Knowledge-Based Systems 235 (2022) 107611

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

MEGNN: Meta-path extracted graph neural network for heterogeneous graph representation learning

Yaomin Chang [a,b], Chuan Chen [a,b,*], Weibo Hu [a,b], Zibin Zheng [a,b], Xiaocong Zhou [a], Shouzhi Chen [c]

[a] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China
[b] National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou, China
[c] Tencent Inc., Shenzhen, China

ARTICLE INFO

Article history:
Received 3 December 2020
Received in revised form 14 October 2021
Accepted 15 October 2021
Available online 21 October 2021

Keywords:
Heterogeneous graph
Graph neural networks
Representation learning
Meta-paths

ABSTRACT

Heterogeneous graphs with multiple types of nodes and edges are ubiquitous in the real world and possess immense value in many graph-based downstream applications. However, the heterogeneity within nodes and edges in heterogeneous graphs has brought pressing challenges for practical node representation learning. Existing works manually define multiple meta-paths to model the semantic relationships in heterogeneous graphs. Such strategies heavily rely on the quality of domain knowledge and require extensive hand-crafted works. In this paper, we propose a novel Meta-path Extracted heterogeneous Graph Neural Network (MEGNN) that is capable of extracting meaningful meta-paths in heterogeneous graphs, providing insights about data and explainable conclusions to the model's effectiveness. Concretely, MEGNN leverages heterogeneous convolution to combine different bipartite sub-graphs corresponding to edge types into a new trainable graph structure. By adopting the message passing paradigm of GNNs through trainable convolved graphs, MEGNN can optimize and extract effective meta-paths for heterogeneous graph representation learning. To enhance the robustness of MEGNN, we leverage multiple channels to yield various graph structures and devise a channel consistency regularizer to enforce the node embeddings learned from different channels to be similar. Extensive experimental results on three datasets not only show the effectiveness of MEGNN compared with the state-of-the-art methods, but also demonstrate the favorable interpretability of the extracted meta-paths.

1. Introduction

Many real-world data are intrinsically represented by graph structures, such as social networks, citation networks, e-commerce systems, and so on, where objects and relationships are represented by nodes and edges. Analyzing and mining knowledge in graph data has been an emerging topic in both academia and industry. However, the complex non-Euclidean and semantics-related graph structures lead to challenges in this field. For example, there is no fixed order and size for node neighbors and the nodes and edges may be related to different types of semantic information in specific scenarios. To address the challenges in graph mining, some representative methods [1,2] propose to leverage some pre-defined metrics, e.g., the Morgan index, to encode the complex graph structures into embeddings.

With the development of the technique of deep learning, graph embedding, which aims to adaptively learn a low-dimensional embedding vector for each node in a graph, has been shown to be effective for various downstream graph-based tasks such as node classification [3,4], link prediction [5,6] and community detection [7,8]. Recently, Graph Neural Networks (GNNs), the newly emerging graph embedding models, have presented powerful expressiveness and achieved state-of-the-art performance on the graph-based tasks. The representative methods include Graph Convolutional Network (GCN) [9] and Graph Attention Network (GAT) [10] and their variants [11–15]. All these GNN-based methods can effectively capture attribute similarity and preserve structure information with the specific message propagation.

Despite the promising performance, these methods can only deal with homogeneous graphs, i.e., the node type and edge type are unique. In the real world, the graph data often contains multiple node types and edge types. Furthermore, each type of nodes

* Corresponding author at: School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China.
E-mail addresses: changym3@mail2.sysu.edu.cn (Y. Chang), chenchuan@mail.sysu.edu.cn (C. Chen), huwb7@mail2.sysu.edu.cn (W. Hu), zhzibin@mail.sysu.edu.cn (Z. Zheng), isszxc@mail.sysu.edu.cn (X. Zhou), easychen@tencent.com (S. Chen).

https://doi.org/10.1016/j.knosys.2021.107611
0950-7051/

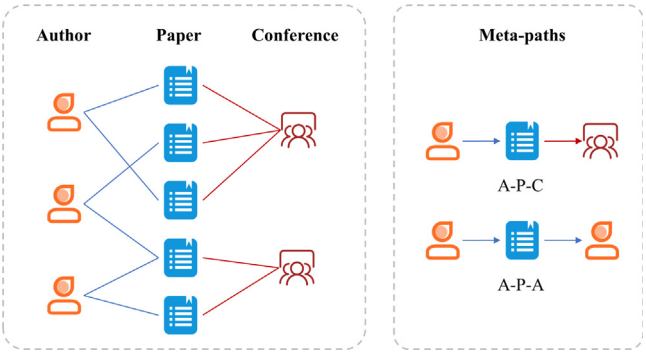


Fig. 1. An example of citation graph and the illustration of meta-paths of length two started from *Author* nodes.

may be associated with attributes in different feature spaces. Such graphs with heterogeneity can also be called heterogeneous graphs or Heterogeneous Information Networks (HINs) [16]. For instance, a citation graph in Fig. 1 contains three types of nodes: *Author*, *Paper*, and *Conference*; two types of edges: *Author-Paper* (*A-P*), and *Paper-Conference* (*P-C*). The attributes of *Author* nodes may involve affiliations, while the attributes of the *Paper* nodes are the bag-of-words vectors of their abstract and keywords.

To tackle the heterogeneity of graphs, meta-path [17], a composite relation scheme, is widely adopted in various heterogeneous graph embedding methods [17–21]. The meta-path can be viewed as extending the idea of the Morgan index [1,2] to heterogeneous graphs to reveal relational relevance by pre-defined knowledge. Fig. 1 illustrates two meta-paths started from *Author* nodes in the citation graph, in which *Author-Paper-Author* (*A-P-A*) is a meta-path that indicates the co-author relationships and *Author-Paper-Conference* (*A-P-C*) is an asymmetric meta-path that reveals publishing relations between *Author* and *Conference*. With the aid of meta-paths, high-order neighbors are directly connected and the high-order proximity between nodes is preserved in heterogeneous graphs.

Although the meta-path based methods have achieved some success in heterogeneous graph embedding, they still suffer from two severe problems. First, some conventional graph embedding methods [18,19] overlook the node attributes due to the limits of the model capacity and expressiveness, consequently performing poorly in heterogeneous graphs with rich node content attributes. Second, some GNN-based methods [20,21] with the ability to incorporate node attributes only utilize limited pre-defined meta-paths, thus the model's effectiveness is significantly affected by the choice of meta-paths. Moreover, these methods usually require extra domain knowledge since meta-paths are diverse in particular scenarios. It is usually exhausting to enumerate all possibilities and examine their importance in advance to obtain the optimized meta-paths. Hence, existing methods relying on a set of manually specified meta-paths cannot fully exploit the semantic relationships in heterogeneous graphs.

To address the aforementioned problems, we propose a novel method called Meta-path Extracted heterogeneous Graph Neural Network (MEGNN), which adopts the idea of the message passing paradigm of GNNs to simultaneously encode graph topological structure, attribute information and semantic relationships into node embeddings. As a result, MEGNN can automatically extract effective meta-paths during the message passing blue over trainable graph structures, which provides explainable conclusions and favorable interpretability for the model's effectiveness. Besides, since there may exist some challenges in the joint optimization of the trainable graph structures and the message passing paradigm, we extend the heterogeneous convolution and message passing into multiple channels with the consistency regularization. In this way, the performance and generalization of the proposed methods can be further enhanced.

Specifically, we firstly apply multiple sophisticatedly designed heterogeneous convolution modules to the raw heterogeneous graph after projecting the attributes of nodes of various types into the same feature space by type-specific transformations (or preprocessing techniques). In this stage, all types of relations in the raw heterogeneous graph are assigned with different trainable weights and combined into the convolved graph. The trainable weights redirect the flow of message aggregation, indicating the relevance between connected nodes by the importance of their relations. Then we perform the message passing on the convolved graphs across layers to generate the node embeddings for the initial heterogeneous graph and introduce channel consistency regularization to enhance the robustness and stability of the model. Owing to the proposed heterogeneous convolution, the node features are guided to flow through the trainable paths, thus valuable meta-paths can be discovered and extracted during the optimization of MEGNN. In this way, explainable insights about the characteristics of graphs in specific scenarios can also be excavated.

