to capture the rich semantics in heterogeneous graph. However, in some applications, meta-path fails to capture more subtle semantic differences among different pairs of nodes connected by the same meta-path. In this paper, we propose Fine-grained Semantics-aware Graph Neural Networks (FS-GNN) to learn the node representations by preserving both meta-path level and fine-grained semantics in heterogeneous graph. Specifically, we first use multi-layer graph convolutional networks to capture meta-path level semantics via convolution on edge type-specific weighted adjacent matrices. Then we use the learned meta-path level semantics-aware node representations as guidance to capture the fine-grained semantics via the coarse-to-fine grained attention mechanism. Experimental results semi-supervised node classification show that FS-GNN achieves state-of-the-art performance.

Keywords: Graph neural network · Heterogeneous graph · Fine-grained semantics · Meta-path

1 Introduction

Graph neural networks (GNNs), which can learn from graph-structured data, have been successfully applied in various tasks, such as node classification [6,8], graph classification [10,26], link prediction [14] and recommendation [16]. Most of the existing GNNs perform on homogeneous graphs, where all objects and relations are of the same type. However, real-world data tends to be presented as a heterogeneous graph that contains multiple types of objects and relations.

A heterogeneous graph combines different aspects of information. Figure 1 illustrates a toy example of heterogeneous graph, including three types of objects (author, paper, and conference) and six types of relations (cite/cited,

Z. Huang et al. (Eds.): WISE 2020, LNCS 12342, pp. 71–82, 2020.
https://doi.org/10.1007/978-3-030-62005-9_6

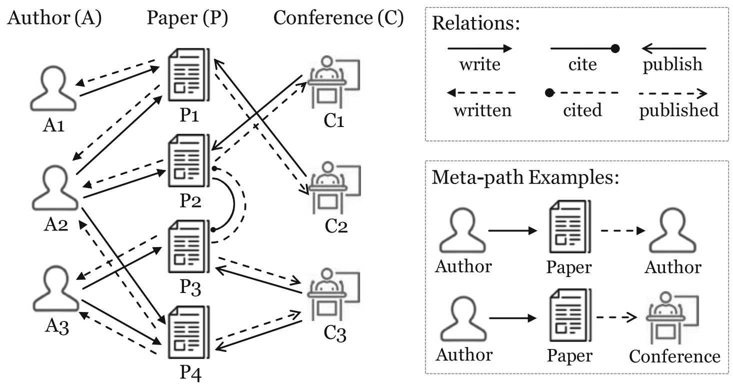

Fig. 1. A toy example of heterogeneous graph.

write/written, and publish/published). Due to the heterogeneity of nodes and edges, a heterogeneous graph contains more comprehensive information and rich semantics. Meta-path [20], a composite relation connecting two nodes, is a widely used structure to capture the semantics. For example, the "Author $\xrightarrow{write}$ Paper $\xrightarrow{written}$ Author" path means authors collaborating on the same papers, while "Author $\xrightarrow{write}$ Paper $\xrightarrow{published}$ Conference" path means authors publishing papers on conferences. Due to the complexity of heterogeneous graph, traditional graph neural networks cannot be directly applied to heterogeneous graph.

Designing a graph neural network for heterogeneous graph have attracted increasing attention in recent years. [24] proposed HAN which transforms a heterogeneous graph into homogeneous graphs by predefined meta-paths and generates nodes representations by fusing the representations learned on each constructed homogeneous graph. This approach requires hand-crafted meta-paths for each problem, but it is hard to exhaustively enumerate and select valuable meta-paths manually. To address this issue, [15] proposed GTN to learn to transform a heterogeneous graph into useful meta-path graphs for each task without any predefined meta-paths and generate node representations via convolution on the learned meta-path graphs.

The key idea of these methods is to identify useful meta-paths to capture the rich semantics in heterogeneous graph. However, in some applications, meta-path fails to capture more subtle semantics. It is because that the information passing by the meta-path, such as features of heterogeneous nodes, is lost in the process of generating meta-path based neighbors. For example, the "Author $\xrightarrow{write}$ Paper $\xrightarrow{written}$ Author" path describes the collaboration relation among authors. However, it cannot depict the fact that *Philip S. Yu* and *Jiawei Han* have many collaborations in data mining field but they seldom collaborate in information retrieval field. This motivates us to design a novel graph neural network to capture the fine-grained semantics under the same meta-path.

