Expert Systems With Applications 225 (2023) 120115

Contents lists available at ScienceDirect

Expert Systems With Applications

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

OSGNN: Original graph and Subgraph aggregated Graph Neural Network

Yeyu Yan [a], Chao Li [a,*], Yanwei Yu [b], Xiangju Li [a], Zhongying Zhao [a,*]

[a] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong 266590, China
[b] College of Computer Science and Technology, Ocean University of China, Qingdao, Shandong 266100, China

ARTICLE INFO

Keywords:
Graph neural networks
Heterogeneous graphs
Graph representation learning
Heterogeneous networks

ABSTRACT

Heterogeneous Graph Embedding (HGE) is receiving a great attention from researchers, as it can be widely and effectively used to solve problems from various real-world applications. The existing HGE models mainly learn node representation directly on the whole heterogeneous graph by aggregating neighboring information, which unavoidably leads to the loss of useful high-order information. Another mainstream is to split heterogeneous graphs into different homogeneous subgraphs and then learn representations separately. However, this isolated handling way is prone to the loss of important interactions between the nodes of the same type. To address the above challenging but interesting problems, we propose an Original graph and Subgraph aggregated Graph Neural Network (OSGNN). Specifically, we first split the original heterogeneous graph into several subgraphs, and then weighted combine them to get a new meaningful homogeneous graph. Finally, the first-order and high-order information of the target node are learned from the original heterogeneous graph and the homogeneous subgraph respectively and concatenated as the final node representation. Extensive experiments on three real-world heterogeneous graphs demonstrate that the proposed framework significantly outperforms the state-of-the-art methods. The source codes of this work are available on https://github.com/ZZY-GraphMiningLab/OSGNN.

1. Introduction

Due to the remarkable achievements of graph neural networks (GNNs) (Welling & Kipf, 2017; Wu et al., 2020) in non-Euclidean fields, researchers are paying attention to their promotion and potential applications in the real world, such as social networks (Moscato & Sperlì, 2021; Xie et al., 2021), transportation networks (Andreoletti et al., 2016; Zhang et al., 2018), academic networks (Atwood & Towsley, 2016; Hamilton et al., 2017) and so on (Li et al., 2022; Qian et al., 2022; Salamat et al., 2021). To avoid the loss of the necessary information and improve the representative ability of graph neural networks (Hamilton et al., 2017; Velickovic et al., 2018; Welling & Kipf, 2017), the great efforts have been made to explore heterogeneous graphs (HGs). Heterogeneous graph neural networks (HGNNs) (Wang et al., 2019; Yang et al., 2020; Yun et al., 2019) have become an effective solution to the above application scenarios because of their strong representation ability.

However, with the natural property of multi-type nodes and edges in heterogeneous graphs, some of the drawbacks of the graph neural networks may be amplified. For example, different types of node features belong to different spaces, how to effectively aggregate different types of node representations is still an open question. In addition, due to the existence of heterogeneous nodes, most nodes of the same type cannot be connected and learn from each other directly. Therefore, when modeling and learning heterogeneous graphs, we are facing two kinds of challenges.

Challenge 1: Semantic confusion is one of the most important problems that limit the performance of HGNNs (Ji et al., 2023). There are two main reasons for this phenomenon. One is that different types of nodes belong to different spaces. If these node features are aggregated directly without considering different spatial information, it undoubtedly introduces more noises. The other is that when the number of network layers of HGNNs increases, each node continuously aggregates more and more noise information, resulting in the phenomenon that representations of different nodes become similar. A straightforward method is to convert the heterogeneous graph into homogeneous graphs by considering different metapaths (Ji et al., 2023; Wang et al., 2019). However, it ignores heterogeneous nodes and edges, which undoubtedly causes information loss. Another solution is to learn the local information of nodes merely. In other words, the HGNNs are designed with small-layer (1-layer or 2-layer) heterogeneous graph convolution to alleviate semantic confusion. However, it leads to the loss of the same type of higher-order information.

* Corresponding authors.
E-mail addresses: yanyeyu-work@foxmail.com (Y. Yan), lichao@sdust.edu.cn (C. Li), yuyanwei@ouc.edu.cn (Y. Yu), skd996730@sdust.edu.cn (X. Li), zyzhao@sdust.edu.cn, zzysuin@163.com (Z. Zhao).

https://doi.org/10.1016/j.eswa.2023.120115
Received 16 September 2022; Received in revised form 7 March 2023; Accepted 8 April 2023
Available online 14 April 2023

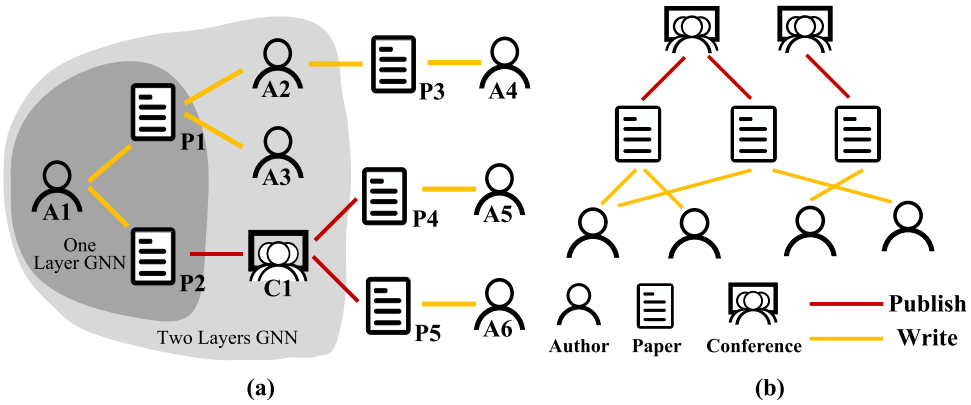

Fig. 1. (a) Illustration of a 2-layer HGNN and (b) A heterogeneous graph of DBLP dataset.

Challenge 2: The nodes of the same type should be well explored to strengthen mutual learning. Different from homogeneous graphs, the nodes of the same type do not connect each other directly. They are linked with multiple intermediate heterogeneous nodes, thus they cannot learn from each other easily. Although some methods based on metapath can directly learn high-order information (Li et al., 2023; Wang et al., 2019), they ignore local heterogeneous information. Fig. 1 is a simple heterogeneous graph to describe the entities and relationships on DBLP dataset. Obviously, there are three kinds of nodes (author, paper, and conference) and two types of relationships (publish and write). Taking $Author_1$ as the target node, a 2-layer HGNN is adopted to learn node representations. We find that the 2-layer HGNN cannot make $Author_1$ learn the information of $Author_5$, even though $Author_5$ and $Author_1$ are the nodes with the same type. On DBLP dataset, the author's labels depend on the conference nodes. If $Author_5$ and $Author_1$ are used to learn from each other directly, this can further improve the representation ability of the HGNN. This view has been proved in heterogeneous graph attention network (HAN) (Wang et al., 2019) and heterogeneous graph propagation network (HPN) (Ji et al., 2023).

Generally speaking, the existing models are able to alleviate semantic confusion by learning the first-order information of nodes, but they cannot effectively learn the information of high-order nodes of the same type. If the higher-order information is aggregated directly with metapath or metagraph, it will bring the loss of local information of nodes. Some recent HGNNs have made some efforts to retain both local and high-order information at the same time. HMSG (Cai et al., 2021) aggregates information from both homogeneous and heterogeneous neighbors to capture structural, semantic and attribute information in heterogeneous graphs. HGNN-AC (Cai et al., 2021) first completes missing attributes of local nodes with attribute completion module, and then aggregates higher-order neighbors by metapaths. However, it has high computational complexity, and it ignores the interaction between different metapaths. As pointed out in Wang et al. (2022), the above methods treat each metapath as an isolated semantic data resource and disregard the interaction among them in learning metapath embeddings.