In summary, our work makes the following major contributions:

(1) We propose a novel heterogeneous graph neural network named MEGNN that can learn effective node representations. Meanwhile, the model is capable of discovering and extracting the most expressive meta-paths, thereby providing explainable insights about specific data.
(2) We introduce the heterogeneous convolution module based on the intrinsic properties of meta-path schemes to generate trainable heterogeneous graph structures. Besides, we introduce the multi-channel mechanism and devise the channel consistency regularizer to further enhance the performance and stability of the model.
(3) We conduct a suite of comprehensive experiments to evaluate the effectiveness of the proposed method. The results not only show the superiority of MEGNN compared with the state-of-the-art methods but also verify the interpretability of meta-paths extracted by MEGNN.

2. Related work

2.1. Plain graph embedding

Plain graph embedding methods are performed in homogeneous graphs, i.e., with a single type of nodes and edges. Some representative works include Deepwalk [22] and node2vec [23], which adopt truncated random walk on graphs to generate the corpus sequences followed by a skip-gram model [24] to train the embeddings. LINE [25] proposes to learn the node representations by preserving the first-order and second-order proximity. Apart from these random walk based methods, there are some other studies based on deep neural networks like SDNE [26] and based on matrix factorization such as GraRep [27], NetMF [28] and NetSMF [29]. However, all of these methods only utilize the topological structure, ignoring the attribute information and semantic relationships.

2.2. Attributed graph embedding

As many graphs in reality contain not only topological structures but also node attributes, a wide variety of models have been proposed for attributed graphs. ANRL [30] proposes a neighbor enhancement autoencoder to jointly integrate graph structures and node attribute information into node representations.

DANE [31] enforces the representations learned from structures and attributes to be consistent and complementary by a carefully designed loss. GCN [9] and GraphSAGE [12] propose to aggregate the neighbor attributes layerwise while exploring the topological structure implicitly. GAT [10] introduces the graph attention mechanism to measure the different contributions of the neighbors in aggregation. These methods have achieved considerable improvements in various tasks in homogeneous graphs. However, they fail to encode the heterogeneity into representations when applied in heterogeneous graphs.

2.3. Heterogeneous graph embedding

Heterogeneous graphs consist of multiple types of nodes and edges, which render it difficult to preserve both the structure information and the pair-specific relational knowledge into node embeddings appropriately. Considering that meta-paths contain estimable prior knowledge, Metapath2vec [18] designs a meta-path based random walk and utilizes the skip-gram model to perform heterogeneous graph embedding. HERec [19] first transforms the heterogeneous graph into multiple homogeneous graphs based on different meta-paths and finally fuses the representations learned from these graphs. There are methods that also leverage co-occurrence information [32] and apply transformations into a unified space [33]. However, these methods have to discard node attributes limited by model capacity.

Owing to the success of Graph Neural Networks (GNNs) to effectively learn representations for graph-structured data, a variety of methods based on GNNs have been extended to model heterogeneous graphs and encode the attribute information into representations. GATNE [34] focuses on the multiplexity of heterogeneous graphs and proposes to learn base embeddings, edge embeddings and attribute embeddings to generate the ultimate node embeddings. RGCN [35] adapts GCN [9] to transform the node into multiple feature spaces for each relation. HAN [20] introduces the attention mechanism in the fusion of the embeddings learned from meta-path based neighbors. MAGNN [21] proposes to intra-aggregate the nodes along the meta-path beyond the aggregation of various meta-path based neighbors. More recently, GTN [36] proposes a novel layer to generate and select important local structures, whereas the layer proposed requires high time and space complexity, which limits its applications.

3. Preliminary

3.1. Attributed heterogeneous graph

A heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$ with a node type mapping function $f_v : \mathcal{V} \rightarrow \mathcal{T}^v$ and an edge type mapping function $f_e : \mathcal{E} \rightarrow \mathcal{T}^e$. $\mathcal{T}^v$ and $\mathcal{T}^e$ denote the pre-defined sets of node types and edge types, respectively, with $|\mathcal{T}^v| + |\mathcal{T}^e| > 2$.

To represent $\mathcal{G}$ more concretely, the heterogeneous graph $\mathcal{G}$ can be decoupled into multiple homogeneous and bipartite subgraphs that only contain two types of nodes and a single type of edge. Let N be the number of nodes including all types of nodes, i.e., $N = |\mathcal{V}|$. The heterogeneous graph can be represented as a set of sparse adjacency matrices $\{\mathbf{A}_e | e \in \mathcal{T}^e\}$, where $\mathbf{A}_e \in \{0, 1\}^{N \times N}$ is the adjacency matrix corresponds to edge type e. Note that each adjacency matrix is extended to include all types of nodes in the graph.

An attributed heterogeneous graph is a heterogeneous graph endowed with an attribute representation for each node, i.e., $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathbf{X} \in \mathbb{R}^{N \times M}$ is the attribute matrix that encodes M attributes for each node.

3.2. Meta-path scheme

A meta-path scheme $\mathcal{P}$ of length l is a path defined in the heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and is denoted as the form of $\mathcal{T}_1 \xrightarrow{R_1} \mathcal{T}_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} \mathcal{T}_{l+1}$ (abbreviated as $\mathcal{P} = \mathcal{T}_1\mathcal{T}_2 \cdots \mathcal{T}_{l+1}$), where $R_l \in \mathcal{T}^e$ denotes the edge types. It describes a composite relation $R = R_1 \circ R_2 \cdots \circ R_l$ between nodes of type $\mathcal{T}_1$ and $\mathcal{T}_{l+1}$, where $\circ$ is the composition operator on relations. For simplicity, we also refer to $\mathcal{P}$ as the composite relation R, i.e., $\mathcal{P} = R$.

3.3. Meta-path based neighbors

Given meta-path $\mathcal{P} = \mathcal{T}_1\mathcal{T}_2 \cdots \mathcal{T}_{l+1}$, some nodes of type $\mathcal{T}_{l+1}$ are connected to the nodes of type $\mathcal{T}_1$ by this meta-path. These nodes are referred to as meta-path based neighbors to each other.

3.4. Meta-path based graph

A meta-path based graph directly represents the connectivity of meta-path neighbors. Given a meta-path $\mathcal{P}$ describing the composite relation $R = R_1 \circ R_2 \cdots \circ R_l$, the adjacency matrix of the meta-path based graph is generated by the multiplication of adjacency matrices of all different relations as

$$\mathbf{A}_{\mathcal{P}} = \mathbf{A}_{R_1}\mathbf{A}_{R_2} \cdots \mathbf{A}_{R_l}, \tag{1}$$

where $\mathbf{A}_{\mathcal{P}}, \mathbf{A}_{R_1}, \ldots, \mathbf{A}_{R_l} \in \{0, 1\}^{N \times N}$.

4. Methodology

In this section, we elaborate on the details of the proposed Meta-path Extracted heterogeneous Graph Neural Network (MEGNN). The framework of MEGNN is illustrated in Fig. 2. The original heterogeneous graph is transformed by multiple independent heterogeneous convolutions to generate various trainable graph structures in each layer. With the message passing along the layers in each channel, meaningful meta-paths are extracted and effective node representations are obtained. To strengthen the stability of the model and the extracted meta-path schemes, heterogeneous convolutions and message passing are parallelly conducted in separate channels and the embeddings learned from different channels are regularized to be similar in the latent space.

4.1. Meta-path generation by GNN

Some methods [18,20,21] leverage the meta-path based graphs to learn node representations for heterogeneous graphs and achieve competitive performance. However, given a meta-path of length l, the multiplications of adjacency matrices of the bipartite sub-graphs are performed l times to obtain the meta-path based graph. It is usually exhausting and time-consuming to enumerate a variety of meta-paths of varying lengths and compute their meta-path based graph.

As the meta-path based graphs benefit the model by explicitly capturing the information of high-order neighbors that imply meta-path semantics, we found it is similar to the paradigm of GNNs [37] which implicitly aggregates messages from high-order neighbors by stacking layers. Hence, we take full advantage of the strength of GNNs and meta-path semantics and propose to generate diverse meta-paths by exploiting the message passing paradigm of GNNs.

