TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 15/31 pp198–214
DOI: 10.26599/TST.2023.9010147
Volume 30, Number 1, February 2025

Restage: Relation Structure-Aware Hierarchical Heterogeneous Graph Embedding

Huanjing Zhao†, Pinde Rui†, Jie Chen, Shu Zhao*, and Yanping Zhang

Abstract: Heterogeneous graphs contain multiple types of entities and relations, which are capable of modeling complex interactions. Embedding on heterogeneous graphs has become an essential tool for analyzing and understanding such graphs. Although these meticulously designed methods make progress, they are limited by model design and computational resources, making it difficult to scale to large-scale heterogeneous graph data and hindering the application and promotion of these methods. In this paper, we propose Restage, a relation structure-aware hierarchical heterogeneous graph embedding framework. Under this framework, embedding only a smaller-scale graph with existing graph representation learning methods is sufficient to obtain node representations on the original heterogeneous graph. We consider two types of relation structures in heterogeneous graphs: interaction relations and affiliation relations. Firstly, we design a relation structure-aware coarsening method to successively coarsen the original graph to the top-level layer, resulting in a smaller-scale graph. Secondly, we allow any unsupervised representation learning methods to obtain node embeddings on the top-level graph. Finally, we design a relation structure-aware refinement method to successively refine the node embeddings from the top-level graph back to the original graph, obtaining node embeddings on the original graph. Experimental results on three public heterogeneous graph datasets demonstrate the enhanced scalability of representation learning methods by the proposed Restage. On another large-scale graph, the speed of existing representation learning methods is increased by up to eighteen times at most.

Key words: heterogeneous graph; graph embedding; relation structure; hierarchical

1 Introduction

Heterogeneous graphs, characterized by their multiple types of entities (nodes) and relations (edges), excel at modeling intricate real-world systems[1, 2]. Performing embedding (or representation learning) on these graphs has emerged as a vital tool for analyzing their structures and semantics, finding applications in diverse domains such as social media[3] and e-commerce[4, 5]. In heterogeneous graphs, the majority of representation learning methods adopt meticulously designed strategies to explore potential relationships between nodes. To preserve these potential relationships, researchers typically utilize shallow models and graph neural network models. However, in large-scale graphs, shallow models are associated with a higher time expense due to their sampling methods, while deep models are constrained by computational

• Huanjing Zhao, Pinde Rui, Jie Chen, Shu Zhao, and Yanping Zhang are with School of Computer Science and Technology, Anhui University, and also with Information Materials and Intelligent Sensing Laboratory of Anhui Province, Hefei 230601, China. E-mail: hwangyeong@163.com; rpdahu@163.com; chenjie200398@163.com; zhaoshuzs2002@hotmail.com; zhangyp2@gmail.com.

† These authors contribute equally to this work.

* To whom correspondence should be addressed.

Manuscript received: 2023-08-30; revised: 2023-11-14; accepted: 2023-12-06

resources. This hinders the widespread adoption of these meticulously designed methods.

Recently, in the field of homogeneous graph embedding, some studies attempt to address this problem. MILE[6] innovatively proposes a hierarchical graph embedding framework. As depicted in Fig. 1, this framework consists of three components. The coarsening process progressively diminishes the scale of the original graph, layer by layer, until it reaches the top level, which is notably smaller in size compared to the original graph. Next, any unsupervised graph representation method can be employed to learn node embeddings in the top-level graph. Finally, the refinement process transfers the learned node embeddings across layers, utilizing the correspondence between nodes in different layers, until the embeddings are refined back to the original graph. As node embeddings only need to be learned at the top-level, where the graph's scale is even close to the magnitude of common datasets (such as Cora, PubMed, etc.) used in the majority of graph machine learning methods, this framework enables numerous meticulously designed graph representation learning methods to be extended to large-scale graphs. Building upon this, Graphzoom[7] and HANE[8] devise coarsening and refinement strategies that take into account node attributes, giving rise to a category of frameworks called hierarchical attribute graph representation learning methods.

Embedding methods on heterogeneous graphs confront an analogous problem. Our focus lies on hierarchical heterogeneous graph representation learning, which entails certain design challenges. Heterogeneous graphs, unlike homogeneous ones, possess distinct relation structures. These relation structures stem from the RHINEs[9–11], which identify two types of relation structures in heterogeneous graphs: interaction relations (IRs) and affiliation relations (ARs). IRs represent peer-to-peer structures, exemplified by connections between papers and authors in an academic graph. For instance, in an academic graph, multiple papers are often linked to multiple authors. In contrast, ARs imply one-centered-by-another structures, such as the association between products and brands in an e-commerce graph where multiple products belong to a single brand. Consequently, handling the relation structures during the coarsening and refinement processes respectively constitutes two challenges of this work.