In this paper, we propose Fine-grained Semantics-aware Graph Neural Networks (FS-GNN) to learn the node representations preserving both meta-path-level and fine-grained semantics in heterogeneous graph without any predefined meta-paths. Specifically, given the node features as input, we first use the type-specific transformation matrix to project different types of node features into the same space. Then we use a pre-trained meta-path level semantics-aware network to learn the node representations for preserving meta-path level semantics. Afterward, we use the learned node representations to guide the fine-grained semantics-aware network to capture more subtle semantics between pairs of nodes connected by the same relation via the coarse-to-fine grained attention mechanism. It enforces the model to learn the optimal combination of the features of heterogeneous neighbors and lead to better node representations.

We conduct extensive experiments on node classification to evaluate our proposed model. And experimental results on several datasets show that FS-GNN achieves state-of-the-art performance. The source code of this paper can be obtained from https://github.com/jadbin/FS-GNN.

2 Related Work

We briefly mention related work in the field of heterogeneous graph mining, graph embedding, and graph neural networks.

Heterogeneous Graph Mining. In recent years, heterogeneous graph analysis attracts much attention because of rich structural and semantic information in this kind of network [18]. As a general information modeling method, the heterogeneous graph has been widely applied to many data mining tasks such as link prediction [28], classification [1] and recommendation [17]. Meta-path [20] can effectively capture subtle semantics among objects, and many works have exploited meta-path based mining tasks. For example, [27] proposed a meta-path based deep convolutional classification model for collective classification in heterogeneous graph.

Graph Embedding. Graph embedding is proposed to embed graph-structured data into a low dimensional space while preserving the graph structure and property so that the learned embeddings can be applied to the downstream tasks. For example, inspired by word2vec [12,13] proposed DeepWalk which uses SkipGram to learn node embeddings from node sequences generated via random walks over the graph. To address the heterogeneity of graph, [3] introduced meta-path guided random walks and proposed metapath2vec for representation learning in heterogeneous graph. HIN2Vec [4] learns representations of nodes and meta-paths simultaneously via multiple prediction training tasks. Besides these random the random walk based methods, many other approaches have been proposed, such as the deep neural network based methods [22], the matrix factorization based methods [23], etc. Furthermore, attributed graph embedding models [9,11,25] have been proposed to leverages both graph structure and node attributes for learning node embeddings.

Graph Neural Networks. Graph neural networks (GNNs) extend the deep neural network to deal with graph-structured data. Recently, many works generalizing convolutional operation on the graph, and they are often categorized as spectral methods and spatial methods. Spectral methods define graph convolution based on the spectral graph theory. [2] developed convolution operation based graph Fourier transformation. [8] then simplified the previous method by using a linear filter to operate one-hop neighboring nodes. GTN [15] generates node representations via convolution on the learned meta-path graphs. Spatial methods which define convolution directly on the graph, operating on groups of spatially close neighbors. For instance, GraphSAGE [6] performs various aggregators such as mean-pooling over a fixed-size neighborhood of each node. GAT [21] adopts attention mechanisms to learn the relative weights between two connected nodes. HAN [24] leverages node-level attention and semantic-level attention to model the importance of nodes.

Unlike the existing GNNs designed for heterogeneous network which learn the node representations preserving meta-path level semantics, we introduce coarse-to-fine grained attention to capture more subtle semantics difference among the different pairs of nodes connected by the same meta-path. And experiments show that this can further improve the effectiveness of the learned node representations.

3 Preliminaries

A heterogeneous graph, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consists of a node set $\mathcal{V}$ and a link set $\mathcal{E}$. A heterogeneous graph is also associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and a link type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{R}$. $\mathcal{A}$ and $\mathcal{R}$ denote the sets of predefined node types and link types. Figure 1 shows a toy example of heterogeneous graph. It consists of three types of objects (author, paper, and conference) and six types of relations (cite/cited, write/written, and publish/published).

In heterogeneous graph, two nodes can be connected via different semantic paths, which are called meta-paths. A meta-path is defined as a path in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$, which describes a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between objects A_1 and A_{l+1}, where $\circ$ denotes the composition operator on relations. Different meta-paths always reveal different semantics. For example, as shown in Fig. 1, the "Author $\xrightarrow{write}$ Paper $\xrightarrow{written}$ Author" path means authors collaborating on the same papers, while "Author $\xrightarrow{write}$ Paper $\xrightarrow{published}$ Conference" path means authors publishing papers on conferences.