First, we directly apply GNNs to heterogeneous graphs without considering heterogeneity. Formally the framework of general GNNs can be formulated as follows:

$$\mathbf{M}^{(l)} = agg^{(l)}(\mathbf{H}^{(l-1)}; \mathbf{A}^{(l)}), \tag{2}$$

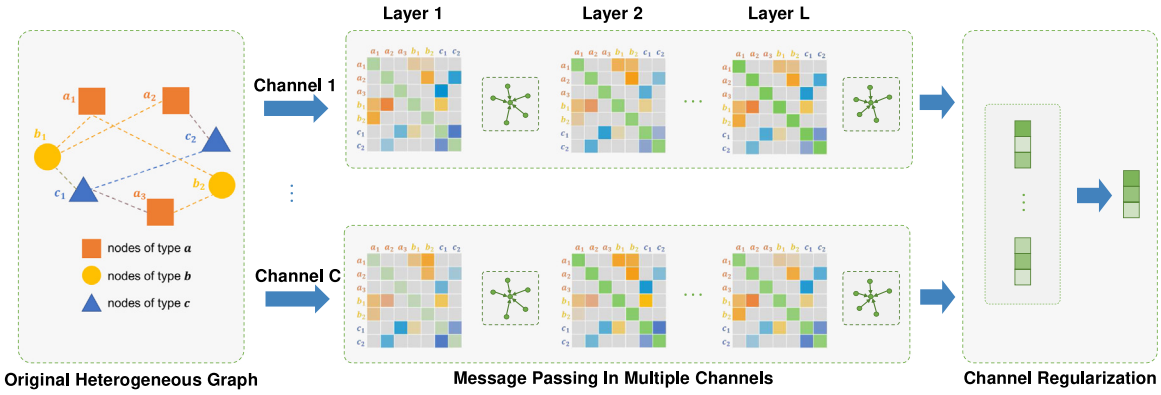


Fig. 2. The overall framework of MEGNN. The left part depicts the topology of the original heterogeneous graph and the shapes of nodes indicate the node types. In the middle of the figure, the convolved heterogeneous graphs are represented by the adjacency matrices, where the types of edges are distinguished by colors and the shade of colors indicates the learned weights of edge types. The right part describes the combination of node embeddings learned from multiple channels.

$$\mathbf{H}^{(l)} = update^{(l)}(\mathbf{H}^{(l-1)}, \mathbf{M}^{(l)}), \tag{3}$$

where the aggregation function $agg^{(l)}(\cdot)$ is defined in the lth layer. It computes the messages from the representations in the previous layer and aggregates them based on the adjacency matrix $\mathbf{A}^{(l)}$. Specifically, $\mathbf{A}^{(1)} = \mathbf{A}^{(2)} = \cdots = \mathbf{A}^{(L)}$ always satisfies on a vanilla graph. The $update^{(l)}(\cdot)$ is a function to combine the representation of the node itself in the previous layer and the result of neighborhood aggregation computed by $agg^{(l)}(\cdot)$. The hidden feature of nodes in the lth layer of the neural network is denoted as $\mathbf{H}^{(l)} \in \mathbb{R}^{N \times d^{(l)}}$, where N is the number of nodes in the graph and $d^{(l)}$ is the dimension of the features in lth layer. The initial features in 0-th layers are usually taken as the node attributes, i.e.,

$$\mathbf{H}^{(0)} = \mathbf{X}. \tag{4}$$

In particular, GCN [9] is a typical and successful GNN model. For simplicity, we take GCN as an example to demonstrate the relevance between GNNs and meta-path based connectivity. In essence, GCN can be derived from Eqs. (2) and (3) when $agg(\cdot)$ and $update(\cdot)$ are set as

$$agg^{(l)}(\mathbf{H}^{(l-1)}; \mathbf{A}^{(l)}) = \hat{\mathbf{A}}^{(l)}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)}, \tag{5}$$

$$update^{(l)}(\mathbf{H}^{(l-1)}, \mathbf{M}^{(l)}) = \sigma(\mathbf{M}^{(l)}), \tag{6}$$

where σ is the non-linear activation function $ReLu(\cdot)$; $\hat{\mathbf{A}}^{(l)} = \tilde{\mathbf{D}}^{(l)^{-\frac{1}{2}}}\tilde{\mathbf{A}}^{(l)}\tilde{\mathbf{D}}^{(l)^{-\frac{1}{2}}}$ and $\mathbf{W}^{(l)}$ denotes the symmetric normalized adjacency matrix and the linear transformation in the lth layer, respectively. Here $\tilde{\mathbf{A}}^{(l)} = \mathbf{A}^{(l)} + \mathbf{I}$ and $\tilde{\mathbf{D}}^{(l)}$ is the degree matrix of $\tilde{\mathbf{A}}^{(l)}$. By stacking L layers, the node representations learned from GCN can be written as

$$\mathbf{H}^{(L)} = \sigma(\hat{\mathbf{A}}^{(L)}\sigma(\hat{\mathbf{A}}^{(L-1)} \cdots \sigma(\hat{\mathbf{A}}^{(1)}\mathbf{X}\mathbf{W}^{(1)})\mathbf{W}^{(L-1)})\mathbf{W}^{(L)}). \tag{7}$$

Intuitively, the message aggregation paradigm of GNNs implies the consecutive multiplication of the adjacency matrices. To be more explanatory, we remove the nonlinear activation function, in other words, let $\sigma(x) = x$, hence the multiple weight matrices across consecutive layers could be collapsed into a single linear transformation. We formulate this overall linear transformation into the message function in the first layer for clarity. Thus the ultimate representations of the nodes can be rewritten as

$$\mathbf{H}^{(L)} = \hat{\mathbf{A}}^{(L)}\hat{\mathbf{A}}^{(L-1)} \cdots \hat{\mathbf{A}}^{(1)}\mathbf{X}\mathbf{W}^{(1)}, \tag{8}$$

where $\mathbf{W} \in \mathbb{R}^{M \times d}$, M is the dimension of nodes attributes and d is the output dimension of the weight matrix $\mathbf{W}^{(1)}$.

It is observed that Eq. (8) is equivalent to directly aggregating features from the L-order neighbors connected by some meta-path schemes, with their weights co-determined by the symmetric normalizations of L layers. For GCN, $\hat{\mathbf{A}}^{(1)} = \hat{\mathbf{A}}^{(2)} = \cdots = \hat{\mathbf{A}}^{(L)}$, since $\mathbf{A}^{(1)} = \mathbf{A}^{(2)} = \cdots = \mathbf{A}^{(L)}$ always satisfies and the symmetric normalization is also identical across layers.

In this paper, we propose Meta-path Extracted heterogeneous Graph Neural Network (MEGNN) to generalize Eq. (8) to heterogeneous graphs. Concretely, we propose a novel heterogeneous convolution module $F^{(l)}(\cdot)$ for each layer to generate different meta-path structures, thus effective meta-path schemes can be generated and extracted. The proposed model can be formulated as follows:

$$\mathbf{H}^{(L)} = \mathbf{A}^{(L)}_{conv}\mathbf{A}^{(L-1)}_{conv} \cdots \mathbf{A}^{(1)}_{conv}\mathbf{X}\mathbf{W}^{(1)}, \tag{9}$$

$$\mathbf{A}^{(l)}_{conv} = F^{(l)}(\{\mathbf{A}_e | e \in \mathcal{T}^e\}), \tag{10}$$

where $\mathbf{A}^{(l)}_{conv} \in \mathbb{R}^{N \times N}$ and $\{\mathbf{A}_e | e \in \mathcal{T}^e\}$ denotes the set of bipartite graphs in the heterogeneous graph. The details of heterogeneous convolution module $F^{(l)}(\cdot)$ are stated later in Section 4.2.

With $F^{(l)}(\cdot)$ transforming the heterogeneous graph into a convolved graph that automatically learns the weights of different types of edges, a meta-path of length two can be extracted by the multiplication of two convolved adjacency matrices. It is worth noting that the explicit multiplication of adjacency matrices can be avoided under the message passing framework of GNNs as the calculation of Eq. (9) can be accomplished from right to left. This significantly reduces the whole time complexity from $O(LN^3 + NMd)$ to $O(LN^2d + NMd)$.

4.2. Heterogeneous convolution

The core idea of heterogeneous convolution module $F^{(l)}(\cdot)$ is to generate diverse and trainable convolved graphs to explore different local relational structures, thus the favorable meta-path schemes can be automatically extracted. The conceptual description of heterogeneous convolution is illustrated in Fig. 3

A naive approach for the construction of the convolved graph is to assign each bipartite graph a normalized coefficient and calculate their linear combination, i.e.,

$$\mathbf{A}^{(l)}_{conv} = F^{(l)}(\{\mathbf{A}_e | e \in \mathcal{T}^e\}) = \sum_{e \in \mathcal{T}^e} \alpha_e \mathbf{A}_e, \tag{11}$$

where $\mathcal{T}^e$ denotes the edge type set of the graph and α_e is a layer-wise independent parameter to be learned indicating the contribution of the sub-graph of type e to the convolved structures.