To solve the above challenges, we propose an **O**riginal graph and **S**ubgraph aggregated **G**raph **N**eural **N**etwork (OSGNN). For Challenge 1, OSGNN learns local and high-order information from two different perspectives and uses concatenation to maximize the retention of these two types of information. For Challenge 2, OSGNN generates different subgraphs from the original heterogeneous graph according to different metapaths, then uses graph-level attention to fuse these subgraphs to generate a new homogeneous graph. This process is able to overcome the problem of semantic information isolation (Wang et al., 2022), so as to capture the interaction between different metapaths. Finally, the aggregate representations and initial features are weighted and summed to obtain the final representations. The model is trained and optimized with the loss function of downstream tasks.

The contributions of this work are summarized as follows:

- We present an original graph and subgraph aggregated graph neural network. It not only directly learns the local information from the original heterogeneous graph, but also learns higher-order information easily by generating a new meaningful homogeneous graph with multiple metapaths.
- Metapath induced subgraphs are assigned to different weights by employing graph level attention mechanism. In addition, the semantic intensity of different metapaths is retained to capture interactions between high-order nodes.
- We conducted experiments on three commonly used real-world datasets. The experimental results demonstrate that the proposed OSGNN is effective and outperforms the state-of-the-art methods significantly.

The remainder of this paper is organized as follows. Section 2 briefly reviews the related work. The related concepts and formal definitions of heterogeneous graphs are introduced in Section 3. In Section 4, we present the framework and the algorithm of the OSGNN. Section 5 evaluates the effectiveness of the OSGNN with the experimental results and analysis. Finally, Section 6 summarizes the research work of this paper.

2. Related work

2.1. Graph Neural Network (GNN)

The purpose of GNNs is to learn the low dimensional vector representation of nodes by aggregating the topological information of the graph and the node features. The existing methods are mainly classified into two categories: spectral GNNs (Bruna et al., 2014; Defferrard et al., 2016) and spatial GNNs (Gilmer et al., 2017; Hamilton et al., 2017; Monti et al., 2017). For spectral GNNs, they learn the node representation by convolution in the Fourier domain of the graph. Graph convolution network (GCN) (Welling & Kipf, 2017) is the pioneer of GNN models, which uses the first-order approximation of spectral convolution to simplify ChebNet (Bruna et al., 2014). However, the drawback of spectral GNNs is that their filters depend on the Laplace basis of the graph, which is closely related to the specific graph structure. Spatial GNNs utilize spatially close neighbors to define convolution operations directly on the graph. Hamilton et al. proposed GraphSAGE (Hamilton et al., 2017), which can directly perform information aggregation on fixed-size node neighbors. Graph attention network (GAT) (Velickovic et al., 2018) aggregates node features with an attention mechanism and assigns different weights to different nodes.

The above GNNs are only applicable to homogeneous graphs. Considering that the node features in heterogeneous graphs are not in the same space, these GNNs cannot model them directly.

2.2. Heterogeneous Graph Neural Network (HGNN)

HGNNs are mainly used to model heterogeneous graphs. They can learn more complete node representations with a variety of heterogeneous node information. The existing HGNNs are mainly divided into two categories: one is to aggregate neighbor information directly based on edge type, and the other is to aggregate higher-order information based on metapath or metagraph.

HGNNs based on edge type: Graph transformer networks (GTN) (Yun et al., 2019) uses the adjacency matrix of multiple edge subgraphs and matrix multiplication to find and generate valuable metapath neighborhood graphs in heterogeneous graphs, and then encodes the graphs using GCN. Metapath extracted graph neural network (MEGNN) (Chang et al., 2022) leverages heterogeneous convolution to combine different relation subgraphs corresponding to edge types into a new graph structure. Heterogeneous graph neural networks with denoising (HGNND) (Dong et al., 2022) uses attention to learn more important information on heterogeneous graph convolution. Then it filters out potential noisy nodes with denoising operations to further enhance the node representation. Heterogeneous graph structural attention neural network (HetSANN) (Hong et al., 2020) aggregates neighborhood node information with a specific graph attention layer, thus avoiding the manual selection of metapaths. Heterogeneous graph structure learning (HGSL) (Zhao et al., 2021) generates a heterogeneous graph with a more perfect structure by aggregating feature graph, semantic graph, and original graph.

HGNNs based on metapath or metagraph: HAN (Wang et al., 2019) is an improved model based on GAT (Velickovic et al., 2018). It first converts a heterogeneous graph into multiple homogeneous subgraphs with different metapaths and then uses GAT to learn the target node representations. Finally, it aggregates the representations of different subgraphs with the attention mechanism to obtain the final representation. Its disadvantage is that it ignores other types of node information, resulting in information loss. Metapath aggregated graph neural network (MAGNN) (Fu et al., 2020) is regarded as an improved HAN (Wang et al., 2019) model, which divides the original graph into multiple bipartite graphs according to different metapaths. Unlike HAN, it retains the information of intermediate heterogeneous nodes. CHEST (Wang et al., 2023) uses heterogeneous subgraph transformer to capture metapath semantics, and performs pre-training of node embedding with three kinds of generative losses. Higher-order attribute-enhancing (HAE) (Li et al., 2023) learns the importance of higher-order information with metagraph, and then aggregates them by node-level attention. HPN (Ji et al., 2023) is similar to HAN in that it preserves the original information with hyper-parameters to avoid semantic confusion.

However, most models (such as HAE (Li et al., 2023), HPN (Ji et al., 2023), HIGCN (Chen et al., 2022), and so on) are based on metapath or metagraph, and they capture high-order information (second order and above information) of nodes by converting the heterogeneous graph into homogeneous subgraphs. As shown in Fig. 3, the academic coauthor heterogeneous network is converted into a homogeneous network which is composed of author nodes. Hence, the features of the paper nodes and the conference nodes are discarded. The models which are based on edge type (such as HetSANN (Hong et al., 2020), HGSL (Zhao et al., 2021), and so on) can directly capture the local information of nodes. However, the high-order information has to be captured by stacking network layers, leading to the problem of semantic confusion stated in challenge 1. Therefore, the model proposed in this paper is able to learn the local information of nodes to effectively avoid semantic confusion on the one hand, and learn the interactive information between high-order nodes on the other hand.

3. Preliminary

In this section, we first define some notations used in this paper, as shown in Table 1. We then introduce some basic concepts and formulate the problem to be solved in this paper.

Definition 1 (*Heterogeneous Graph (HG)*)**.** A **heterogeneous graph** (Sun & Han, 2012) (a.k.a., Heterogeneous Information Network) is denoted as a network $G = (V, E)$. It is composed of node set V and edge set E, including node type mapping function $\Phi : V \rightarrow T$ and edge type mapping function $Y : E \rightarrow R$, where T and R represent the set of node type and edge type respectively, $|T| + |R| > 2$.

Definition 2 (*Metapath*)**.** A **metapath** (Sun & Han, 2012; Sun et al., 2011) is defined as a path in the heterogeneous graph G: $v_1 \xrightarrow{r_1} v_2 \xrightarrow{r_2} \cdots \xrightarrow{r_l} v_{l+1}$, where $v \in T$, $r \in R$. It describes the composite relationship $r = r_1 \circ r_2 \circ \cdots \circ r_l$ between node v_1 and node v_{l+1}, where $\circ$ represents the compound operator on relationships. Fig. 2 shows three common metapaths on citation networks, named: *apa*, *apcpa*, and *apapa*.

Definition 3 (*Metapath Induced Subgraph*)**.** Given a heterogeneous graph $G = (V, E)$, we define the metapath induced subgraph G_P as a subgraph of G. All its nodes are of the same type, and nodes are selected based on metapath $P : v_1 \xrightarrow{r_1} v_2 \xrightarrow{r_2} \cdots \xrightarrow{r_l} v_{l+1}$. In other words, the "metapath induced subgraph" is a homogeneous graph that is extracted from G according to a specific metapath.

Taking the DBLP network in Fig. 1(a) as an example, two metapaths $\{apa, apcpa\}$ are selected to construct the metapath induced subgraphs. The results are shown in the right part of Fig. 3. It can be seen that the metapath induced subgraphs maintain different semantics well.

Definition 4 (*Semantic Intensity*)**.** Semantic intensity E_{ij} represents the importance of node v_i to node v_j on the metapath induced subgraph. Since the adjacency matrix in the metapath induced subgraph is symmetric, the semantic intensity is the same for the same pair nodes, that is, $E_{ij} = E_{ji}$. The semantic intensity E_{ij} is determined by the number of metapaths between the target node v_i and its neighbor node v_j. The more the number of metapaths passed between a pair of nodes, the stronger their semantic intensity is which implies a strong relevance.