To address the two aforementioned challenges, we separately consider graph coarsening and refinement strategies under the relation structures. The core of the coarsening is determining topologically similar nodes and merging them into super nodes. For IRs, different types of nodes usually exhibit a tendency to cluster. For instance, certain authors frequently collaborate and publish papers together, creating robust connections between these papers and authors, while maintaining weaker links with other authors and papers. We utilize the community detection method to quantify the similarity of nodes in IRs. For ARs, nodes centered around a single entity demonstrate similarity, such as

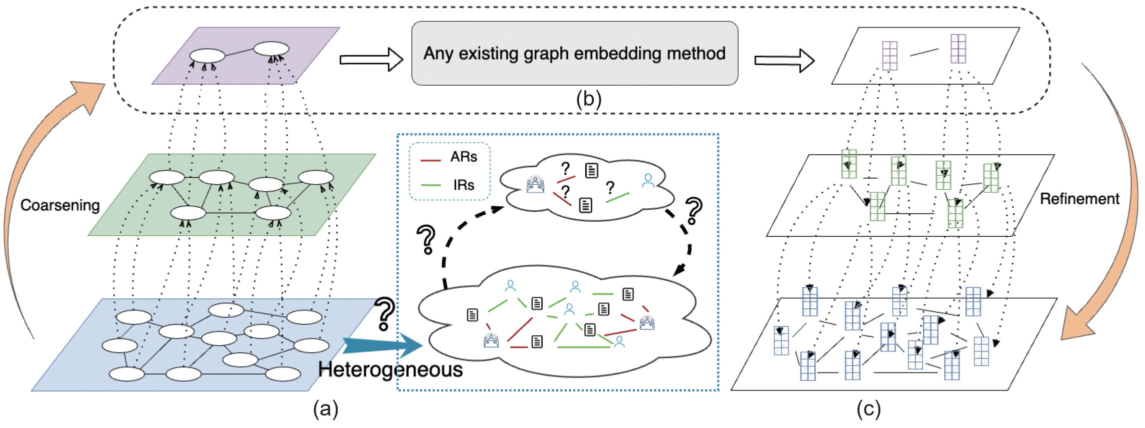

Fig. 1 Hierarchical graph representation learning framework: (a) Coarsening the original graph to the top-level graph. (b) Any existing graph representation method is applied to embed the top-level graph. (c) Refining the node embeddings from the top-level graph back to the original graph. This framework enhances the scalability of graph representation learning methods. However, in the case of heterogeneous graphs, due to the presence of two types of relation structures, interaction relations (IRs) and affiliation relations (ARs), it is not appropriate to consider both relations equally. This poses challenges for coarsening and refinement on heterogeneous graphs.

different products under a specific brand or papers published at an identical conference. We utilize second-order neighbors to identify similar nodes under ARs. Regarding the refinement strategy, we continue the strategy of MILE which using graph neural networks to reinforce the node representations within layers. To accommodate heterogeneous graphs, we additionally consider the information aggregation methods individually under the two relation structures.

In this paper, we propose Restage, a relation structure-aware hierarchical heterogeneous graph embedding framework, that effectively improves the scalability of current representation learning methods on heterogeneous graphs. First, by fully considering the similarity of nodes in IRs and ARs, we merge similar nodes to achieve a relation structure-aware graph coarsening method. Second, existing unsupervised graph representation methods are application to obtain node embeddings in the top-level graph. Finally, we design a relation structure-aware refinement method to transfer node embeddings of the top-level graph layer by layer back to the original graph. The contributions of our work are as follows:

- To address the coarsening problem of heterogeneous graphs, we propose a relation structure-aware graph coarsening method. For the two types of relation structures that exist in heterogeneous graphs, we separately consider the similarity of nodes under different relation structures during the graph coarsening process to identify and merge the most similar nodes into the next layer.
- To address the refinement problem of heterogeneous graphs, we propose a relation structure-aware graph refinement method. This method allows node embedding to be transferred across layers through the correspondence between nodes in different layers. Furthermore, we design different information propagation ways for the two relation structures to further reinforce the node embeddings at each level.
- The proposed Restage framework enhances the scalability of existing representation learning methods on heterogeneous graphs. Experiments on a large-scale heterogeneous graph show that, compared to directly applying representation learning methods to the original graph, the speed is improved by five to eighteen times within the proposed framework.

2 Related Work

Embedding methods on heterogeneous graph. Model design plays a critical role in representation learning on heterogeneous graphs, with the field predominantly divided into two primary classifications: shallow models[12–14] and deep models[15, 16]. In the initial stages, heterogeneous graph representation learning methodologies were chiefly shallow models, which sought to quantify node similarities through the design of various metric methods. The Metapath2vec[13], for instance, implements meta-paths for guidance in sampling while introducing the heterogeneous skip-gram to maximize same type node similarity within a sliding window. HHNE[17] migrates metapath2vec to the hyperbolic space, and the metric of nodes is replaced by the Poincare distance. HERec[18] takes into account nodes of a particular type during the sampling phase and introduces a weighted fusion method for different meta-path node embeddings. RHINE[9] delves into the relation structures found in heterogeneous graphs, proposing unique strategies for modeling nodes association within various relation structures. Shallow model-based methods establish a strong base for heterogeneous graph analysis. However, they suffer from the increased complexity brought about by sampling on large-scale graphs. These challenges make these models difficult to apply to extensive graphs and simultaneously drive the motivation for this work.