The naive approach in Eq. (11) has two obvious drawbacks. First, the parameters are independently learned in different layers, indicating that the sub-structures of extracted meta-paths

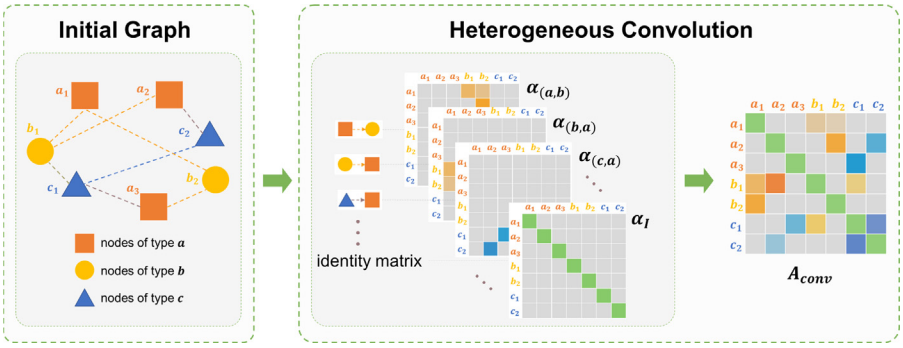


Fig. 3. A conceptual description of heterogeneous convolution.

may not correlate to each other. Still taking an example on a citation graph, we may expect both the structure *A-P* in the 1st layer and *P-C* in the 2nd layer are important at the same time so that the meta-path scheme *A-P-C* can be extracted. Therefore, it is essential to capture the relationship between the convolved graph of adjacent layers. Second, the length of generated meta-path schemes is always equal to the number of GNN layers, thus the *L*-layer model cannot extract relatively short meta-path schemes in heterogeneous graphs. This defect limits the ability of the model to utilize shallow semantic relationships, which may also possess essential dependencies between nodes. Hence, a mechanism to learn variable-length meta-path schemes is required when multiple GNN layers are stacked.

To address the first issue, we introduce the type-wise weight sharing mechanism into our heterogeneous convolution module, which is intuitively shown in Fig. 4. Recalling that the naive approach assigns each edge type a learnable weight, we naturally decompose this edge type weight as the product of the importance of the source node type and the target node type, e.g., the importance of *A-P* can be obtained by multiplying the node importance of *A* and *P*. Note that there are two importance vectors in each layer, $s^{(l)}$ for the source node types and $t^{(l)}$ for the target node types, since *A-P* and *P-A* should have different importance in a directed heterogeneous graph. With two trainable vectors that indicate source and target node type importance respectively, we share the importance vector of source node type in the current layer with the importance vector of target node type in the previous layer. Hence, the local structures learned from consecutive layers could better correlate to each other.

In order to enable GNNs to learn variable-length meta-path schemes, we additionally introduce an identity matrix $\mathbf{I} \in \mathbb{R}^{N\times N}$ into the heterogeneous convolution module and learn an independent parameter as its importance. The essential insight behind adding the identity matrix is to provide the model an alternative to decrease the influence of all types of edges and mainly focus on the node itself. When the importance of the identity matrix increases, the GNN layer performs aggregation mostly on the node itself, thus producing little contribution to the expansion of meta-path schemes.

Given the above statement, this heterogeneous convolution in *l*th layer is collectively formulated as

$$\mathbf{A}_{conv}^{(l)} = \alpha_{\mathbf{I}}^{(l)}\mathbf{I} + \sum_{(e_1,e_2)\in\mathcal{T}^e} \alpha_{(e_1,e_2)}^{(l)}\mathbf{A}_{(e_1,e_2)}, \tag{12}$$

where $\alpha_{\mathbf{I}}^{(l)}$ and $\alpha_{(e_1,e_2)}^{(l)}$ are the normalized coefficients for the identity matrix and the edge type (e_1, e_2), respectively. They can be calculated as follows:

$$\alpha_{(e_1,e_2)}^{(l)} = \frac{\exp(s_{e_1}^{(l)} t_{e_2}^{(l)})}{z^{(l)}}, \tag{13}$$

$$\alpha_{\mathbf{I}}^{(l)} = \frac{\alpha'^{(l)}_{\mathbf{I}}}{z^{(l)}}, \tag{14}$$

$$z^{(l)} = \exp(\alpha'^{(l)}_{\mathbf{I}}) + \sum_{(e_1,e_2)\in\mathcal{T}^e} \exp(s_{e_1}^{(l)} t_{e_2}^{(l)}). \tag{15}$$

Here $\alpha'^{(l)}_{\mathbf{I}}$ denotes the unnormalized weight of the identity matrix in the *l*th layer, $s^{(l)}$ and $t^{(l)}$ denotes the importance vector of source node type and target node type, respectively. The type-wise weight sharing mechanism can be formally written as

$$s^{(l)} = t^{(l-1)} \quad (2 \leqslant l \leqslant L). \tag{16}$$

4.3. The multi-channel mechanism

After stacking *L* heterogeneous convolution layers, a set of meta-paths can be extracted with their contributions calculated by multiplying the edge type importance across layers.

It is noted that the trainable graph structures have to be initialized randomly, which poses challenges to prevent the model from falling into local optimum in the training stage. Hence, we introduce the multi-channel mechanism and devise the channel consistency regularization to further improve the performance of the model and the generalization of the extracted meta-paths.

Concretely, in each channel, *L* GNN layers are stacked thus multiple sets of meta-paths and node representations are obtained. Considering that node representations learned from various meta-path based augmentation should be similar in the same latent space, we propose to restrain the node embeddings learned from different channels to be close during the training phase. With *C* channels employed in the model and *L* GNN layers stacked in each channel, representations of different channels in the *L*th layer are denoted as $\{\mathbf{Z}^{(1)}, \mathbf{Z}^{(2)}, \ldots, \mathbf{Z}^{(C)}\}$. The ultimate representations learned from MEGNN are calculated by average all distributions:

$$\bar{\mathbf{Z}} = \frac{1}{C}\sum_{c=1}^{C} \mathbf{Z}^{(c)}. \tag{17}$$

During training, we introduce an extra channel consistency regularization loss into the model which additionally minimizes the distributional distance between $\bar{\mathbf{Z}}$ and $\mathbf{Z}^{(c)}$ by Frobenius norm, i.e., minimize

$$Dist(\mathbf{Z}^{(c)}, \bar{\mathbf{Z}}) = \|\mathbf{Z}^{(c)} - \bar{\mathbf{Z}}\|_F^2. \tag{18}$$

Then, the channel consistency regularization loss can be defined as the sum of the distances between each $\mathbf{Z}^{(c)}$ and the central representations $\bar{\mathbf{Z}}$ as Eq. (19). Thus we can enforce the model to learn similar representations from various meta-path structures.

$$\mathcal{L}_{con} = \frac{1}{C}\sum_{c=1}^{C} \|\mathbf{Z}^{(c)} - \bar{\mathbf{Z}}\|_F. \tag{19}$$

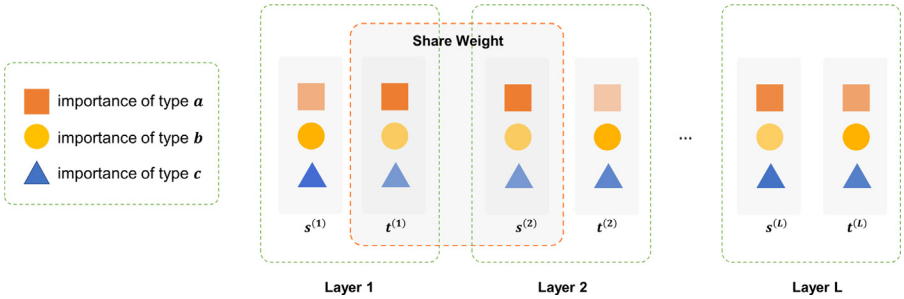


Fig. 4. The general idea of the type-wise weight sharing mechanism. In each layer, there is a source type vector and a target type vector. Different colors represent different node types and the shade of the color indicates the importance of the node type it represents. The source type vector and target type vector are shared between adjacent layers.

From another perspective, the heterogeneous convolution can be viewed as the process of data augmentation on heterogeneous graphs, in which different relational graph structures, i.e., meta-path schemes, are either enhanced or weakened. In each channel, GNNs are mainly trained with the reweighted edges determined by heterogeneous convolution. Therefore, the multi-channel mechanism can be regarded as multiple data augmentation on the original heterogeneous graph, and the channel consistency regularization can be viewed as the constraint on the augmentation. Through the multi-channel mechanism, the proposed MEGNN can learn more effective and robust node representations.