For instance, Fig. 3 is a subgraph generated via *apa* metapath. As $Author_2$ and $Author_3$ collaborate on $Paper_1$ and $Paper_3$, their semantic intensity is defined as $E_{23} = E_{32} = 2$.

With the above definitions, we formalize the problem of heterogeneous graph embedding as follows.

Problem (*Heterogeneous Graph Embedding*)**.**. Given a heterogeneous graph $G = (V, E)$ as input, the goal is to learn a mapping function $f : v \rightarrow z_v \in \mathbb{R}^d$, where z_v is a low-dimensional dense vector learned for node v, and d is the dimension of the learned embedding, $d \ll |V|$. The function f aims to preserve both the topological information and multi-typed features in the heterogeneous graph simultaneously.

4. The proposed method

In this section, we first propose the framework of the OSGNN, then present each component in detail. Finally, we propose the algorithm for our model.

The overall framework of the OSGNN is shown in Fig. 4. It mainly consists of two components: **Subgraph generation of target nodes**; **Node information propagation and embedding connection**.

Given a heterogeneous graph G and node features X as input, we first construct a subgraph G_S composed of target node types with the *subgraph generation of target nodes*. Then OSGNN learns the graph G

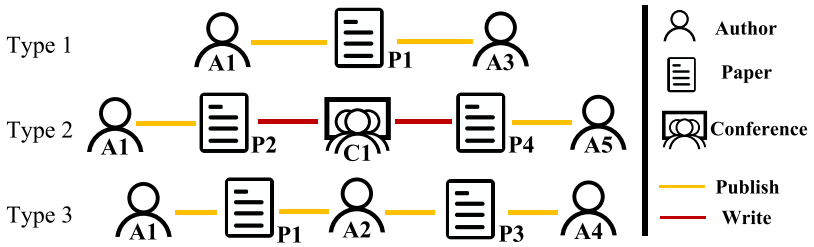

Fig. 2. An example of three types of metapaths.

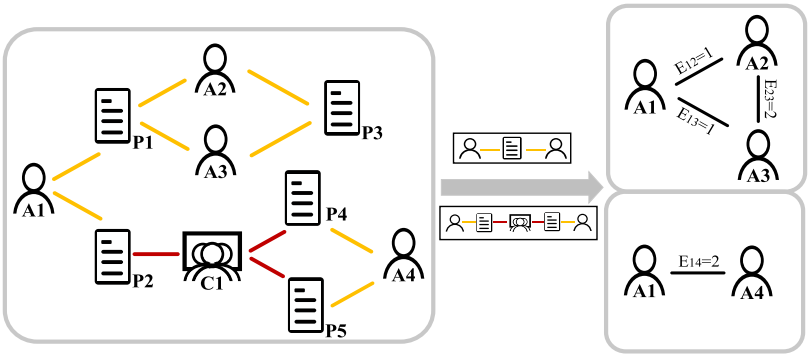

Fig. 3. The generation of metapath induced subgraph.

Table 1
The notations used in this paper.

Notations	Descriptions
G	The original heterogeneous graph.
V, E	The set of nodes/edges in network G.
G_{P_m}	The subgraph induced by the metapath P_m.
G_S	The subgraph as the input of the heterogeneous graph embedding.
h_{v_i}	The initial feature of node v_i.
α_{v_i,v_j}	Attention weight between node v_i and node v_j.
$h'_{v_i\|G}$	The hidden layer embedding learned by node v_i in graph G.
h''_{v_i}	The new representation of node v_i after concatenation.
γ	The hyper-parameter as initial feature weight.
z_{v_i}	The final representation of node v_i.
Y_L	The set of node index with labels.
Y_P	The set of positive node pairs.
Y_N	The set of negative node pairs.

and subgraph G_S respectively, and obtains two kinds of node representations Z_G and Z_{G_S} which maximally keep information from different aspects. The next step is to aggregate the above two kinds of node representations to obtain Z_{Agg}. Finally, the final node representation Z is learned by weighting and summed with Z_{Agg}. The details are presented in the following subsections.

4.1. Subgraph generation of target node

As shown in Fig. 4, OSGNN requires two graphs as input: The first one is the original graph G. The other is the G_S combined with multiple metapath induced subgraphs $\{G_{P_1}, G_{P_2}, \cdots, G_{P_M}\}$. Assuming that a heterogeneous graph G has M types of metapaths $\{P_1, P_2, \ldots, P_M\}$, the set of metapath induced subgraphs are denoted by $\{G_{P_1}, G_{P_2}, \cdots, G_{P_M}\}$. Its equation can be expressed as:

$$G_{P_m} = f_{extract}(G|P_m), \tag{1}$$

where G is the original heterogeneous graph, P_m is the mth metapath, $f_{extract}(\cdot|\cdot)$ is the subgraph extraction function. To preserve the semantic intensity in each metapath induced subgraph, we use the matrix multiplication of the relational subgraph to obtain the adjacency matrix A_{P_m} of the metapath induced subgraph. Taking metapath *apa* as an example, its adjacency matrix $A_{P_{apa}}$ can be expressed as:

$$A_{P_{apa}} = A_{P_{ap}} \cdot A_{P_{pa}}. \tag{2}$$

The above subgraphs are fused to capture the potential interaction information between these semantics. A very simple method is to add these subgraphs directly. However, it ignores the importance of different metapaths. In this section, we introduce a graph-level attention to get the importance of each metapath induced subgraph, and then fuse them as a new graph.

$$A_S = \Psi([A_{P_1}, A_{P_2}, \ldots, A_{P_M}]), \tag{3}$$

where Ψ is a graph-level attention layer with parameters $W_\Psi \in \mathbb{R}^{1\times1\times M}$ represents the importance of different metapath induced subgraphs. A_S is the adjacency matrix of the target node subgraph. To facilitate the training of the model, we use L_2 normalization to constrain A_S:

$$\widetilde{A} = \|A_S\|_2. \tag{4}$$

In addition, subgraph generation can expand and integrate various types of metapath induced subgraphs. We have expanded more types of metapaths in ablation study.

4.2. Node information propagation

It is a fact that different types of nodes contain different feature information and feature dimensions. Therefore, it is necessary to map different types of features into the same feature space with transformation projection. Heterogeneous feature transformation can be described as:

$$h_{v_i} = W_A \cdot x_{v_i} \tag{5}$$

where W_A is the learnable weight matrix and x_{v_i} is the initial feature of the node v_i.

In the original heterogeneous graph, OSGNN aims to capture the information of local heterogeneous nodes, so node-level attention (Wang et al., 2019) is able to effectively accomplish this task.

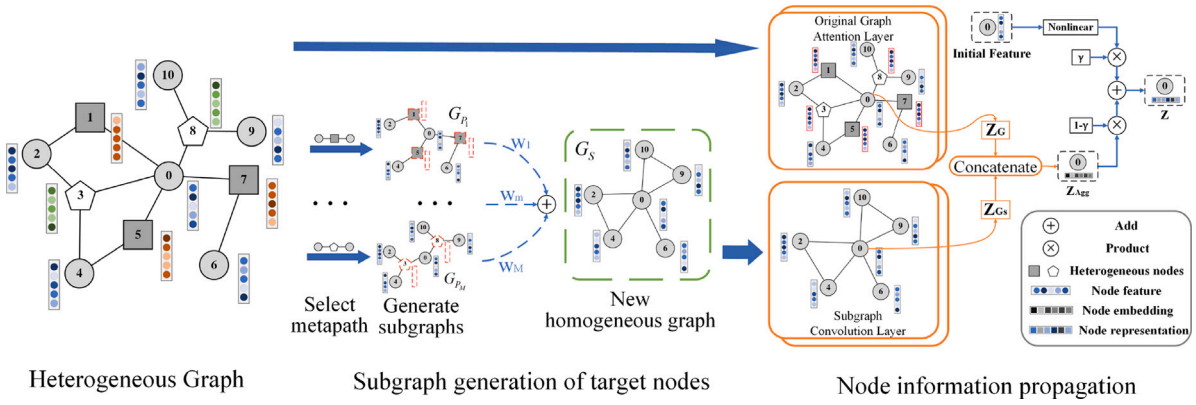

Fig. 4. The framework of the OSGNN model.