As graph neural networks evolve[19–21], the studies start to delve into the realm of convolution operations on heterogeneous graphs. RGCN[22] sets adaptable weight matrices for distinct relation types, thereby modeling diverse node attribute aggregation methods. On the other hand, HAN[16] advocates for two attention mechanisms that deal with heterogeneity by node-level and semantic-level view. GATNE[23] fully considers the node types and edge types and learns representations for each node in different edge types. MAGNN[24] not only considers the nodes at the ends of the meta-path in designing node-level aggregation but also includes the influence of nodes within the meta-path. NSHE[25] capitalizes on network schemes to retain the local information of nodes. Deep models become the prevailing approach at present, but due to computational resource constraints, they exhibit high complexity when operating on large-scale graphs. Recent advancements include a series of decoupled graph neural network methodologies[26–29] aim at augmenting the efficiency of representation learning, and they have found successful applications in large-scale graphs. Despite their innovative modeling, they cannot expand the current meticulously designed representation learning methods on heterogeneous graphs to a large-scale format. This limitation makes these methods difficult to apply in real-world scenarios.

Graph embedding with hierarchical approaches. Hierarchical structures offer multiple layers of perspective for analyzing node relationships, and their exploration aids in preserving the global information of the graph. The HARP[30] generates hierarchical structures via edge folding and employs Deepwalk[3] to extract embeddings at each layer, which are subsequently combined. Utilizing community detection algorithms, HSRL[31] applies progressive coarsening to construct hierarchies and deploys existing representation learning techniques at each level, integrating them through concatenation. LouvainNE[32] employs a top-down strategy for constructing hierarchical structures, identifying communities within each subdivided community to form a tree-like configuration. MILE[6], in contrast to the aforementioned methods, designs a hierarchical graph representation learning framework that reduces the original graph to a top-level graph of smaller scale using edge folding. The reduced graph is embedded to attain node representations within it, and a refinement method is developed to relay these representations back to the original graph, thereby enhancing the efficiency of graph representation learning. GraphZoom[7] and HANE[8] examine the hierarchical community structure in attribute graphs, thus forming hierarchical attribute graph representation frameworks. However, these hierarchical frameworks, designed for homogeneous graphs, are not readily adaptable to heterogeneous graphs. The existence of two types of relationship structures in heterogeneous graphs presents coarsening and refinement challenges, as these structures cannot be addressed uniformly.

3 Problem Statement

In this section, we delineate the problem to be addressed, furnish essential definitions, and offer the necessary background knowledge to facilitate a comprehensive understanding of this work.

Definition 1 Heterogeneous graph. A heterogeneous graph is defined as $G=(V,E,\phi,\varphi)$, in which V and E are the sets of nodes and edges, respectively. It is also associated with a node type mapping function $\phi: V \longrightarrow \phi_V$ and an edge type mapping function $\varphi: E \longrightarrow \varphi_E$, where ϕ_V and φ_E denote the sets of node and edge type, respectively. There exist $|\phi_V| >= 1$ and $|\varphi_E| >= 1$.

By coarsening the heterogeneous graph, the scale of the graph is reduced layer by layer until the top-level layer is reached. The all-levels graphs form a heterogeneous hierarchical structure.

Definition 2 Heterogeneous hierarchical structure. Given a heterogeneous graph $G=(V,E,\phi,\varphi)$. Utilize G to construct a series of different hierarchical levels of heterogeneous graphs: $G^0 \succ G^1 \succ \cdots \succ G^L$, where $G^0=G$. The resulting set $\mathrm{HG}=\left\{G^0,G^1,\ldots,G^L\right\}$ represents the L-layer heterogeneous hierarchical structure, where $\succ$ represents the structural transformation between graphs. In the heterogeneous hierarchical structure HG, the heterogeneous graph at the $(l+1)$-th layer, $G^{l+1}=(V^{l+1},E^{l+1},\phi,\varphi)$, is coarsened from the heterogeneous graph at the l-th layer, $G^l=(V^l,E^l,\phi,\varphi)$, denoted as $G^l \succ G^{l+1}$, where $l\in\{0,1,\ldots,L\}$. Specifically, the node set V^{l+1} and edge set E^{l+1} are constructed from V^l and E^l, satisfying $|V^l|>|V^{l+1}|$ and $|E^l|>|E^{l+1}|$.

By embedding the top-level graph and subsequently refining it back to the original graph, we obtain the node embeddings on the original graph.