Existing graph neural networks designed for heterogeneous graph [15,24] leverage meta-paths to capture the rich semantics. However, due to the information passing by the meta-path, such as features of heterogeneous nodes, is lost in the process of aggregating features of meta-path based neighbors, these models fail to capture more subtle semantic difference among different pairs of

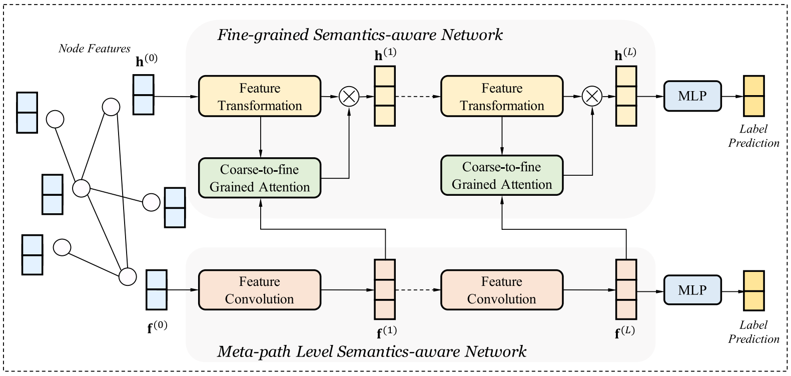

Fig. 2. The overall architecture of FS-GNN.

nodes connected by the same meta-path. In this paper, we propose a graph neural network, namely FS-GNN, to exploit fine-grained semantics under the same meta-path in heterogeneous graph.

4 Proposed Model

In this section, we will give more details of FS-GNN. The overall structure of FS-GNN is shown in Fig. 2. FS-GNN consists of a Meta-path Level Semantics-aware Network (MSN) and a Fine-grained Semantics-aware Network (FSN). We first use MSN to capture meta-path level semantics without any predefined meta-paths. We then use the node representations learned by the MSN as guidance for FSN to capture fine-grained semantics via the coarse-to-fine grained attention. Finally, we present the objectives for model training.

4.1 Meta-path Level Semantics-Aware Network

The goal of MSN is to learn the node representations preserving meta-path level semantics without any predefined meta-paths. Following [15], we view the meta-path as a combination of a list of edges with specific types and assign a type-specific weight to each edge to evaluate the importance of the meta-path.

Specifically, we use a multi-layer graph convolutional network [8] to aggregate the features of neighboring nodes on the weighted adjacent matrices. The layer-wise propagation rule of feature convolution can be defined as follows:

$$\mathbf{f}^{(l+1)} = \sigma\left(\widetilde{D}^{(l)^{-1}} \widetilde{A}^{(l)} \mathbf{f}^{(l)} W_f^{(l)}\right) \tag{1}$$

Here, $\widetilde{A}^{(l)}$ is the layer-wise weighted adjacency matrix as explained in the following paragraph. $\widetilde{D}^{(l)^{-1}}$ is the degree matrix of $\widetilde{A}^{(l)}$, and $\widetilde{D}_{ii}^{(l)} = \sum_j \widetilde{A}_{ij}^{(l)}$. $\sigma(\cdot)$ denotes an activation function such as $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$. $\mathbf{f}^{(l)} \in \mathbb{R}^{|\mathcal{V}| \times d_f^{(l)}}$ denotes the hidden representations of node i in the l^{th} layer.

The heterogeneous graph can be represented as a set of adjacency matrices $\{A_r\}_{r\in\mathcal{R}}$, and A_r is a standard adjacency matrix with the edge type r. Following [15], we include an identity matrix in A as a specific type of edge for self-connections. This trick allows the MSN to learn any length of meta-paths up to l when l layers are stacked. The layer-wise weighted adjacent matrix can be calculated as follows:

$$\widetilde{A}^{(l)} = \sum_{r\in\mathcal{R}} \omega_r^{(l)} A_r \tag{2}$$

where $\omega_r^{(l)}$ is a type-specific weight factor which can be calculated from a learnable parameter $\mathbf{w}_r^{(l)} \in \mathbb{R}^{|\mathcal{R}|}$ as follows:

$$\omega_r^{(l)} = \frac{\exp(\mathbf{w}_r^{(l)})}{\sum_{k\in\mathcal{R}} \exp(\mathbf{w}_k^{(l)})} \tag{3}$$