Given a node pair (v_i, v_j), we learn the node-level importance e_{v_i,v_j} of node $v_j \in N_{v_i}$ to target node v_i with the self-attention mechanism (Velickovic et al., 2018), and its equation is expressed as follows:

$$e_{v_i,v_j} = \sigma(p^T[W_\alpha h_{v_i} \parallel W_\alpha h_{v_j}]), \tag{6}$$

where $\sigma(\cdot)$ indicates the activation function, $\parallel$ is the row-wise concatenation operation, $h_{v_i} \in \mathbb{R}^{d_0}$ is the initial feature of node v_i, $W_\alpha \in \mathbb{R}^{d \times d_0}$ is the linear transformation matrix of node features, d denotes the transformed dimension, $p \in \mathbb{R}^{2d}$ is the parameterized attention vector.

With the importance of each node to the target node v_i, we use $softmax$ for normalization (Velickovic et al., 2018):

$$\alpha_{v_i,v_j} = softmax(e_{v_i,v_j}) = \frac{exp(e_{v_i,v_j})}{\sum_{v_n \in N_{v_i}} exp(e_{v_i,v_n})}, \tag{7}$$

$$h'_{v_i|G} = \sigma\Big(\sum_{v_j \in N^G_{v_i}} \alpha_{v_i,v_j} \cdot W_a h_{v_j}\Big). \tag{8}$$

Therefore, the target node representation of the original graph G is expressed as $Z_G = \{h'_{v_1|G}, h'_{v_2|G}, \dots, h'_{v_n|G}\}$.

For the target node subgraph A_S, it is regarded as a weighted graph to learn the node representation. When generating metapath induced subgraphs, the semantic intensity under each pair of nodes is preserved and used as the weight between nodes. Specifically, we put A_S into GCN (Welling & Kipf, 2017) for learning:

$$h'_{v_i|G_S} = \sigma\Big(\sum_{v_j \in N^{G_S}_{v_i}} \widetilde{A}_{v_i,v_j} \cdot W_c h_{v_j}\Big), \tag{9}$$

where $N^{G_S}_{v_i}$ denotes the neighbors of node v_i under subgraph G_S. The node representation of the subgraph G_S is $Z_{G_S} = \{h'_{v_1|G_S}, h'_{v_2|G_S}, \dots, h'_{v_n|G_S}\}$.

Since $h'_{v_i|G}$ and $h'_{v_i|G_S}$ learn heterogeneous graph information from different perspectives, aggregating these two kinds of embeddings with *mean* or *attention* (Wang et al., 2019) leads to loss of information independence (Yang et al., 2022). Therefore, we adopt a simple but effective operation named concatenation to preserve the independence of two kinds of information (He et al., 2022):

$$h''_{v_i} = h'_{v_i|G} \parallel h'_{v_i|G_S}, \tag{10}$$

where $\parallel$ stands for the concatenation operation, which can overcome the interference between different graphs and maintain the information independence of the two graphs. After that, the node representation is denoted as $Z_{Agg} = \{h''_{v_1}, h''_{v_2}, \cdots, h''_{v_n}\}$.

4.3. Retention of initial node feature

In the end, we reconnect Z_{Agg} to the h_{v_i} to further enhance the representation ability. We first transform the initial features to the same dimension as Z_{Agg} with nonlinear transformation, and then conduct weighted summation with the node representation Z_{Agg}:

$$z_{v_i} = (1-\gamma) \cdot h''_{v_i} + \gamma \cdot tanh(W_\gamma h_{v_i} + b_\gamma), \tag{11}$$

where γ is an adjustable hyperparameter, the interval is set to [0,1], W_γ and b_γ are trainable weight and bias, and the final node representation set can be expressed as $Z = \{z_{v_1}, z_{v_2}, \dots, z_{v_n}\}$.

4.4. Loss function and algorithm

We evaluate the OSGNN with different downstream tasks. For semi-supervised node classification, we minimize the cross entropy of the real and predicted values of all labeled nodes and apply back propagation and gradient descent methods to optimize the parameters of all nodes:

$$Loss = -\sum_{l \in Y_L} Y_{v_l} \cdot ln(C \cdot z_{v_l}), \tag{12}$$

where Y_L is a set of node index with labels, C is the classifier parameter, Y_{v_l} and z_{v_l} are the labels and embedding vector of node v_l, respectively.

For link prediction, we optimize the model weights by minimizing similarity between positive links:

$$Loss = -\sum_{(v,u) \in Y_P} \log \sigma(h_v^\top \cdot h_u) - \sum_{(v,w) \in Y_N} \log \sigma(-h_v^\top \cdot h_w) \tag{13}$$

where $\sigma(\cdot)$ is the sigmoid function, Y_P is the set of positive node pairs, Y_N is the set of negative node pairs sampled from all unobserved node pairs.

The overall process of OSGNN is shown in Algorithm 1. In terms of time complexity, OSGNN needs to calculate the time consumption of three processes. The time complexity of feature transformation is $\mathcal{O}(|V| \times d_1 d_2)$, where d_1 represents the initial dimension of the feature, d_2 represents the dimension of hidden layer. In the original heterogeneous graph, the time complexity of information propagation is $\mathcal{O}(|V| \times {d_2}^2 + |E| \times d_2)$. In the subgraph, the time complexity of information propagation is $\mathcal{O}(|V_S| \times {d_2}^2)$, where V_S is the node set of the subgraph G_S. OSGNN is highly efficient because it can perform parallel computation on the original graph and subgraph. Therefore, the time complexity of OSGNN is $\mathcal{O}(|V|(d_1 d_2 + {d_2}^2) + |E| d_2)$.

5. Experiments

In this section, we conduct experiments and demonstrate the results to prove the performance of this work. In particular, we show the merits of our method with node classification, node clustering, visualization, and link prediction. The source code of our method is available on https://github.com/ZZY-GraphMiningLab/OSGNN.

Algorithm 1 The overall learning algorithm of OSGNN

Input: the heterogeneous graph $G = (V, E)$, the node feature $X = \{x_{v_1}, x_{v_2}, \cdots, x_{v_i}\}$, the metapath set $\{P_1, P_2, \cdots, P_M\}$, the adjustable hyper-parameter γ.
Output: the final node representation Z.

for $P_i \in \{P_1, P_2, \cdots, P_M\}$ **do**
  Generate the metapath induced subgraph G_{P_m};
end for
Merge subgraphs to generate G_S according to Eq. (3);
for $v_i \in V$ **do**
  Find the node neighbors $N_{v_i}^G$;
  for $v_j \in N_{v_i}^G$ **do**
    Calculate the importance of node pair α_{v_i,v_j} according to Eq. (6);
  end for
  Calculate the graph G embedding $h'_{v_i|G}$ according to Eq. (8);
  Find the node neighbors $N_{v_i}^{G_S}$;
  for $v_j \in N_{v_i}^{G_S}$ **do**
    Calculate the graph G_S embedding $h'_{v_i|G_S}$ according to Eq. (9);
  end for
end for
Aggregate the embedding of the original graph and subgraph according to Eq. (10);
Weighted calculation with node representation according to Eq. (11);
Return Z.

5.1. Experimental datasets

Three real-world heterogeneous graph datasets are used to evaluate our model. The statistics of the datasets are described in Table 2.