4.4. The objective function

For the task of node classification, the representations obtained by Eq. (17) are applied with a fully connected layer followed by a softmax layer. With P labeled nodes among all N nodes in heterogeneous graph, a standard cross entropy loss is employed on the labeled nodes as the supervised classification objective:

$$\mathcal{L}_{sup} = \frac{1}{P}\sum_{i=1}^{P} \mathbf{Y}_i \cdot \log Softmax(FC(\bar{\mathbf{Z}}_i)), \tag{20}$$

where $\mathbf{Y}_i$ is the one-hot label vector for the ith node and $FC(\cdot)$ refers to the fully connected layer.

In each epoch, we employ both the supervised classification loss in Eq. (19) and the channel consistency regularization loss in Eq. (20). Hence, the overall loss of MEGNN is

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda_{con}\mathcal{L}_{con}, \tag{21}$$

where λ_{con} is a hyper-parameter that controls the balance between the supervised classification loss and the channel consistency regularization loss.

4.5. Meta-path interpretation

With the aid of the heterogeneous convolution module, the importance of meta-paths can be calculated by multiplying the importance of each edge type on the path. For example, when only two layers are stacked in the model, the importance score of meta-path *A-P-A* is obtained by multiplying the weight of edge type *A-P* in the first layer by the weight of edge type *P-A* in the last layer.

Now we calculate the importance of the meta-paths extracted by MEGNN for further interpretation. For simplicity, we firstly assume the number of channels is one. Given a meta-path $\mathcal{P} = \mathcal{T}_1\mathcal{T}_2\cdots\mathcal{T}_{L+1}$, the importance of the meta-path $t_{\mathcal{P}}$ be calculated as

$$t_{\mathcal{P}} = \alpha^{(1)}_{R_1}\alpha^{(2)}_{R_2}\cdots\alpha^{(L)}_{R_L}, \tag{22}$$

where the coefficient $\alpha^{(k)}_{R_k}$ is the weight optimized in Eq. (12). It could either correspond to $\alpha^{(k)}_{\mathcal{T}_k\mathcal{T}_{k+1}}$ representing the importance of edge type $\mathcal{T}_k\mathcal{T}_{k+1}$ or correspond to $\alpha^{(k)}_{\mathbf{I}}$ representing the importance of the identity matrix in the kth heterogeneous convolution layer.

More generally, we denote the set of the importance of all meta-paths in the cth channel as $\{t^c_{\mathcal{P}_1}, t^c_{\mathcal{P}_2}, \ldots, t^c_{\mathcal{P}_Q}\}$, where Q is the total number of possible meta-paths. With the ultimate representations obtained by averaging different channels, the overall importance of meta-path $\mathcal{P}_i$ is calculated as

$$t_{\mathcal{P}_i} = \frac{1}{C}\sum_{c=1}^{C} t^c_{\mathcal{P}_i}. \tag{23}$$

Based on the calculation of meta-path importance, MEGNN is able to interpret its effectiveness and provide explainable insights about datasets.

5. Experiments

To demonstrate the effectiveness and interpretability of the proposed MEGNN, we conduct extensive experiments on real-world graph datasets, including node classification, meta-path interpretation and the ablation study of heterogeneous convolution as well as the sensitivity of hyper-parameters.

5.1. Dataset

We use the raw datasets provided in [20], i.e., ACM,[1] DBLP[2] and IMDB,[3] with their statistics shown in Table 1, and preprocess them as follows:

- **ACM**. Papers published in different conferences including KDD, SIGCOMM, MobiCOMM, SIGMOD and VLDB are extracted and divided into three classes (*Data Mining*, *Wireless Communication*, *Database*). This dataset consists of three kinds of nodes, including 3025 *Paper* nodes (*P*), 5845 *Author* nodes (*A*) and 57 *Subject* nodes (*S*). There are four types of edges (*P-A*, *A-P*, *P-S*, *S-P*). *Paper* nodes are the target nodes to be classified and labeled by the conferences they are published in.
- **DBLP**. A subset of DBLP dataset is extracted after preprocessing, containing 14,328 *Paper* nodes (*P*), 4057 *Author* nodes (*A*) and 20 *Conference* nodes (*C*) and four types of edges (*P-A*, *A-P*, *P-C*, *C-P*). In this heterogeneous graph, there are four research areas including *Database*, *Data Mining*, *Machine Learning* and *Information Retrieval*. *Author* nodes are the target nodes and labeled by their main research areas.

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

Table 1
Detailed statistics of the datasets.

Dataset	# Node	Node types	Edge	# Features	Target node	Dataset splitting	# Classes
ACM	8927	# *Paper* (*P*) : 3025 # *Author* (*A*) : 5845 # *Subject* (*S*) : 57	# *P-A* : 9889 # *P-S* : 3025	1902	*Paper*	# training : 600 # validation : 300 # test : 2125	3
DBLP	18,405	# *Paper (P)* : 14,328 # *Author* (A) : 4057 # *Conference* (*C*) : 20	# *P-A* : 19,645 # *P-C* : 14,328	354	*Author*	# training : 800 # validation : 400 # test : 2,857	4
IMDB	9317	# *Movie* (*M*) : 3189 # *Actor* (*A*) : 4435 # *Director* (*D*) : 1693	# *M-A* : 9563 # *M-D* : 3189	479	*Movie*	# training : 640 # validation : 320 # test : 2229	3

- **IMDB**. A subset of IMDB dataset is extracted which contains 3189 *Movie* (*M*), 4435 *Actor* (*A*) and 1693 *Director* (*D*) and four types of edges (*M-A*, *A-M*, *M-D*, *D-M*). For IMDB dataset, the task is to classify the *Movie* nodes into three genres: (*Comedy*, *Action*, *Drama*). In this preprocessed dataset, there are no movies that belong to two categories. In addition, it is worth mentioning that this dataset is imbalanced compared with ACM and DBLP, where the numbers of *Drama* movies and *Comedy* movies are three times and two times as large as the number of *Action* movies, respectively.

For the citation datasets (ACM and DBLP), the features of *Paper* nodes are the bag-of-words representations of their keywords. For nodes without attributes, each *Author* node is represented by the features of their own papers and each *Conference* node is represented by the features of papers published in that conference. For the IMDB dataset, the features of *Movie* nodes are the bag-of-words representation of different plots. Likewise, each *Actor* node is represented as bag-of-words features of the movies they played in and each *Director* node is represented as bag-of-words of the features of the director's movies.

5.2. Baselines

To evaluate the effectiveness of MEGNN in heterogeneous graphs, we compare MEGNN with different state-of-the-art methods. Based on the characteristics of these methods and how they are applied in heterogeneous graphs, these methods are categorized into three groups: conventional graph embedding methods, GNN methods based on pre-defined meta-paths, and GNNs methods based on modeling heterogeneity. The list of baseline methods is listed as follows:

Conventional graph embedding methods:

- **Deepwalk** [22] is a random walk based graph embedding method designed for homogeneous graphs. Here we ignore the heterogeneity of nodes and edges and perform Deepwalk in the whole heterogeneous graph.
- **Metapath2vec** [18] is a heterogeneous graph embedding method adapted from Deepwalk, which learns embeddings from meta-path based random walks. Here we test all meta-paths and report the best results.

GNN methods based on pre-defined meta-paths:

- **GCN** [9] is a homogeneous graph neural network that performs convolutional operations to update the embeddings. Here we apply GCN in all meta-path based homogeneous graphs and report the best results.
- **GAT** [10] is a homogeneous graph neural network that considers attention mechanism to update the embeddings. Here we apply GAT in all meta-path based homogeneous graphs and report the best results.
- **HAN** [20] is a heterogeneous graph neural network that leverages node-level attention and semantic-level attention to obtain node embeddings.
- **MAGNN** [21] is a heterogeneous graph neural network that proposes to aggregate the intermediate nodes along the meta-path instances with intra- and inter-aggregation.

GNN methods based on modeling heterogeneity:

- **RGCN** [35] is a heterogeneous graph neural network adapted from GCN, which learns specific transformation in each layer for each type of edges.
- **GTN** [36] is a heterogeneous graph neural network that transforms the heterogeneous graph into multiple new meta-path based graphs and learns node embeddings via graph convolutional network.