Due to the heterogeneity of nodes in heterogeneous graph, different types of nodes have different feature spaces. Therefore, initially, we apply a type-specific transformation $M_{\phi(i)} \in \mathbb{R}^{d_f^{(0)} \times d_{in}^{\phi(i)}}$ to project the features of different types of nodes into the same feature space:

$$\mathbf{f}_i^{(0)} = M_{\phi(i)}\mathbf{x}_i \tag{4}$$

where $\mathbf{x}$ can be either attribute features of nodes or one-hot vector for the nodes without attributes.

The node representations $\mathbf{f}^{(l+1)}$ are obtained by aggregating information from the features of their neighborhoods $\mathbf{f}^{(l)}$. After going through L layers of feature convolution, the output representation of node i learned by MSN can be represented as $\mathbf{f}_i = \mathbf{f}_i^{(L)}$. We use the output $\mathbf{f}$ as meta-path level semantics-aware node representations to guide FSN to capture the subtle semantics difference under the same meta-path.

To stabilize the learning process of MSN, we have found extending our method to employ multi-head mechanism to be beneficial, similarly to [21]. Specifically, with K heads executing the procedure of Eq. 1, we concatenate the low-dimensional vectors of different heads and output the representation of each node in the layer $l+1$ as follows:

$$\mathbf{f}_i^{(l+1)} = \mathop{\Big\Vert}_{k=1}^{K} \mathbf{f}_i^{(l+1,k)} \tag{5}$$

where $\Vert$ is the concatenation operator and M denotes the number of heads.

4.2 Fine-Grained Semantics-Aware Network

The goal of FSN is to exploit the fine-grained semantics in heterogeneous graph to learn better node representations under the guidance of meta-path level semantics-ware node representations.

FSN consists of several stacked layers. In each layer l, we would like the output $\mathbf{h}_i^{(l+1)}$ to be composed of K independent components, i.e., $\mathbf{h}_i^{(l+1)} = [\mathbf{c}_i^{(l+1,1)}, \mathbf{c}_i^{(l+1,2)}, \cdots, \mathbf{c}_i^{(l+1,K)}]$ where $\mathbf{c}_k \in \mathbb{R}^{\frac{d_h^{(l+1)}}{K}} (1 \leq k \leq K)$. Each component $\mathbf{c}_k$ is for describing the aspect of node i with fine-grained semantics. The key challenge is to identify which neighbors actually affect the k^{th} aspect of node i. To this end, we propose the coarse-to-fine grained attention mechanism which is presented as follows.

In each layer, similar to MSN, we first apply a type-specific linear transformation $W_c^{(l,k)} \in \mathbb{R}^{\frac{d_h^{(l+1)}}{K} \times \frac{d_h^{(l)}}{K}}$ to each node i:

$$\widehat{\mathbf{c}}_i^{(l,k)} = W_c^{(l,k)} \mathbf{c}_i^{(l,k)} \tag{6}$$

where $\mathbf{h}_i^{(l)} \in \mathbb{R}^{d_h^{(l)}}$ denotes the hidden representations of node i in the l^{th} layer.

We use the meta-path level semantics-aware node representations learned from MSN to guide FSN to capture the fine-grained semantics via the coarse-to-fine grained attention. The attention scores between connected nodes can be calculated as follows:

$$a_{ij}^{(l,k)} = \mathbf{v}^{(l)^\top} \tanh(W_1^{(l)} \mathbf{f}_i^{(l+1)} + W_2^{(l)} \mathbf{f}_j^{(l+1)} + W_3^{(l)} \widehat{\mathbf{c}}_i^{(l,k)} + W_4^{(l)} \widehat{\mathbf{c}}_j^{(l,k)}) \tag{7}$$