Definition 3 Hierarchical heterogeneous graph embedding. Given a heterogeneous graph G, we design a mapping function $\rho: G^l \rightarrow G^{l+1}$ to achieve structural transformations between graphs: $G^l \succ G^{l+1}$, and thus construct an L-layer heterogeneous hierarchical structure HG. The goal of hierarchical heterogeneous graph representation is to utilize existing graph embedding methods to learn the node embeddings $Z^L \in \mathbb{R}^{|V^L|\times d}$ of the top-level heterogeneous graph $G^L \in mathrm HG$, and design a mapping function $\xi: Z^{l+1} \rightarrow Z^l$, so that the top-level layer node embeddings are refined back to the original graph $Z^L \succ Z^{L-1} \succ \cdots \succ Z^0$, obtaining the node embeddings $Z=Z^0$ for graph G. Here, $Z\in\mathbb{R}^{|V|\times d}$, $d \ll |V|$.

According to the above definition, we provide detailed introductions to the aspects of hierarchical heterogeneous graph embedding in the following sections.

4 Methodology

We implement hierarchical heterogeneous graph embedding based on relation structure and propose the Restage framework. This framework uses relation structure-aware graph coarsening to progressively reduce the size of the heterogeneous graph to a top-level graph with smaller scale while retaining the correspondence between nodes at different layers. Existing graph embedding methods for heterogeneous graphs can be applied to the top-level graph. Subsequently, the top-level embeddings are transferred to the original graph through the proposed relation structure-aware refinement method, and further reinforcing the transferred embeddings in each layer during this process. The proposed Restage framework shown in Fig. 2 that improves the efficiency of existing representation learning methods for heterogeneous graphs and enhances their scalability.

4.1 Relation structure-aware graph coarsening

In terms of form, graph coarsening methods belong to structural transformations between graphs, which reduce the size of large-scale graphs to smaller-scale ones while preserving the structure and features of the original graph as much as possible. Graph coarsening can be achieved through matrix operations on the adjacency matrix of a graph, which has been applied in many graph pooling works[33, 34]. Specifically, given the adjacency matrix A_l of the graph G^l at the l-th layer, the graph coarsening process is represented as follows:

$$A_{l+1} = M_{l,l+1}^{\mathrm{T}} A_l M_{l,l+1} \tag{1}$$

where $M_{l,l+1}$ is the cluster assignment matrix between layer l and layer $l+1$, which guides the graph coarsening process. The construction of $M_{l,l+1}$ is the key task in this section. The essence of the graph coarsening problem is the node merging problem. The heterogeneous hierarchical structure proposed in this paper merges nodes from the lower layer l to the $l+1$ layer through the cluster assignment matrix $M_{l,l+1}$. Specifically, the rows of $M_{l,l+1}$ represent the nodes in the l layer, and the columns represent the nodes in the $l+1$ layer. For instance, if two nodes v_i^l and v_j^l from layer l are merged into node v_k^{l+1} in level $l+1$, then there exists $m_{i,k} = 1$ and $m_{j,k} = 1$. From the perspective of layer $l+1$, nodes v_i^l and v_j^l should have some commonalities, as they are replaced by node v_k^{l+1} in