5.3. Parameter settings

For a fair comparison, we set the dimension of ultimate embeddings to 64 for all models. The splitting of the training set, validation set and test set is exactly the same for all methods above. For methods based on random walk, including Deepwalk, Metapath2vec, we set the window size to 5, walk length to 100, walks per node to 40, and the negative samples to 5. For methods based on pre-defined meta-paths, the meta-path candidate sets are set as {*P-A-P*, *P-S-P*}, {*A-P-A*, *A-P-C-P-A*} and {*M-D-M*, *M-A-M*} for ACM, DBLP and IMDB, respectively. For MAGNN, we adopt the relational rotation encoder for the meta-path instances as the authors suggested. For GCN, GAT and RGCN, we set the dropout rate to 0.5, the number of attention heads to 8 and the number of base transformations as 4. For HAN and GTN, we run the released code with suggested hyper-parameters. For the proposed MEGNN, we apply early stopping with a patience of 50 and employ the Adam optimizer with a learning rate of 0.005 and the L2 regularization parameter of 0.001. The number of channels and layers are both set as 4 for the balance of performance and efficiency.

5.4. Node classification

The performance of all kinds of methods is evaluated in the task of node classification. Approximately 20% of the target nodes (i.e. *Paper* nodes for ACM, *Author* nodes for DBLP and *Movie* nodes for IMDB) are used for training, 10% of the nodes are used for tuning the hyperparameters and the remaining 70% nodes are used for testing. We run each method for 10 times and report the average *Micro-F1* and *Macro-F1* in Table 2.

As shown in Table 2, the proposed method MEGNN outperforms other baselines on all datasets, especially on the unbalanced IMDB dataset. We can also conclude from the table that the GNN-based methods which simultaneously combine topological structures and node attributes generally perform better than conventional graph embedding methods that neglect node attributes. This verifies the benefit of incorporating node attributes into the model. Besides, it is interesting that Metapath2vec performs even worse than DeepWalk on the ACM dataset, which indicates

Table 2
Experiment results (%) in the node classification task.

	ACM		DBLP		IMDB	
	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1
DeepWalk	80.87	81.31	79.91	78.35	55.81	55.95
Metapath2vec	67.46	67.56	87.36	85.79	56.05	46.60
GCN	90.86	90.93	92.14	91.18	62.22	53.86
GAT	90.22	90.30	92.12	91.34	61.72	56.56
HAN	91.03	91.10	92.71	92.51	63.14	57.09
MAGNN	90.85	90.63	93.72	92.71	64.86	58.39
GTN	91.34	91.49	93.57	92.18	65.10	58.55
RGCN	92.39	92.49	92.91	92.17	63.78	55.17
MEGNN	**92.57**	**92.66**	**94.84**	**94.22**	**67.80**	**62.73**

Table 3
Quantitative results for the efficiency of MEGNN and the baselines. We record the total number of parameters and the training time per epoch in milliseconds (ms). For convenience, we take the result of GCN as 1, and record the relative multiples of other methods.

	# params	Multiples	Training time (ms)	Multiples
GCN	22.9k	1	8.14	1
GAT	23.1k	1.01	10.64	1.31
HAN	54.4k	2.38	95.75	11.76
MAGNN	62.5k	2.73	103.41	12.70
GTN	102.4k	4.47	118.75	14.58
RGCN	114.7k	5.01	18.38	2.26
MEGNN	92.2k	4.03	35.75	4.39

that the methods based on limited pre-defined meta-paths may have negative impacts on the model's effectiveness. This is also demonstrated by the fact that GCN, GAT and HAN that utilize pre-defined meta-paths perform worse in most cases compared to other methods that directly model the edge type heterogeneity, i.e., RGCN and GTN. It indicates that the heterogeneity of graphs should be carefully modeled beyond pre-defined meta-paths, otherwise the models may ignore some meaningful local structures in the graphs. Though the method MAGNN achieves rather competitive performance against GTN while only using limited meta-path schemes, in essence, it can also be viewed as a special case of the models that automatically capture heterogeneity. With the relational rotational encoder, it could extract additional heterogeneity semantics within the meta-path sequences. However, this ability is still limited within the given meta-path schemes. On account of MEGNN generating different meta-paths and training their weights, it can actually both learn meta-path knowledge and discover effective graph local structures. We will verify the effectiveness of extracted meta-paths next.

5.5. The interpretability of extracted meta-path

We conduct an experiment to verify the interpretability of meta-paths extracted by MEGNN. In this experiment, we stack four heterogeneous convolution layers on DBLP, and present the importance of extracted meta-paths of the initial epoch and the final epoch in Figs. 5(a) and 5(b), respectively. As *Author* is the node type to be classified, hence we only show the importance of meta-path schemes targeted at *Author* nodes.

We firstly focus on the initial importance of extracted meta-paths in Fig. 5(a). We could observe that meta-paths with middle lengths have relatively high weights. This is because the weight of α_e and $\alpha_{\mathbf{l}}$ are both uniformly initialized, thus whether meta-path schemes expand is subject to a Bernoulli distribution. Totally, the extracted meta-paths in the initial epoch are determined by the initialization strategy and cannot reflect the semantic relationships in the heterogeneous graph.

Then we examine the meta-paths extracted by MEGNN of the final epoch in Fig. 5(b). It is obvious that the top three meta-paths are *A-P-C-P-A*, *A-P-C-P* and *A-P-C-P-C*, which collectively contributes approximately 80% importance among all meta-path schemes. The most significant meta-path *A-P-C-P-A* is the conventional meta-path usually used in the citation graph. It suggests that MEGNN can learn some meta-paths consistent with pre-defined domain knowledge. Besides, MEGNN also discovers some important meta-paths that are not included in the pre-defined meta-path set. For example, *A-P-C-P* also ranks top of the extracted importance. This meta-path scheme reveals the semantic relationships that the authors' research fields are similar to the papers co-published at the same conferences with these authors' papers, which is reasonable and commonly neglected by conventional meta-path based methods.

Besides, we also observe that another conventional meta-path *A-P-A* is not indicative as expected, i.e., merely contributes about 4% importance to the ultimate representations. We reason that the importance of different semantic relationships varies according to the properties of datasets and co-authorship does not suggest adequate information may be due to its sparsity in this dataset. With the interpretation above, we conclude that MEGNN is able to discover significative meta-path structures and provide insights about graph data by meta-paths importance.

5.6. Efficiency analysis

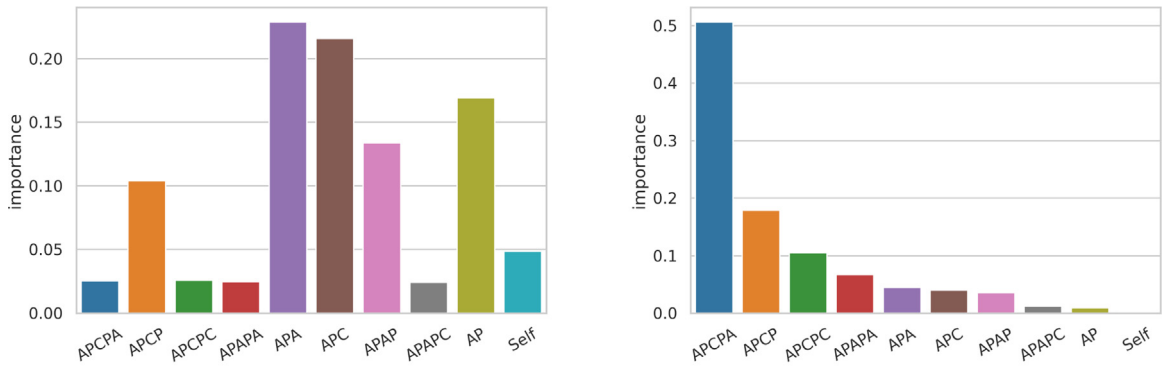
Now we firstly show the time complexity of the proposed MEGNN and then provide the results and the analysis in the efficiency experiments.

MEGNN comprises the heterogeneous convolutions in multiple channels and the channel consistency regularization. For each heterogeneous convolution module, it only requires training additional $(|\mathcal{T}^e| + 1)$ parameters to transform the heterogeneous graph, and its time complexity is $O(|\mathcal{E}|\,|T^e|)$, where $|\mathcal{E}|$ is the number of edges in the graph. The message propagation in MEGNN in a single channel has the complexity of $O(L|\mathcal{E}|d + NMd)$, where d is the dimension of node embeddings and M is the number of original node attributes. The complexity of the channel consistency regularization is $O(CNd)$. If we use C channels, which is usually a small number like 4, the complexity of the whole MEGNN is $O(CL|\mathcal{E}||T^e| + CL|\mathcal{E}|d + CNMd + CNd)$, which is both linear with the number of nodes and edges in the graph.