Here $\mathbf{v}^{(l)} \in \mathbb{R}^{d_h^{(l)}}$ is a learnable attention vector. $W_1^{(l)} \in \mathbb{R}^{d_h^{(l+1)} \times d_f^{(l+1)}}$, $W_2^{(l)} \in \mathbb{R}^{d_h^{(l+1)} \times d_f^{(l+1)}}$, $W_3^{(l)} \in \mathbb{R}^{d_h^{(l+1)} \times \frac{d_h^{(l+1)}}{K}}$ and $W_4^{(l)} \in \mathbb{R}^{d_h^{(l+1)} \times \frac{d_h^{(l+1)}}{K}}$ are trainable transformation matrices. $\cdot^\top$ represents transposition. Then we can obtain the attention weights by normalizing the attention scores with the softmax function:

$$\alpha_{ij}^{(l)} = \frac{\exp(\omega_{\psi_{i,j}}^{(l)} a_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(\omega_{\psi_{i,k}}^{(l)} a_{ik})} \tag{8}$$

where $\mathcal{N}_i$ is the neighborhood of node i in the graph. $\omega_S^{(l)}$ is a edge type-specific weight factor which can be calculated from a learnable parameter $\mathbf{w}_S^{(l)} \in \mathbb{R}^{|\mathcal{R}|}$:

$$\omega_r^{(l)} = \frac{\exp(\mathbf{w}_r^{(l)})}{\sum_{k \in \mathcal{R}} \exp(\mathbf{w}_k^{(l)})} \tag{9}$$

The representation of node i in aspect k can be aggregated by the projected features with the corresponding coefficients as follows:

$$\mathbf{z}_i^{(l+1,k)} = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \widehat{\mathbf{c}}_j^{(l)} \right) \tag{10}$$

Finally, we concatenate the low-dimensional vectors of different aspects and output the representation of each node in the layer $l+1$ as follows:

$$\mathbf{h}_i^{(l+1)} = \mathop{\Big\|}_{k=1}^{K} \mathbf{z}_i^{(l+1,k)} \tag{11}$$

Similar to MSN, initially, we apply a type-specific transformation $M_{\phi(i)} \in \mathbb{R}^{d_h^{(0)} \times d_{in}^{\phi(i)}}$ to project the features of different types of nodes into the same feature space as follows:

$$\mathbf{h}_i^{(0)} = M_{\phi(i)}\mathbf{x}_i \tag{12}$$

The node representations $\mathbf{h}^{(l+1)}$ are obtained by aggregating information from the features of their neighborhoods $\mathbf{h}^{(l)}$. After going through L attention layers, the output representation of node i learned by FSN can be represented as $\mathbf{h}_i = \mathbf{h}_i^{(L)}$. And $\mathbf{h}_i$ can be transferred to downstream tasks such as node classification.

4.3 Model Training

We train MSN and FSN separately and here we give the objective functions. To optimize the representations toward semi-supervised node classification task, we integrate the node representations learned by the MSN or FSN into a classifier implemented with a one-layer MLP with softmax function to predict the labels of nodes as follows:

$$\widehat{\mathbf{y}}_i = \text{softmax}(\text{MLP}_\theta(\mathbf{o}_i)) \tag{13}$$

where θ is the trainable parameters of the classifier, $\mathbf{o}_i$ is output representation of node i from MSN or FSN, i.e., $\mathbf{o}_i = \mathbf{f}_i$ or $\mathbf{o}_i = \mathbf{h}_i$. Then we can minimize the cross-entropy loss over all labeled nodes between the ground-truth and the prediction:

$$\mathcal{L} = -\frac{1}{|\mathcal{V}_L|}\sum_{i \in \mathcal{V}_L}\sum_{j=1}^{C}\mathbf{y}_{ij} \cdot \log \widehat{\mathbf{y}}_{ij}^T \tag{14}$$

where $\mathcal{V}_L$ is the set of labeled nodes and C denotes the number of classes.

5 Experiments

This section first introduces datasets and experimental settings, and then presents performance comparison results with baselines in order to validate the effectiveness of FS-GNN.

5.1 Datasets and Baselines

We follow existing studies [15, 24] and use three heterogeneous graph benchmark datasets for evaluation, including two citation network datasets ACM, DBLP and one movie dataset IMDB. The statistics of datasets are summarized in Table 1.

ACM contains three types of nodes (paper (P), author (A), subject (S)) and four types of edges (PA, AP, PS, SP). The papers are labeled according to the conference they published and divided into three classes (*Database*, *Wireless Communication*, *Data Mining*). Paper features correspond to elements of a bag-of-words represented of keywords.

Table 1. The statistics of datasets.