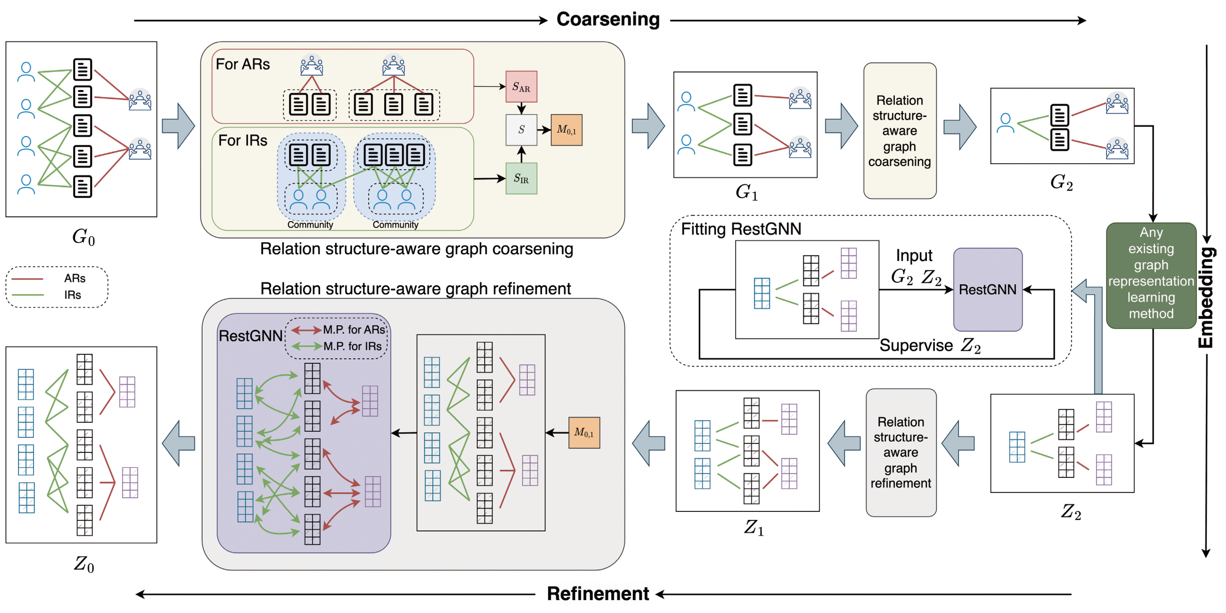

Fig. 2 Restage overall framework. Taking an academic graph as an example, which is coarsened twice, the original graph contains three types of nodes: authors, papers, and conferences, as well as two relation structures: IRs between authors and papers, and ARs between papers and conferences. In the coarsening part, we separately calculate the similarity of nodes in IRs and ARs, then merge the two similarity matrices to obtain S, find the most similar nodes in S to merge, construct the cluster assignment matrix M, and obtain the next-level graph G_1. Any graph representation learning (GRL) method can be used for embedding on the top-level graph G_2. During the refinement process, first, a RestGNN is fitted on the top-level layer, and the node embeddings in the top-level graph serve as both input and supervision. Then, according to the inter-layer cluster assignment matrix, the node embeddings from the upper layer are transferred to the current layer, and the parameters of RestGNN are fixed to reinforce the transferred embeddings. In this process, RestGNN models message passing both IRs and ARs.

layer $l+1$. Therefore, selecting similar nodes for merging in the graph is an essential step in the graph coarsening problem.

In heterogeneous graphs, there exist two relation structures: IRs and ARs. During the graph coarsening process, these two relations should be discussed separately. First, we introduce a method for calculating node similarity under IRs. In the l layer, the similarity matrix based on IRs is denoted as $S_{\mathrm{IR}} \in \mathbb{R}^{|V^l|\times|V^l|}$. As previously discussed, IRs resemble a peer-to-peer structure, and we use community detection methods to partition the connections between nodes under this structure. Nodes within the same community interact more closely and have some similarities, while nodes between communities interact sparsely and have weaker associations. Heterogeneous graphs contain multiple types of nodes, and using community detection for IRs preserves the semantic information of the heterogeneous graph to some extent. Specifically, by using the community detection algorithm **com** we divide the heterogeneous graph G^l at layer l into K communities, with each community serving as a new heterogeneous graph:

$$\left\{G_1^l, G_2^l, \ldots, G_K^l\right\} = \mathbf{com}\left(G^l\right) \tag{2}$$

After applying the community detection algorithm **com**, we obtain the heterogeneous graph $G^l = \left\{G_1^l, G_2^l, \ldots, G_K^l\right\}$. In this work, we adopt the Louvain algorithm[35] as the community detection method, which has better time efficiency and is more suitable for the application scenarios of our study. Notice that, Louvain algorithm is a community detection method based on modularity gain[36, 37] and the number of communities K cannot be predetermined.

In heterogeneous graphs, there are different types of nodes. Under the premise of merging the most similar nodes in the graph, we adopt the strategy of only merging nodes of the same type. Therefore, when calculating the similarity of nodes under the IRs, only the similarity between nodes of the same type is considered. Specifically, in the community G_k^l, nodes v_i and v_j of node type T_t, the element $s_{\mathrm{IR}}(i,j)$ in the similarity matrix $S_{\mathrm{IR}} \in \mathbb{R}^{|V^l|\times|V^l|}$, which is denoted as

$$s_{\mathrm{IR}}(i,j) = \frac{|\mathcal{N}_i \cap \mathcal{N}_j|}{\sum\limits_{k\in\left\{k|v_k\in G_k^l, \phi(v_k)=T_t\right\}} |\mathcal{N}_i \cap \mathcal{N}_k|} \tag{3}$$

where $\mathcal{N}_i$ means the neighbor nodes of v_i in the community G_k^l, $|\mathcal{N}_i \cap \mathcal{N}_j|$ represents the number of common neighbors between nodes v_i and v_j, $\phi(v_k)$ represents the node type of node v_k. The equation indicates that the similarity between two nodes of the same type is dependent on the quantity of their common neighbors within the community that share that same type.

The ARs are described as a one-centered-by-another structure. We suggest that the similarity based on the second-order structure can better reflect the relevance between peripheral nodes that belong to the same central node. For example, with the conference as the central node, papers published at the same conference have similarities. Specifically, given nodes v_i and v_j of node type T_t, the element $s_{\mathrm{AR}}(i,j)$ in the similarity matrix $S_{\mathrm{AR}} \in \mathbb{R}^{|V^l|\times|V^l|}$, which is denoted as

$$s_{\mathrm{AR}}(i,j) = \frac{1}{\left|\mathrm{sub}\left(\mathrm{sup}\left(v_i, v_j\right)\right)\right|} \tag{4}$$

where $\mathrm{sup}\left(v_i, v_j\right)$ represents the common center of nodes v_i and v_j, and $\mathrm{sub}(v)$ means the peripheral nodes of v. As seen from the equation, all peripheral nodes belonging to the same center have the same similarity. For ARs, the similarity between central nodes is not calculated, as we assume that there is no way to establish potential connections between central nodes under the ARs. For example, in the affiliation relation between papers (peripheral) and conferences (central), it is impossible to establish the similarity between conferences based solely on these two types of nodes because a paper is not published at two conferences simultaneously. Spatially speaking, the graph composed of paper and conference is discrete, and there is no reachable path between conferences. Although semantics such as "authors who have published papers at one conference have also published papers at another conference" (meta-path: CPAPC) can establish connections between conference nodes, we think that introducing IRs (relationships between authors and papers) in the ARs may increase model uncertainty. Therefore, our work does not consider this aspect for now.