- **ACM**[1]: This is a subset of the ACM dataset, which contains 4019 papers (p), 7167 authors (a) and 60 conference subjects (s). The papers belong to three categories: database, wireless communication, and data mining. The feature of each node is a word bag composed of keywords. $\{psp, pap\}$ is selected to generate metapath induced subgraphs.
- **DBLP**[2]: This is a subset of the DBLP dataset, which contains 14,328 papers (p), 4057 authors (a), 7723 terms (t) and 20 conferences (c). The author can be classified into four fields: database, data mining, machine learning, and information retrieval. The authors' feature is a word bag composed of the author's relevant paper keywords. $\{apa, aptpa, apcpa\}$ is selected to generate metapath induced subgraphs.
- **IMDB**[3]: It is a movie network consisting of 4780 movies (m), 5841 actors (a) and 2269 directors (d). The movies are considered as the target nodes and they are categorized as: action, comedy, and drama. $\{mam, mdm\}$ is selected to generate metapath induced subgraphs.
- **Yelp**[4]: It is a social network consisting of 2614 business (b), 1286 users (u), 4 serves (s), and 9 levels (l). The business nodes are labeled by their category. $\{bub, blb, bsb\}$ is selected to generate metapath induced subgraphs.

5.2. Baselines

We compare OSGNN with the state-of-the-art models, including two representative homogeneous graph neural network models and six representative heterogeneous graph neural network models:

- **GCN** (Welling & Kipf, 2017): It is a semi-supervised graph convolution network designed for homogeneous graphs. It updates its own information by uniformly aggregating the information of its neighborhood.
- **GAT** (Velickovic et al., 2018): It is a GNN considering the attention mechanism, which can adjust the importance of neighbor nodes according to the attention mechanism. Like GCN, it is designed for homogeneous graphs.
- **HAN** (Wang et al., 2019): It applies graph attention network on multiplex network considering the inter- and intra-network interactions, which exploit manually selected metapaths to learn node embedding.
- **GTN** (Yun et al., 2019): It multiplies the adjacency matrix of multiple subgraphs to obtain the neighborhood graphs of different metapaths, so it can automatically find valuable metapaths without manual selection.
- **MAGNN** (Fu et al., 2020): It realizes learning heterogeneous graph representations with intra-metapath aggregation and inter-metapath aggregation. For simplicity, we denote it as MAGN.
- **ie-HGCN** (Yang et al., 2023): It contributes to metapath discovery of in heterogeneous graphs by specific type-level attention mechanism. For simplicity, we denote it as ieHG.
- **HGSL** (Zhao et al., 2021): It is a state-of-the-art heterogeneous GNN, which jointly performs heterogeneous graph structure learning and GNN parameter learning for node classification.
- **HMSG** (Cai et al., 2021): It decomposes the HG into multiple metapath-based subgraphs to comprehensively capture structural, semantic and feature information from multi-view.
- **HGNN-AC** (Jin et al., 2021): Heterogeneous graph neural network via attribute completion (HGNN-AC) completes the node features with the pre-trained topology and takes the new features as input to MAGNN to learn the node representation. For simplicity, we denote it as HGAC.

5.3. Experimental settings

To ensure the fairness of the experiment, the embedding dimension of the hidden layer is set to 64. The multi-head number is set to 8 for models (Fu et al., 2020; Jin et al., 2021; Velickovic et al., 2018; Wang et al., 2019) as suggest by references. For the HGNNs that require metapaths (Fu et al., 2020; Wang et al., 2019; Zhao et al., 2021) and our model OSGNN, we use the same metapath set $\{pap, psp\}$, $\{apa, aptpa, apcpa\}$, $\{mam, mdm\}$, and $\{bub, blb, bsb\}$.

The learning rate of all models is set to 0.001, the weight decay is set to 0.0005, the dropout is set to 0.5, and the patience value of early stopping is set to 20. All models are randomly initialized the parameters and used Adam (Kingma & Ba, 2015) optimization models.

5.4. Semi-supervised node classification

We evaluate the performance of OSGNN on the task of semi-supervised node classification. The target node representations in the test set are fed into the Support Vector Machine (SVM) (Suykens, 2001) for classification. All experiments are repeated five times, with the average results being the final outcome. Macro-F1 (Ma-F1) and Micro-F1 (Mi-F1) are adopted as evaluation indicators. The results are shown in Table 3, in which the best results are bold and the second best results are underlined.

From this table, we can find that OSGNN has achieved outstanding performance on the three datasets. Specifically, compared with HAN, OSGNN improves Ma-F1 and Mi-F1 scores by 2.02% and 1.98% on ACM dataset, 1.63% and 1.47% on DBLP dataset, 1.32% and 1.18% on IMDB dataset. The main reason is that OSGNN considers different types of nodes and learns the information between heterogeneous nodes, while HAN directly learns the interaction between target nodes with the metapath, ignoring the role of other types of nodes. Compared

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

Table 2
Statistics of the datasets.

Dataset	Nodes	Edges	Number	Target	Label	Task
ACM	11246	pa/ap ps/sp	13407 4019	p (4019)	3	node classification node clustering
DBLP	26128	ap/pa pc/pc pt/tp	19645 14328 85810	a (3257)	4	node classification node clustering
IMDB	11616	ma/am md/dm	12828 4278	m (4278)	3	node classification node clustering
Yelp	3913	bu/ub bl/lb bs/sb bb	30838 2614 2614 1905	bb (1905)	–	link prediction

Table 3
Performance evaluation on node classification (%).

Dataset	Metrics	Training	GCN	GAT	HAN	GTN	MAGN	ieHG	HGSL	HMSG	HGAC	OSGNN
ACM	Ma-F1	20%	90.51	89.09	90.71	90.88	90.02	91.35	92.43	91.69	91.62	**92.63**
		40%	90.68	89.32	91.33	91.36	91.39	92.14	92.58	92.41	93.05	**93.29**
		60%	90.80	89.43	91.73	91.74	92.18	92.59	92.73	92.66	93.63	**93.72**
		80%	90.58	89.33	91.91	91.81	92.67	92.79	92.83	92.81	93.79	**93.83**
	Mi-F1	20%	90.49	89.20	90.59	90.76	89.94	91.27	92.38	91.63	91.51	**92.55**
		40%	90.68	89.40	91.22	91.24	91.38	92.11	92.54	92.24	93.09	**93.14**
		60%	90.79	89.49	91.60	91.61	92.13	92.53	92.69	92.58	**93.65**	93.62
		80%	90.56	89.40	91.76	91.70	92.61	92.73	92.77	92.72	**93.87**	93.74
DBLP	Ma-F1	20%	89.04	89.76	92.63	93.32	92.93	92.73	93.72	93.07	**94.12**	94.00
		40%	89.05	89.75	92.87	94.06	93.32	93.57	93.65	93.43	94.25	**94.27**
		60%	89.01	89.77	93.05	94.15	93.69	93.66	93.81	93.71	94.27	**94.47**
		80%	89.17	89.83	93.16	94.26	94.01	94.09	94.09	93.96	94.38	**94.79**
	Mi-F1	20%	89.71	90.53	93.20	94.25	93.45	93.24	94.19	93.51	**94.53**	94.44
		40%	89.72	90.53	93.43	94.53	93.82	94.00	94.09	94.04	94.65	**94.69**
		60%	89.70	90.56	93.61	94.60	94.18	94.10	94.23	94.18	94.68	**94.88**
		80%	89.85	90.61	93.69	94.70	94.48	94.47	94.52	94.40	94.76	**95.16**
IMDB	Ma-F1	20%	49.03	58.60	58.11	57.26	57.87	58.24	58.16	**59.72**	59.67	59.52
		40%	49.15	58.67	58.56	57.90	59.23	59.33	58.04	60.13	60.18	**60.36**
		60%	49.71	58.78	58.73	58.04	59.72	59.65	58.19	60.42	60.60	**60.76**
		80%	41.94	58.66	58.88	58.84	59.94	59.87	58.76	60.57	60.75	**60.99**
	Mi-F1	20%	49.43	58.74	58.14	57.12	57.89	58.16	58.54	59.73	**59.84**	59.53
		40%	49.63	58.84	58.58	57.81	59.29	59.26	58.49	60.14	60.38	**60.40**
		60%	49.95	58.92	58.72	57.89	59.80	59.57	58.56	60.39	60.79	**60.80**
		80%	50.12	58.80	58.91	58.74	60.06	59.82	59.09	60.59	60.98	**61.03**

with HMSG, OSGNN improves Ma-F1 and Mi-F1 scores by 1.02% and 0.98% on ACM dataset, 0.19% and 0.21% on IMDB dataset. The outstanding performance of HMSG also proves the necessity of simultaneously learning local information and higher-order information in heterogeneous graphs. OSGNN exploits metapath induced subgraphs thus it improves the computational efficiency. Compared with HGNN-AC, OSGNN only improves Ma-F1 scores by 0.04% on ACM dataset, 0.41% on DBLP dataset, and 0.24% on IMDB dataset. This is because HGNN-AC completes the missing features, and excellent node features will help the model learn better node representation.