Dataset	ACM	DBLP	IMDB
# Nodes	8,994	18,405	12,772
# Edges	25,922	67,946	37,288
# Edge Type	4	4	4
# Features	1,902	334	1,256
# Classes	3	4	3
# Training	600	800	300
# Validation	300	400	300
# Test	2,125	2,857	2,339

DBLP contains three types of nodes (paper (P), author (A), conference (C)) and four types of edges (PA, AP, PC, CP). The authors are labeled research area according to the conferences they submitted, and they are divided into four areas (*Database*, *Data Mining*, *Machine Learning*, *Information Retrieval*). Author features are the elements of a bag-of-words represented of keywords.

IMDB contains three types of nodes (movie (M), actor (A), director (D)) and four types of edges (MA, AM, MD, DM). The movies are divided into three classes (*Action*, *Comedy*, *Drama*) according to their genre. Movie features correspond to elements of a bag-of-words represented of plots.

To evaluate the performance of our proposed FS-GNN, we compare against several state-of-the-art baselines as specified in [15], including the graph embedding methods and graph neural network based methods.

DeepWalk [13] A graph embedding method that learns node embeddings from the node sequences generated via random walks.

metapath2vec [3] A heterogeneous graph embedding method which performs meta-path based random walk and utilizes skip-gram with negative sampling technique to generate node embeddings.

GCN [8] A graph convolutional network which utilizes a localized first-order approximation of the spectral graph convolution.

GAT [21] A graph neural network which uses attention mechanism to model the differences between the node and its one-hop neighbors.

HAN [24] A graph neural network that exploits manually selected meta-paths and leverages node-level attention and semantic-level attention to model the importance of nodes and meta-paths respectively.

GTN [15] A graph neural network transforms a heterogeneous graph into multiple new graphs defined by meta-paths and generates node representations via convolution on the learned meta-path graphs.

5.2 Experimental Setup

In our experiments, we conduct the semi-supervised node classification task to evaluate the performance of our proposed model. The partition of datasets is the same as the previous studies [15,24].

We train a two-layer FS-GNN for IMDB, and a three-layer FS-GNN for ACM and DBLP. We initialized parameters $\mathbf{w}^{(l)}$ in each layer of MSN and FSN with a constant value. We randomly initialize other parameters following [5]. We adopt the Adam optimizer [7] for parameter optimization with weight decay as 0.0005. We set the learning rate as 0.005 for DBLP and 0.001 for ACM and IMDB. We apply dropout [19] with $p = 0.5$ to both layers' inputs, as well as to the normalized attention coefficients. For a fair comparison, we set the output dimension to 64 following [15,24]. We set the number of heads M in MSN as 8, and this means the output dimension of each head is 8. The number of different aspects of each node in FSN is set as 8.

5.3 Node Classification Results

The results of the semi-supervised node classification task are summarized in Table 2, where the best results are highlighted in bold. Following [15], we use the Macro-F1 metric for quantitative evaluation. We present the mean F1 score over 10 runs of our method and reuse the results already reported in [15] for baselines. FS-GNN-M denotes that we only use the output of MSN to predict the labels of nodes.

We can observe that: (1) Our FS-GNN outperforms all the baselines and achieve state-of-the-art results. The performance gain is from two folds. First, the guidance from MSN enforces the model to capture the meta-path level semantics. Second, the model successfully learns to capture fine-grained semantics under the same meta-path via the coarse-to-fine grained attention. (2) FS-GNN-M consistently outperforms GCN and GAT which are designed for homogeneous graph.

Table 2. Results of node classification in terms of F1 score (%). Bold marks highest number among all models. * marks statistically significant improvements over GTN with $p < 0.01$ under a student t-test.

Method	ACM	DBLP	IMDB
DeepWalk [13]	67.42	63.18	32.08
metapath2vec [3]	87.61	85.53	35.21
GCN [8]	91.60	87.30	56.89
GAT [21]	92.33	93.71	58.14
HAN [24]	90.96	92.83	56.77
GTN [15]	92.68	94.18	60.92
FS-GNN-M (ours)	92.46	93.81	60.54
FS-GNN (ours)	**93.57***	**94.72***	**63.20***

This proves that capturing rich semantics in heterogeneous graph leads to effective node representations to achieve better performance. (3) FS-GNN-M achieves competitive results compared with GTN. It demonstrates that the multi-layer convolution on type-specific weighted adjacent matrices is capable of learning the importance of meta-path.