After obtaining the similarity matrices S_{IR} and S_{AR} for nodes under the two types of relation structures, using a weighted summation method to fuse the two similarities. The calculation method of similarity matrix S is as follows:

$$S = \omega_{\mathrm{IR}} S_{\mathrm{IR}} + \omega_{\mathrm{AR}} S_{\mathrm{AR}} \tag{5}$$

where ω_{IR} and ω_{AR} are two hyper-parameters, representing the weights of IRs and ARs, respectively. Based on the similarity matrix S, traversing all nodes in the current layer l, selecting the node with the highest similarity to merge. If a node has already been selected as the most similar node by another node, then this node will be skipped in this iteration. Specifically, if a node's most similar node has already formed a super-node with other nodes, then this node will join them. Finally, based on the merged results, construct the cluster assignment matrix $M_{l,l+1}$.

4.2 Embedding on the top-level graph

According to the coarsen method introduced in the previous section, for a heterogeneous graph G, a hierarchical structure $\mathrm{HG}=\{G^0,G^1,...,G^L\}$ is constructed through layer-by-layer coarsening. Next, existing graph embedding methods to embed the top-level heterogeneous graph G^L. It can be formalized as follows:

$$Z_L=\mathbf{emb}\left(G^L\right) \tag{6}$$

where $\mathbf{Z}_L\in\mathbb{R}^{|V^L|\times d}$ means the node embedding of the L layer, **emb** means the embedding function. In the proposed hierarchical heterogeneous graph embedding framework, **emb** can be any unsupervised graph embedding method, which learns the node embeddings only utilizing the topology of the top-level heterogeneous graph. The graph coarsening method constructs a top-level graph, which to some extent preserves the structural information in the original heterogeneous graph.

4.3 Relation structure-aware graph refinement

By obtaining the node embedding through graph embedding method on the top-level graph, we acquire the embeddings for the top-level nodes. In this section, we introduce a relation structure-aware graph refinement method to refine these embeddings layer by layer to the original heterogeneous graph, obtaining the node representation of the original heterogeneous graph. An obvious way is to use the cluster assignment matrix $M_{l-1,l}$ between layers to transfer the embeddings Z_l from the l layer to the $l-1$ layer, specifically as follows:

$$Z_{l-1}=M_{l-1,l}Z_l \tag{7}$$

However, this way cannot preserve the interactions among intra-layer nodes. Moreover, such a transfer method causes nodes belonging to the same community within the $l-1$ layer to have the same representation, leading to the vector of nodes in the graph becoming more homogenized.

MILE proposes an intra-layer enhancement strategy, which allows the graph topology at $l-1$ layer to be integrated with the node embeddings Z_{l-1} during the refinement process. Theoretically, this enhancement strategy can be any model, even a simple multi-layer perceptron (MLP). To accommodate node embeddings and topology, while also considering the characteristics of heterogeneous graphs, we design a relation structure-aware graph refinement method. Specifically, this method first fits the node embeddings Z_L with a relation structure-aware graph neural network model at the top-level graph. Then, during the refinement process, the model parameters are fixed, and the inference step is performed at each layer, allowing the transformed embeddings of the nodes within the layer to be integrated with the topology structure. The loss function of fitting process of RestGNN at the top-level graph is represented as follows:

$$\mathcal{L}=\underset{\Theta}{\operatorname{argmin}}(Z_L-\mathrm{RestGNN}_\Theta\left(Z_L,G^L\right)) \tag{8}$$

where Θ represents the trainable parameters of the model. RestGNN is a relation structure-aware graph neural network, and the specific process for node v_i is as follows:

$$h_i^{(q+1)}=\sigma\left(\sum_{t\in\{\mathrm{IR},\mathrm{AR}\}}\sum_{j\in\mathcal{N}_i^t}W_t^{(q)}h_j^{(q)}+W_0^{(q)}h_i^{(q)}\right) \tag{9}$$

where $\{\mathrm{AR},\mathrm{IR}\}$ represents the combination of two types of relation structures, interaction relations and affiliation relations; $W_t^{(q)}$ is the weight matrix corresponding to relation structure t in the q-th layer of the RestGNN network, while $W_0^{(q)}$ represents the weight matrix for the node's own representation, both of which are learnable parameters. $\mathcal{N}_i^t$ denotes the set of neighbor nodes of node v_i under the relation structure t; σ represents the activation function.

As the equation shows, the aggregation process in RestGNN is determined by two relation structures. Different message passing ways are set for different relation structures. The fitting process initializes the node attributes by the embedding of the top-level graph using the existing graph representation method **emb**, i.e., for node v_i, there exists $h_i^{(0)}=z_i^L$, where $z_i^L\in Z_L$.