In general, OSGNN achieves the best performance. This implies that the information of nodes can effectively complement each other with learning from the two perspectives of subgraph and original graph, which overcomes the single perspective of the existing HGNNs. In addition, the performance of our model does not decrease significantly as the training ratio decreases on three datasets, which further shows that OSGNN is more stable than others.

5.5. Node clustering and visualization

The learned node embeddings are clustered with K-means with average normalized mutual information (NMI) and adjusted rand index (ARI) as metrics. The results are shown in Table 4, from which we get the following observations.

Compared with MAGNN, OSGNN improves NMI and ARI scores by 2.30% and 2.37% on ACM dataset, 2.95% and 2.47% on DBLP dataset, and 2.91% and 5.51% on IMDB dataset. It indicates that local information and higher-order information are helpful to improve the clustering performance. Compared with HMSG, OSGNN improves NMI and ARI scores by 1.99% and 1.74% on ACM dataset, 4.67% and 3.73% on DBLP dataset. All these results further proves the effectiveness of OSGNN.

Table 4
Performance of the proposed OSGNN and the competitors on the task of node clustering.

Dataset	ACM		DBLP		IMDB	
	NMI	ARI	NMI	ARI	NMI	ARI
HAN	0.7026	0.7415	0.7278	0.7833	0.1196	0.1197
GTN	0.6845	0.7195	0.7782	0.8269	0.1223	0.1258
MAGN	0.7016	0.7214	0.7867	0.8402	0.1308	0.1267
ieHG	0.5947	0.5489	0.5233	0.6721	0.1308	0.1304
HGSL	0.7025	0.7425	0.7763	0.8247	0.0621	0.0878
HMSG	0.7047	0.7277	0.7695	0.8276	0.1482	0.1514
HGAC	0.7176	0.6891	0.7880	0.8426	0.1368	0.1450
OSGNN	**0.7246**	**0.7451**	**0.8162**	**0.8649**	**0.1599**	**0.1818**

For a more intuitive comparison, we use t-SNE (Van der Maaten & Hinton, 2008) to project the node embeddings into two-dimensional space, and color them according to their labels. Fig. 5 shows the visualization results of all the methods on DBLP dataset. Specifically, although ie-HGCN performs well in node classification, the nodes of the same type are scattered and not dense enough in visualization.

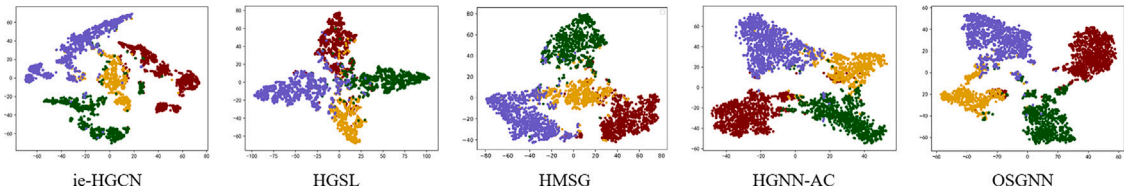

Fig. 5. Visualization of the author node embeddings on DBLP dataset.

Table 5
Performance of OSGNN on the task of link prediction.

Dataset	Metrics	HAN	GTN	MAGN	ieHG	HGSL	HMSG	HGAC	OSGNN
Yelp	AUC	0.6787	0.5961	0.7083	0.5558	0.5885	0.7169	0.7384	**0.7399**
	AP	0.6343	0.5727	0.6521	0.5246	0.5525	0.6657	0.6716	**0.6836**

The nodes of each type in HGSL are densely distributed, but their boundaries are not clear enough. Compared with OSGNN, HGNN-AC is the most competitive. We guess this is because HGNN-AC focuses on completing the missing node features. The heterogeneous graph after feature completion contains more useful information. However, OSGNN shows denser cluster structures and a clearer boundary to distinguish different classes.

5.6. Link prediction

In this section, we conduct the task on Yelp dataset to evaluate the performance of OSGNN and other baselines in link prediction. We treat the connected business–business (bb) pair as positive node pairs, and consider all unconnected business–business links as negative node pairs. The performance is evaluated by the area under the ROC curve (AUC) and average precision (AP) scores. The results are shown in Table 5, in which the best results are in bold.

From Table 5, the performance of metapath based models (HAN, MAGNN, HMSG, HGAC, and OSGNN) is significantly better than that of other models (GTN, ieHG, and HGSL). It indicates that higher-order information (metapath) is able to explore the potential similarity between nodes. Compare with HAN, OSGNN improves AUC and AP scores by 6.12% and 4.93%. Compare with HGAC, OSGNN improves AUC and AP scores by 0.15% and 1.20%. This result supports further proves that local information and higher-order information in HG are important to the node embeddings.

5.7. Ablation study

To examine the effect of different components of our OSGNN model, we compare several variants of it in an ablation study:

- OSGNN$_{-w/o-\gamma}$: It refers to a variant of OSGNN model that does not retain the initial features.
- OSGNN$_{-w/o-O}$: Original heterogeneous graph learning is removed from the OSGNN model. Thus, it learns representations merely from subgraphs.
- OSGNN$_{-w/o-S}$: Subgraphs learning is removed from the OSGNN model. It means that this model learns information from the original heterogeneous graph merely and ignores the information from the subgraphs.
- OSGNN$_{-w/o-Att}$: The mean aggregation is used in fusing metapath induced subgraphs in this model. Thus it is a variant that ignores the significance of different subgraphs.

The results are shown in Fig. 6. We can observe that the performances of the ablated models decrease unanimously, which demonstrates the utility of these components. Specifically, we can derive the following interesting conclusions from the results:

(1) For all datasets, the performance decreases obviously after the removal of subgraphs learning (OSGNN$_{-w/o-S}$). This phenomenon indicates that the subgraphs are helpful in learning the potential high-order information. In fact, the model HAN learns high-order interaction information between target nodes also performs well on three datasets. It further proves the importance of high-order information.

(2) The performance decreases when the information of the original graph is removed from OSGNN (OSGNN$_{-w/o-O}$), especially on the ACM dataset. This indicates that local information also plays an essential role in heterogeneous graph learning. GCN and GAT are such models that only learn the local information of nodes, and have nevertheless achieved good results on the ACM dataset.

(3) Among all variants, OSGNN$_{-w/o-\gamma}$ achieves the best performance. A possible reason is that the learning of the original graph is able to effectively preserve the information of the nodes themselves. We will analyze this parameter in detail in Section 5.9.

In addition, we add three high-order metapaths to verify the necessity of high-order information on ACM and IMDB datasets:

- OSGNN-2: It is a variant of OSGNN model that uses two common metapaths. Two metapaths *pap* and *psp* are selected on ACM dataset, and *mam* and *mdm* are selected on IMDB dataset.
- OSGNN-3: It is a variant of the OSGNN model that uses three metapaths. $\{pap, psp, papsp\}$ are selected to generate the subgraph on ACM dataset, and $\{mam, mdm, mamdm\}$ are selected on IMDB dataset.
- OSGNN-4$_1$: It is a variant of the OSGNN model that uses four metapaths. We choose metapath set {*pap*, *psp*, *papsp*, *pspsp*} and metapath set {*mam*, *mdm*, *mamdm*, *mamam*} on ACM and IMDB datasets respectively.
- OSGNN-4$_2$: It is a variant of the OSGNN model that uses four metapaths. We choose metapath set {*pap*, *psp*, *papsp*, *papap*} and metapath set {*mam*, *mdm*, *mamdm*, *mdmdm*} on ACM and IMDB datasets respectively.
- OSGNN-5: It is a variant of the OSGNN model that uses five kinds of metapaths. We choose metapath set {*pap*, *psp*, *papsp*, *papap*, *pspsp*} and metapath set {*mam*, *mdm*, *mamdm*, *mdmdm*, *mamam*} on ACM and IMDB datasets respectively.