Practically, we record the number of parameters and the training time averaged in each epoch of MEGNN and other GNN-based baselines in Table 3. We do not include the random walk-based methods in the comparison for fairness, since the way they perform batch training is different from GNNs which are usually trained in the transductive setting. As is shown in the table, GTN is the most time-consuming model over all methods, as it requires consecutive multiplication of the adjacency matrix. HAN and MAGNN have large overhead as they have to spend much time on the calculation of attentions. In contrast, MEGNN leverages the multi-channel heterogeneous convolutions to avoid the computation of attention, thus its training time only increases linearly with the number of channels, which is set as 4 in the experiments. To summarize, the results in the table demonstrate that MEGNN can learn the node representation effectively and efficiently, and is superior to the baseline methods considering the balance between performance and computation.

5.7. Ablation study

In addition to analyzing the interpretability of MEGNN, we conduct an ablation study to validate the effectiveness of the components of the proposed model. We derive four MEGNN variants and evaluate their performance in the three datasets. The four variants are described as follows:



(a) Initial epoch (b) Final epoch

Fig. 5. The importance of extracted meta-paths in different training stages.

Table 4
Quantitative results (%) for ablation study of MEGNN.

	ACM		DBLP		IMDB	
	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1
MEGNN	**92.57**	**92.66**	**94.84**	**94.22**	**67.80**	**62.73**
MEGNN$_{-hetconv}$	85.39	84.06	90.71	90.47	60.58	54.54
MEGNN$_{-id}$	91.36	91.51	94.09	93.91	64.40	56.57
MEGNN$_{-sharing}$	91.98	92.10	93.59	92.73	65.50	61.27
MEGNN$_{-reg}$	92.32	92.44	93.81	93.11	66.44	60.97
MEGNN$_{-channel}$	91.84	91.26	92.17	91.22	65.83	59.81

- MEGNN$_{-hetconv}$: Remove the whole heterogeneous convolution module in each layer, i.e., the heterogeneity of nodes and edges is ignored.
- MEGNN$_{-id}$: Remove the identity matrix included in each heterogeneous convolution module. In this case, the model can only extract meta-path schemes that have exactly the same length as the number of layers.
- MEGNN$_{-sharing}$: Remove the type-wise weight sharing mechanism in each heterogeneous module, indicating that the structures of extracted meta-paths in different layers do not correlate with each other.
- MEGNN$_{-reg}$: Remove the channel consistency regularization and still keep the mechanism of multi-channels in the model, i.e., $\lambda_{con} = 0$.
- MEGNN$_{-channel}$: Remove the mechanism of multi-channels in the model, including the message passing in multiple channels and the channel consistency regularization.

The results of the ablation study are reported in Table 4. As can be seen, MEGNN achieves better performance than the MEGNN$_{-reg}$, demonstrating channel consistency regularization is effective for generating better representations. If we further remove the whole multi-channel mechanism as MEGNN$_{-channel}$, the model presents further performance degradation, which verifies the necessity of the multi-channel mechanism to improve the model's performance. Besides, we can conclude that the proposed whole heterogeneous convolution module has a significant contribution to the effectiveness of the method based on the performance degeneration of MEGNN$_{-hetconv}$. More specifically, missing the identity matrix or the type-wise weight sharing mechanism within the heterogeneous module leads to varying degrees of performance decline. Note that the performance of MEGNN$_{-id}$ has only a slight decrease on DBLP, indicating that long meta-path schemes are more beneficial on the dataset. This also corresponds to the majority importance of *A-P-C-P-A* shown in Fig. 5(b) and the significant performance promotion in the DBLP case of Fig. 7 when the model's depth varies from 1 to 4.

5.8. Parameter analysis

In this section, we investigate the sensitivity of the hyper-parameters in MEGNN. More specifically, we evaluate how the number of channels, the number of stacked layers and the coefficient of the channel consistency regularization λ_{con} affect the results of node classification.

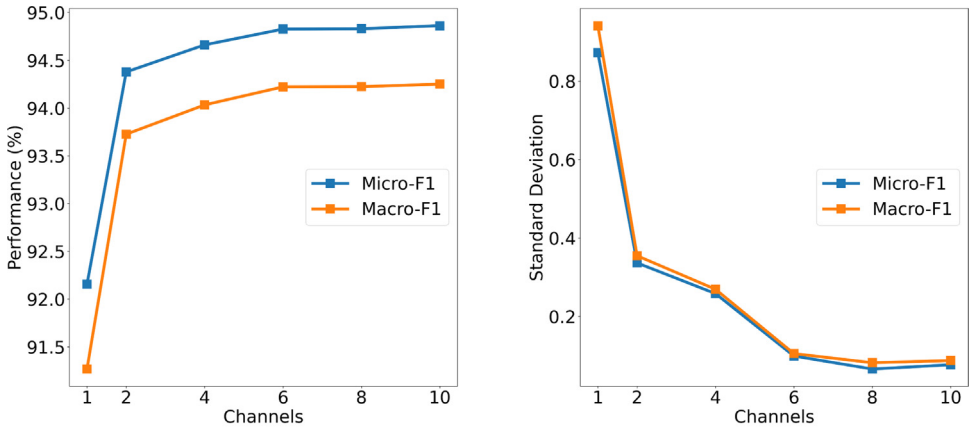
The number of channels. The effect of channels is studied in two aspects, including both performance and stability. We run MEGNN on DBLP with different number of channels for 10 times, and record the average *Micro-F1*, *Macro-F1* and their standard deviation. The results of performance and stability are shown in Figs. 6(a) and 6(b), respectively. As the growth of channels, MEGNN gains considerable improvements in *Micro-F1* initially and remains steady ultimately. Meanwhile, the standard deviation of MEGNN declines by a big margin at first and continues to fall slowly. Based on the analysis above, we can conclude that MEGNN can achieve better performance and higher stability with more channels. Empirically the number of channels is set to 4 considering the balance between performance and efficiency.

The number of layers.. As we can observe in Fig. 7, the best depth varies according to the datasets. Concretely, MEGNN can quickly capture adequate information on ACM and IMDB datasets with only two or three layers, and has slightly poorer performance with layers continuing to grow. For DBLP, the model with four layers achieves significant improvement in *Micro-F1* compared to shallow MEGNN with two or three layers and tends to become stable when the model's depth further increases. It indicates the meta-paths of over length four contain unique information neither included in 1-order nor 2-order neighbors in the dataset.

The impact of λ_{con}. We investigate the sensitivity of the channel consistency regularization coefficient λ_{con}. The results on three datasets are illustrated in Fig. 8. As λ_{con} grows, the performance of MEGNN increases at first and reaches the peak, then begins to drift down slowly. The reason for this two-stage change is that MEGNN requires an appropriate weight to control the consistency of learned representations and a larger regularization may deteriorate the model's ability to exploit diverse meta-path structures.

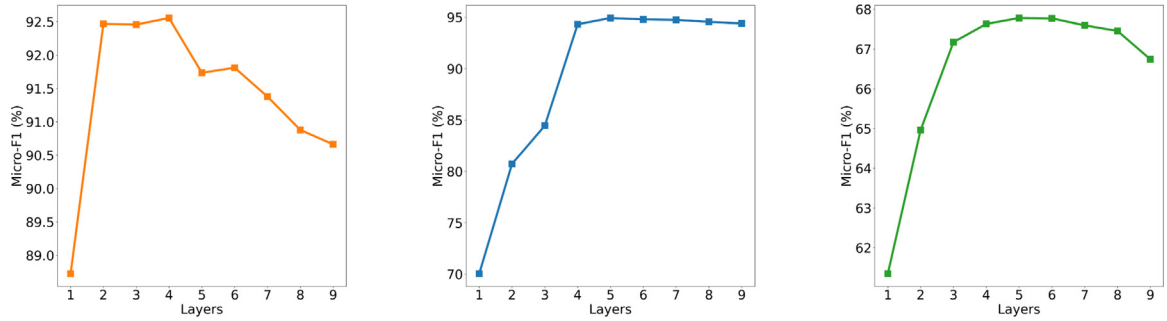
6. Conclusion

In this paper, we present MEGNN for node representation learning in heterogeneous graphs. MEGNN tactfully adopts the message passing paradigm to encode the graph topological structures, node attributes and semantic relationships into node embeddings simultaneously. Besides, MEGNN can extract significative meta-paths without any prior knowledge of the specific