This fitting process also utilizes the embedding $\mathbf{Z}_L$ of the top-level graph G^L as supervision. Taking a two-layer RestGNN as an example, the loss from Eq. (8) is calculated as follows:

$$\mathcal{L} = \frac{1}{|V^L|} \sum_{i \in \{i | v_i \in V^L\}} \left\| z_i^L - h_i^{(2)} \right\|^2 \tag{10}$$

where $h_i^{(2)}$ means the embedding of v_i after aggregating through two layers of RestGNN.

After fitting RestGNN on the top-level graph, the model will fix the parameters Θ. In the subsequent refinement process, the model only performs inference and no longer trains. the relation structure-aware graph refinement method from the l layer to the $l-1$ layer can be described as

$$\begin{aligned} Z_{l-1} &= M_{l-1,l} Z_l, \\ Z'_{l-1} &= \text{RestGNN}_\Theta\left(Z_{l-1}, G^{l-1}\right), \\ Z_{l-1} &= Z'_{l-1} \end{aligned} \tag{11}$$

Repeat this process, and ultimately obtain the node embeddings Z_0, which serves as the final representation for the original heterogeneous graph. The overall steps of Restage are shown in Algorithm 1.

The Restage framework is primarily composed of two main components: coarsening and refinement. Given the number of nodes N within the current layer, the number of IRs E_{IR}, the number of ARs E_{AR}, and the number of communities K, the complexity analysis is as follows. In the coarsening phase, community detection is employed to calculate the similarity of nodes of the same type within a community for IRs. For ARs, nodes sharing the same central node possess identical similarity. Using the Louvain algorithm, which is applied in this work, as an example, the time and space complexity of a single coarsening operation is $O(E_{\text{IR}} \log E_{\text{IR}} + E_{\text{IR}} + E_{\text{AR}})$. This phase also necessitates consideration of the space occupied by the adjacency matrix and the cluster assignment matrix, leading to a corresponding space complexity of $O(N^2 + N * K)$. During the refine-ment phase, RestGNN is employed to adapt the node embeddings using any graph representation methods. Given the two types of relation structures, the time complexity of this phase is $O(E_{\text{IR}} + E_{\text{AR}} + N * 2)$. In terms of space occupancy, it necessitates the consideration of the transfer and reinforcement of embeddings, resulting in a corresponding complexity of $O(N^2 + K * N)$.

Algorithm 1 Restage framework

Input: A heterogeneous graph G and its corresponding adjacency matrix A, the number of coarsening layers L, and an arbitrary graph embedding algorithm **emb**.

Output: The representation Z of the graph G.

```
 1: $G^0 := G, A_0 := A$.
 2: for $l = 0$ to $L$ do
 3:     construct the similarity matrix under IRs, $S_{\text{IR}}$, based on Eq. (3).
 4:     construct the similarity matrix under ARs, $S_{\text{AR}}$, based on Eq. (4).
 5:     compute the similarity matrix $S$ for nodes in $G^l$ based on Eq. (5).
 6:     construct the cluster assignment matrix $M_{l,l+1}$
 7:     $A_{l+1} := M_{l,l+1}^{\text{T}} A_l M_{l,l+1}$
 8: end for
 9: $Z_L := \textbf{emb}\left(G^L\right)$
10: fitting the parameters $\Theta$ of RestGNN with Eq. (8)
11: for $l = L$ to $0$ do
12:     $Z_{l-1} := M_{l-1,l} Z_l$
13:     $Z'_{l-1} := \text{RestGNN}_\Theta\left(Z_{l-1}, G^{l-1}\right)$
14:     $Z_{l-1} := Z'_{l-1}$
15: end for
16: $Z := Z_0$
17: return $Z$
```

5 Experiment

In this section, we conduct experiments and analyses to validate the proposed method. The effectiveness of Restage is verified on four publicly available heterogeneous graph datasets.

5.1 Datasets

To demonstrate the effectiveness of our model, we conduct experiments on publicly available heterogeneous graph datasets that cover interactions in various complex real-world systems. For example, the ACM[25] and AMiner[38] datasets describe the relationships between papers and authors in academia, while the Freebase[39] dataset describes the relationships between movies, authors, and directors in the film industry. The overall information of the four datasets is shown in Table 1.

- ACM (Association for Computing Machinery) is an international association for computer science and information technology. The data is collected and extracted from the open-source heterogeneous graph toolkit, openHINE*, and has been used in many heterogeneous graph research works.

* https://github.com/BUPT-GAMMA/OpenHINE

Table 1 Statistics of the datasets.