The results are shown in Table 6. It can be observed that the higher-order metapaths has further improved the performance. On ACM dataset, OSGNN-5 has achieved outstanding performance. It is because that subgraph G_S is able to fuse more effective interactive information, with the help of more high-order metapaths. However, metapath based models such as HAN (Wang et al., 2019) and MAGNN (Fu et al., 2020) cannot effectively extend and integrate the information in multiple high-order metapaths. On IMDB dataset, OSGNN-3 and OSGNN-4$_2$ achieved similar performances. It implies that different kinds of metapaths contain different information, and the noises may be amplified on some metapaths such as *mamam*. In a word, the addition of high-order metapaths can indeed improve the performance of OSGNN, but the selection of metapaths is still a problem worth considering.

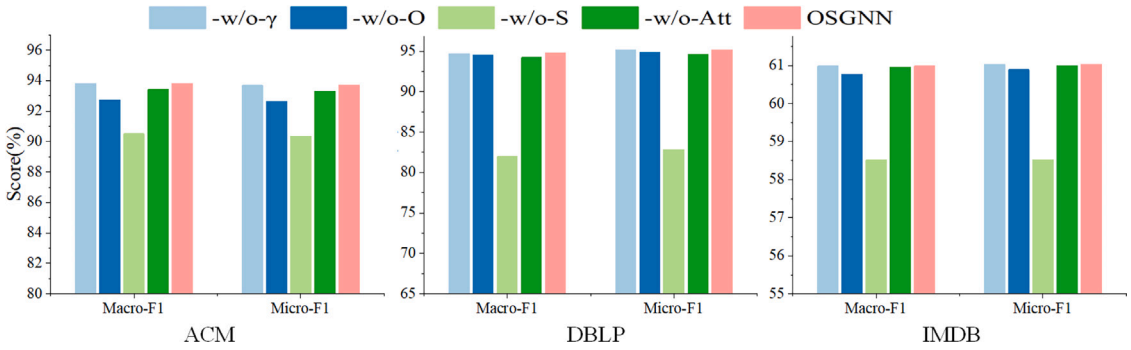

Fig. 6. The results of ablation study.

Table 6
The results of OSGNN under different metapaths (%).

Datasets	Metrics	Training	OSGNN-2	OSGNN-3	OSGNN-4_1	OSGNN-4_2	OSGNN-5
ACM	Ma-F1	20%	92.63	92.71	92.31	92.98	**93.02**
		40%	93.29	93.39	92.93	93.62	**93.66**
		60%	93.72	93.83	93.40	94.01	**94.07**
		80%	93.83	93.86	93.50	94.09	**94.10**
	Mi-F1	20%	92.55	92.58	92.19	92.86	**92.88**
		40%	93.14	93.27	92.84	93.53	**93.54**
		60%	93.62	93.70	93.29	93.91	**93.95**
		80%	93.74	93.72	93.40	**93.99**	93.97
IMDB	Ma-F1	20%	59.52	**60.00**	59.68	59.97	59.95
		40%	60.36	**60.78**	60.46	60.71	60.68
		60%	60.76	61.01	60.70	**61.15**	61.00
		80%	60.99	61.12	60.77	**61.33**	61.06
	Mi-F1	20%	59.53	**60.90**	59.79	60.02	60.02
		40%	60.40	**60.90**	60.62	60.79	60.79
		60%	60.80	61.12	60.86	**61.25**	61.11
		80%	61.03	61.23	60.94	**61.43**	61.18

5.8. Importance study of graph-level attention based fusion

In this section, we visualize the attention coefficient allocated by graph-level attention when fusing subgraphs. As shown in Fig. 7, OSGNN assigns different weights to different subgraphs. On ACM datasets, OSGNN assigns nearly the same weight to each metapath induced subgraph. However, the subgraph under *pap* metapath is more important because different papers written by the same author have more similar research types. On DBLP datasets, OSGNN assigns higher weights to the subgraphs under the *apcpa* metapath, which can be explained because the labels of the author nodes are divided according to the conference type. On IMDB dataset, the subgraph under *mdm* metapath is more important than *mam*. As an actor tends to play different roles in different movies, the styles of the two movies connected by *mam* may be different. However, various movies made by the same director tend to present similar styles.

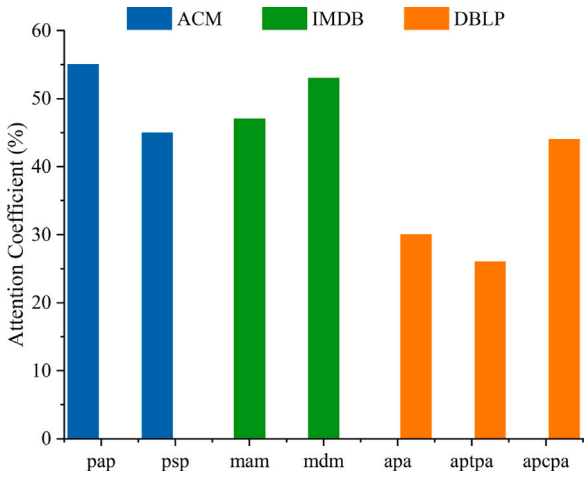

Fig. 7. The results of graph-level attention.

5.9. Parameter analysis

In this section, we demonstrate and analyze the impact of parameter γ on OSGNN results as shown in Table 7. The scores of Mi-F1 first increase and then decrease with the growth of γ. This phenomenon indicates that preserving initial features can further alleviate semantic confusion. On IMDB dataset, OSGNN gets the best performance when the parameter γ is set to 0. This is because the original features on IMDB dataset may be noisy, and preserving the original features will add redundant information to the node representation. In addition, when the parameter is changed to 1, the result of OSGNN decreases significantly. This is because OSGNN only learns node feature information and completely loses the graph structure information.

5.10. Complexity analyses

In this section, we visualize the time and memory consumption of OSGNN and baselines on the ACM dataset. The visualization results are shown in Fig. 8. The horizontal axis represents the average time required for the model to train an epoch, and the vertical axis represents the memory required for the training model. All experiments are conducted on a Windows computer with an Intel Core i7-10875H CPU.

From Fig. 8, we have the following observations. Among all models, ie-HGCN has the smallest time and memory consumption. This is because ie-HGCN discards the calculation of attention coefficient between

Table 7
Parameter Analysis.

γ	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
ACM	93.71	**93.74**	93.55	93.63	93.42	93.43	92.98	92.68	92.34	90.84	87.86
IMDB	**61.03**	60.93	60.36	60.21	60.07	59.75	59.26	59.29	58.68	57.28	51.17
DBLP	94.58	94.67	**95.16**	95.09	94.70	94.81	94.56	94.27	94.26	92.62	78.91

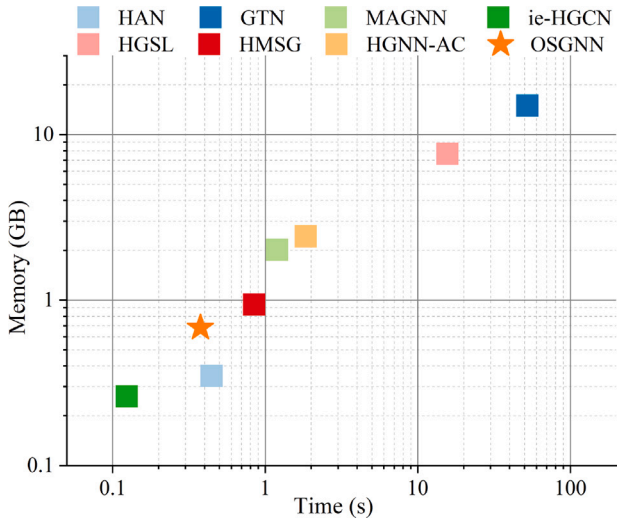

Fig. 8. Comparison of computational complexity between OSGNN and baseline models.

nodes, which greatly saves the computational overhead. Meanwhile, OSGNN performs the most competitive in all metapath based models, because subgraph fusion can fuse multiple metapath induced subgraphs into a more informative graph.