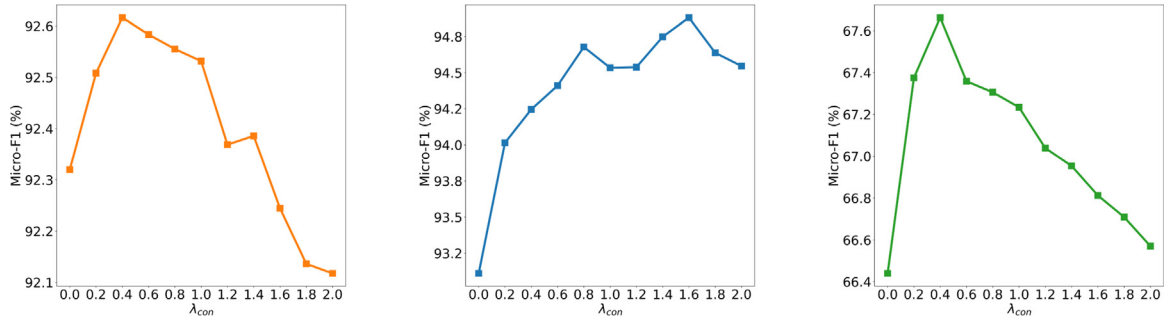
(a) Performance (b) Stability

Fig. 6. The impact of channels on the model's performance and stability on DBLP.



(a) ACM (b) DBLP (c) IMDB

Fig. 7. The impact of the model's depth on the datasets.



(a) ACM (b) DBLP (c) IMDB

Fig. 8. The impact of λ_{con} on the datasets.

scenarios, providing explainable conclusions to the model's effectiveness. The comprehensive experiments on three real-world datasets demonstrate that the proposed MEGNN outperforms the state-of-the-art methods in the task of node classification and verify the interpretability of extracted meta-paths. Moreover, the core component of MEGNN, i.e., heterogeneous convolution, has a considerable superiority to model the graph heterogeneity and strong adaptability and operability for large-scale datasets. With the aid of its domain-independent nature and efficiency, MEGNN is applicable for heterogeneous graphs in different heterogeneous scenarios, e.g., e-commerce systems, in-game friendship recommendations and so on.

CRediT authorship contribution statement

Yaomin Chang: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Chuan Chen:** Conceptualization, Methodology, Writing – review & editing, Supervision. **Weibo**

Hu: Conceptualization, Methodology, Visualization, Writing – review & editing. **Zibin Zheng:** Resources, Project administration. **Xiaocong Zhou:** Validation, Writing – review & editing. **Shouzhi Chen:** Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The research is supported by the Key-Area Research and Development Program of Guangdong Province (2020B010165003), the National Natural Science Foundation of China (62176269, 11801595), the Guangdong Basic and Applied Basic Research Foundation (2019A1515011043) and the Tencent Wechat Rhinobird project (2021321).

References

[1] N. Dahm, H. Bunke, T. Caelli, Y. Gao, A unified framework for strengthening topological node features and its application to subgraph isomorphism detection, in: International Workshop on Graph-Based Representations in Pattern Recognition, Springer, 2013, pp. 11–20.
[2] P. Riba, J. Lladós, A. Fornés, A. Dutta, Large-scale graph indexing using binary embeddings of node contexts for information spotting in document image databases, Pattern Recognit. Lett. 87 (2017) 203–211.
[3] Z. Chen, T. Cai, C. Chen, Z. Zheng, G. Ling, Sine: Side information network embedding, in: International Conference on Database Systems for Advanced Applications, 2019, pp. 692–708.
[4] H. Gao, Z. Wang, S. Ji, Large-scale learnable graph convolutional networks, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1416–1424.
[5] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: International Conference on Machine Learning, 2016, pp. 2071–2080.
[6] Z. Liu, V.W. Zheng, Z. Zhao, F. Zhu, K.C.-C. Chang, M. Wu, J. Ying, Semantic proximity search on heterogeneous graph by proximity embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2017, pp. 154–160.
[7] F. Ye, C. Chen, Z. Zheng, R.-H. Li, J.X. Yu, Discrete overlapping community detection with pseudo supervision, in: 2019 IEEE International Conference on Data Mining, 2019, pp. 708–717.
[8] Z. Chen, C. Chen, Z. Zhang, Z. Zheng, Q. Zou, Variational graph embedding and clustering with laplacian eigenmaps, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019, pp. 2144–2150.
[9] T.N. Kipf, M. Welling, Semi-Supervised classification with graph convolutional networks, in: 5th International Conference on Learning Representations, ICLR 2017.
[10] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: 6th International Conference on Learning Representations, ICLR, 2018.
[11] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, K. Weinberger, Simplifying graph convolutional networks, in: International Conference on Machine Learning, 2019, pp. 6861–6871.
[12] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems, 2017, pp. 1024–1034.
[13] S. Abu-El-Haija, B. Perozzi, R. Al-Rfou, A.A. Alemi, Watch your step: Learning node embeddings via graph attention, in: Advances in Neural Information Processing Systems, 2018, pp. 9180–9190.
[14] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, S. Jegelka, Representation learning on graphs with jumping knowledge networks, in: International Conference on Machine Learning, 2018, pp. 5453–5462.
[15] J. Chen, T. Ma, C. Xiao, FastGCN: Fast learning with graph convolutional networks via importance sampling, in: 6th International Conference on Learning Representations, ICLR 2018.
[16] C. Shi, Y. Li, J. Zhang, Y. Sun, S.Y. Philip, A survey of heterogeneous information network analysis, IEEE Trans. Knowl. Data Eng. 29 (1) (2016) 17–37.
[17] Y. Sun, J. Han, X. Yan, P.S. Yu, T. Wu, Pathsim: Meta path-based top-k similarity search in heterogeneous information networks, Proc. VLDB Endowment 4 (11) (2011) 992–1003.
[18] Y. Dong, N.V. Chawla, A. Swami, metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.
[19] C. Shi, B. Hu, W.X. Zhao, S.Y. Philip, Heterogeneous information network embedding for recommendation, IEEE Trans. Knowl. Data Eng. 31 (2) (2018) 357–370.
[20] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P.S. Yu, Heterogeneous graph attention network, in: The World Wide Web Conference, 2019, pp. 2022–2032.
[21] X. Fu, J. Zhang, Z. Meng, I. King, MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding, in: Y. Huang, I. King, T. Liu, M. van Steen (Eds.), The World Wide Web Conference, 2020, pp. 2331–2341.
[22] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 701–710.
[23] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.
[24] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: Y. Bengio, Y. LeCun (Eds.), 1st International Conference on Learning Representations, ICLR 2013.
[25] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 1067–1077.
[26] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1225–1234.
[27] S. Cao, W. Lu, Q. Xu, Grarep: Learning graph representations with global structural information, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015, pp. 891–900.
[28] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, J. Tang, Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 459–467.
[29] J. Qiu, Y. Dong, H. Ma, J. Li, C. Wang, K. Wang, J. Tang, Netsmf: Large-scale network embedding as sparse matrix factorization, in: The World Wide Web Conference, 2019, pp. 1509–1520.
[30] Z. Zhang, H. Yang, J. Bu, S. Zhou, P. Yu, J. Zhang, M. Ester, C. Wang, Anrl: Attributed network representation learning via deep neural networks., in: Twenty-Seventh International Joint Conference on Artificial Intelligence, 2018, pp. 3155–3161.
[31] H. Gao, H. Huang, Deep attributed network embedding, in: Twenty-Seventh International Joint Conference on Artificial Intelligence, 2018, pp. 3364–3370.
[32] J. Tang, M. Qu, Q. Mei, Pte: Predictive text embedding through large-scale heterogeneous text networks, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 1165–1174.
[33] S. Chang, W. Han, J. Tang, G.-J. Qi, C.C. Aggarwal, T.S. Huang, Heterogeneous network embedding via deep architectures, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 119–128.
[34] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, J. Tang, Representation learning for attributed multiplex heterogeneous network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1358–1368.
[35] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: European Semantic Web Conference, 2018, pp. 593–607.
[36] S. Yun, M. Jeong, R. Kim, J. Kang, H.J. Kim, Graph transformer networks, in: Advances in Neural Information Processing Systems, 2019, pp. 11960–11970.
[37] J. Gilmer, S.S. Schoenholz, P.F. Riley, O. Vinyals, G.E. Dahl, Neural message passing for quantum chemistry, in: Proceedings of the 34th International Conference on Machine Learning, 2017, pp. 1263–1272.