6 Conclusion

In this paper, we propose FS-GNN to learn the node representations of a heterogeneous network without any predefined meta-paths. It is able to learn meta-path level semantics-aware node representations via multi-layer convolution on type-specific weighted adjacent matrices. And it learns fine-grained semantics-aware node representations under the guidance of the meta-path level semantics-aware node representations via the coarse-to-fine grained attention mechanism. Experimental results demonstrate that our FS-GNN achieves state-of-the-art performance on several semi-supervised node classification benchmarks. The future directions include studying the efficacy of coarse-to-fine grained attention layers combined with other GNNs such as GTN [15]. Also, we will try to extend FS-GNN and apply it to other tasks such as graph classification [10,26].

Acknowledgment. This work is supported by the National Key Research and Development Program of China (grant No. 2016YFB0801003) and the Strategic Priority Research Program of Chinese Academy of Sciences (grant No. XDC02040400).

References

1. Bangcharoensap, P., Murata, T., Kobayashi, H., Shimizu, N.: Transductive classification on heterogeneous information networks with edge betweenness-based normalization. In: WSDM, pp. 437–446 (2016)
2. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. In: ICLR (2013)
3. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: scalable representation learning for heterogeneous networks. In: KDD, pp. 135–144 (2017)
4. Fu, T.Y., Lee, W.C., Lei, Z.: HIN2Vec: explore meta-paths in heterogeneous information networks for representation learning. In: CIKM, pp. 1797–1806 (2017)
5. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feed for leveraging graph wavelet transform to address the short-comings of previous spectral graphrd neural networks. In: AISTATS, pp. 249–256 (2010)
6. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NIPS, pp. 1024–1034 (2017)
7. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
9. Li, J., Dani, H., Hu, X., Tang, J., Chang, Y., Liu, H.: Attributed network embedding for learning in a dynamic environment. In: CIKM, pp. 387–396 (2017)

10. Li, R., Wang, S., Zhu, F., Huang, J.: Adaptive graph convolutional neural networks. In: AAAI (2018)
11. Liao, L., He, X., Zhang, H., Chua, T.S.: Attributed social network embedding. TKDE **30**(12), 2257–2270 (2018)
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
13. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: KDD, pp. 701–710 (2014)
14. Schütt, K., Kindermans, P.J., Felix, H.E.S., Chmiela, S., Tkatchenko, A., Müller, K.R.: SchNet: a continuous-filter convolutional neural network for modeling quantum interactions. In: NIPS, pp. 991–1001 (2017)
15. Seongjun, Y., Jeong, M., Kim, R., Kang, J., Kim, H.: Graph transformer networks. In: NIPS, pp. 11960–11970 (2019)
16. Shi, C., et al.: Deep collaborative filtering with multi-aspect information in heterogeneous networks. TKDE (2019)
17. Shi, C., Hu, B., Zhao, W.X., Philip, S.Y.: Heterogeneous information network embedding for recommendation. TKDE **31**(2), 357–370 (2018)
18. Shi, C., Li, Y., Zhang, J., Sun, Y., Philip, S.Y.: A survey of heterogeneous information network analysis. TKDE **29**(1), 17–37 (2016)
19. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. JMLR **15**(1), 1929–1958 (2014)
20. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: PathSim: meta path-based top-k similarity search in heterogeneous information networks. VLDB **4**(11), 992–1003 (2011)
21. Veličković P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
22. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: KDD, pp. 1225–1234 (2016)
23. Wang, X., Cui, P., Wang, J., Pei, J., Zhu, W., Yang, S.: Community preserving network embedding. In: AAAI (2017)
24. Wang, X., et al.: Heterogeneous graph attention network. In: WWW, pp. 2022–2032 (2019)
25. Zhang, C., Swami, A., Chawla, N.V.: SHNE: representation learning for semantic-associated heterogeneous networks. In: WSDM, pp. 690–698 (2019)
26. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: AAAI (2018)
27. Zhang, Y., Xiong, Y., Kong, X., Li, S., Mi, J., Zhu, Y.: Deep collective classification in heterogeneous information networks. In: WWW, pp. 399–408 (2018)
28. Zhang, Y., Tang, J., Yang, Z., Pei, J., Yu, P.S.: COSNET: connecting heterogeneous social networks with local and global consistency. In: KDD, pp. 1485–1494 (2015)