6. Conclusion

In this paper, we first highlight two challenges faced by heterogeneous graph neural networks, and then propose the OSGNN model to resolve these two challenges. The proposed OSGNN is able to learn both local information from the original heterogeneous graph and high-order information from the new constructed meaningful graph. Extensive experiments are conducted on real-world datasets and the results prove that the proposed OSGNN outperforms the state-of-the-art methods.

In addition, the baseline (Jin et al., 2021) proves the advantage of feature completion. In the future, we will focus on designing HGNN model by optimizing the graph structure and node features together.

CRediT authorship contribution statement

Yeyu Yan: Investigation, Writing – original draft, Data curation. **Chao Li:** Methodology, Writing – review & editing, Supervision, Funding acquisition. **Yanwei Yu:** Writing – review & editing, Investigation. **Xiangju Li:** Conceptualization, Investigation. **Zhongying Zhao:** Methodology, Writing – review & editing, Supervision, Funding acquisition.

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.

Data availability

I have share the codes in the abstract via a link.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant No. 62072288, 61702306, 61433012), the Taishan Scholar Program of Shandong Province, Shandong Youth Innovation Team, the Natural Science Foundation of Shandong Province (Grant No. ZR2018BF013, ZR2022MF268).

References

Andreoletti, D., Troia, S., Musumeci, F., Giordano, S., Maier, G., & Tornatore, M. (2016). Network traffic prediction based on diffusion convolutional recurrent neural networks. In *Proceedings of the IEEE INFOCOM 2019-IEEE conference on computer communications workshops* (pp. 246–251).

Atwood, J., & Towsley, D. (2016). Diffusion-convolutional neural networks. In *Proceedings of the neural information processing systems* (pp. 1993–2001).

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2014). Spectral networks and deep locally connected networks on graphs. In *Proceedings of the 2nd international conference on learning representations*.

Cai, X., Shang, J., Hao, F., Liu, D., & Zheng, L. (2021). HMSG: Heterogeneous graph neural network based on metapath subgraph learning. arXiv preprint arXiv:2109.02868.

Chang, Y., Chen, C., Hu, W., Zheng, Z., Zhou, X., & Chen, S. (2022). MEGNN: Meta-path extracted graph neural network for heterogeneous graph representation learning. *Knowledge-Based Systems*, *235*, Article 107611.

Chen, K., Lu, H., Liu, Z., & Zhang, J. (2022). Heterogeneous graph convolutional network with local influence. *Knowledge-Based Systems*, *236*, Article 107699.

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of neural information processing systems* (pp. 3844–3852).

Dong, X., Zhang, Y., Pang, K., Chen, F., & Lu, M. (2022). Heterogeneous graph neural networks with denoising for graph embeddings. *Knowledge-Based Systems*, *238*, Article 107899.

Fu, X., Zhang, J., Meng, Z., & King, I. (2020). MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the web conference 2020* (pp. 2331–2341).

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *Proceedings of the 34th international conference on machine learning. Vol. 70* (pp. 1263–1272).

Hamilton, W. L., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the neural information processing systems* (pp. 1024–1034).

He, Y., Yan, D., Zhang, Y., He, Q., & Yang, Y. (2022). Semantic tradeoff for heterogeneous graph embedding. *IEEE Transactions on Computational Social Systems*, 1–14.

Hong, H., Guo, H., Lin, Y., Yang, X., Li, Z., & Ye, J. (2020). An attention-based graph neural network for heterogeneous structural learning. In *Proceedings of the 34th AAAI conference on artificial intelligence* (pp. 4132–4139).

Ji, H., Wang, X., Shi, C., Wang, B., & Yu, P. S. (2023). Heterogeneous graph propagation network. *IEEE Transactions on Knowledge and Data Engineering*, *35*(1), 521–532.

Jin, D., Huo, C., Liang, C., & Yang, L. (2021). Heterogeneous graph neural network via attribute completion. In *Proceedings of the web conference 2021* (pp. 391–400).

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd international conference on learning representations*.

Li, J., Peng, H., Cao, Y., Dou, Y., Zhang, H., Yu, P. S., & He, L. (2023). Higher-order attribute-enhancing heterogeneous graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, *35*(1), 560–574.

Li, C., Peng, H., Li, J., Sun, L., Lyu, L., Wang, L., Yu, P. S., & He, L. (2022). Joint stance and rumor detection in hierarchical heterogeneous graph. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(6), 2530–2542.

Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., & Bronstein, M. M. (2017). Geometric deep learning on graphs and manifolds using mixture model CNNs. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5425–5434).

Moscato, V., & Sperlì, G. (2021). A survey about community detection over On-line Social and Heterogeneous Information Networks. *Knowledge-Based Systems*, *224*, Article 107112.

Qian, L., Wang, J., Lin, H., Xu, B., & Yang, L. (2022). Heterogeneous information network embedding based on multiperspective metapath for question routing. *Knowledge-Based Systems*, *240*, Article 107842.

Salamat, A., Luo, X., & Jafari, A. (2021). HeteroGraphRec: A heterogeneous graph-based neural networks for social recommendations. *Knowledge-Based Systems*, *217*, Article 106817.

Sun, Y., & Han, J. (2012). Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery*, *3*(2), 1–159.

Sun, Y., Han, J., Yan, X., Yu, P. S., & Wu, T. (2011). Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, *4*(11), 992–1003.

Suykens, J. A. (2001). Support vector machines: a nonlinear modelling and control perspective. *European Journal of Control*, *7*(2–3), 311–327.

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11), 2579–2605.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. In *Proceedings of the 6th international conference on learning representations*.

Wang, X., Ji, H., Shi, C., Wang, B., Cui, P., Yu, P., & Ye, Y. (2019). Heterogeneous graph attention network. In *Proceedings of the world wide web conference* (pp. 2022–2032).

Wang, C., Zhou, S., Yu, K., Chen, D., Li, B., Feng, Y., & Chen, C. (2022). Collaborative knowledge distillation for heterogeneous information network embedding. In *Proceedings of the ACM web conference 2022* (pp. 1631–1639).

Wang, H., Zhou, K., Zhao, X., Wang, J., & Wen, J.-R. (2023). Curriculum pre-training heterogeneous subgraph transformer for top-N recommendation. *ACM Transactions on Information Systems*, *41*(1), 1–28.

Welling, M., & Kipf, T. N. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th international conference on learning representations*.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24.

Xie, F., Zheng, A., Chen, L., & Zheng, Z. (2021). Attentive Meta-graph Embedding for item Recommendation in heterogeneous information networks. *Knowledge-Based Systems*, *211*, Article 106524.

Yang, Y., Guan, Z., Li, J., Zhao, W., Cui, J., & Wang, Q. (2023). Interpretable and efficient heterogeneous graph convolutional network. *IEEE Transactions on Knowledge and Data Engineering*, *35*(2), 1637–1650.

Yang, C., Pal, A., Zhai, A., Pancha, N., Han, J., Rosenberg, C., & Leskovec, J. (2020). Multisage: Empowering GCN with Contextualized Multi-embeddings on Web-scale Multipartite Networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2434–2443).

Yang, L., Zhou, W., Peng, W., Niu, B., Gu, J., Wang, C., Cao, X., & He, D. (2022). Graph neural networks beyond compromise between attribute and topology. In *Proceedings of the ACM web conference 2022* (pp. 1127–1135).

Yun, S., Jeong, M., Kim, R., Kang, J., & Kim, H. J. (2019). Graph transformer networks. In *Proceedings of the neural information processing systems* (pp. 11960–11970).

Zhang, J., Shi, X., Xie, J., Ma, H., King, I., & Yeung, D.-Y. (2018). GaAN: Gated attention networks for learning on large and spatiotemporal graphs. In *Proceedings of the 34th conference on uncertainty in artificial intelligence* (pp. 339–349).

Zhao, J., Wang, X., Shi, C., Hu, B., Song, G., & Ye, Y. (2021). Heterogeneous graph structure learning for graph neural networks. In *Proceedings of the 35th AAAI conference on artificial intelligence* (pp. 4697–4705).