Dataset	Node	Relation	Relation structure
ACM	#Paper(P): 4019	#P-S: 4019	AR
	#Author(A): 7167		IR
	#Subject(S): 60	#P-A: 13 407	
Freebase	#Movie(M): 3492	#M-A: 65 341	AR
	#Actor(A): 33 401	#M-D: 3762	IR
	#Director(D): 2502		IR
	#Writer(W): 4459	#M-W: 6414	
AMiner	#Paper(P): 6564	#P-A: 18 007	IR
	#Author(A): 13 329		IR
	#Reference(R): 35 890	#P-R: 58 831	
AMiner_L	#Paper(P): 127 623	#P-A: 355 072	IR
	#Author(A): 164 473		IR
	#Reference(R): 147 251	#P-R: 392 519	

• Freebase is an open-knowledge graph-based database[40] that aims to structure all the world's knowledge. The graph composed of four types of entities from Freebase as the dataset, which includes movies, directors, actors, and editors.

• AMiner is an academic search and academic data analysis platform[41]. The dataset used is a subgraph extracted from a citation network, containing three types of nodes: articles, authors, and references.

• AMiner_L and AMiner come from the same citation graph, with AMiner_L being larger in scale. The use of this dataset allows for further validation of the scalability of the proposed framework for existing heterogeneous graph representation learning methods.

5.2 Baselines

To validate the scalability of Restage, we utilize two classic representation learning methods, Node2vec[42] (N2V) and Metapath2vec[13] (MP2V), as foundation methods. These two methods are applied to the embeddings of the top-level heterogeneous graphs, corresponding to the **emb** function in Eq. (6). Simultaneously, we introduce several baselines for evaluation. For heterogeneous graph embedding, in addition to the classic MP2V, we consider methods based on meta-paths, such as HERec[18] and HAN[16], as well as methods based on network schemes like NSHE[25] and HeCo[43]. For homogeneous graph embedding, we adopt for MILE[6], which also employs a hierarchical framework. Notice that N2V, MP2V, HERec, and the proposed Restage adopt unsupervised training without features. NSHE and HeCo are also unsupervised training but utilizes node attributes. HAN is a semi-supervised training method using features and partly node labels. Implementation details of baselines are described as follows:

• Node2vec is an unsupervised graph representation method by leveraging random walks and optimizing these representations using the skip-gram from word2vec[44].

• Metapath2vec is the classical heterogeneous representation method. It is based on meta-path sampling and the heterogeneous skip-gram model.

• HERec employs multiple meta-paths for sampling and integrates node representations under different meta-paths, making it suitable for recommendation tasks on heterogeneous graphs.

• HAN is a deep model based on graph neural networks that learn embeddings of a node based on node-level and semantic-level attentions.

• NSHE uses network scheme sampling and designs a multi-task learning task to maintain the heterogeneous structure of each schema instance.

• HeCo employs a contrastive learning setup that compares network schemes and meta-paths, thereby facilitating a self-supervised representation learning approach.

• MILE is a notable work in the homogeneous graph representation domain that investigates hierarchical structures. Its proposed coarsening and refinement framework effectively improves the efficiency of existing representation learning algorithms.

5.3 Implementation details

Running environment. The experiments in this paper are conducted on Intel(R) Core(TM) i5-10210U CPU@1.60 GHz, 16 GB of RAM, and 512 GB HDD. In order to ensure a fair comparison with all baselines, the proposed Restage framework, including both the coarsening, embedding and refinement processes, operates entirely on the CPU, including the fitting of RestGNN on the top-level graph. Although we acknowledge that using a GPU for RestGNN fitting could further reduce the time consumption, we avoid doing so in the interest of ensuring the framework's scalability.

Parameter configuration. The community detection algorithm **com** used in this work is the Louvain algorithm, which includes a modularity gain threshold. In the coarsening process, the threshold value is kept constant at 10^{-5}. The results presented in the experiments are the average values obtained from ten runs. The embedding dimension d for all baseline methods is set to 128 to ensure a fair comparison.

5.4 Scalability of Restage

In the proposed Restage framework, the original heterogeneous graph is compressed into a top-level graph with a smaller scale through graph coarsening. The node embeddings are obtained on the top-level graph using existing graph embedding methods, and then the node embeddings on the top-level graph are transferred back to the original graph using the graph refinement method. In this section, we use two classical graph embedding methods, N2V and MP2V, to embed the top-level graph. We adopt node classification as the downstream task to evaluate the performance of the model, in order to better understand the effectiveness and performance of the proposed method in practical applications. We input the final node embeddings obtained on the original graph into a logistic regression classifier and set the training ratios to 20%, 40%, 60%, and 80%. The experimental results are shown in Table 2, where l represents the number of times the graph is coarsened.

The results on the three datasets further confirm the effectiveness of the Restage framework. Compared to directly using N2V or MP2V on the original graph, better results are obtained using these two methods under the Restage with the time consumption is significantly reduced. This demonstrates that the Restage framework has enhanced the scalability of existing graph embedding methods on heterogeneous graphs to some extent. On the AMiner and ACM datasets, the MP2V method in the Restage framework

Table 2 Macro-F1 and Micro-F1 scores (%) and time consumption (seconds) for the node classification task on the three datasets.

Dataset	Metric	Train	N2V	Restage (N2V) (L=1)	Restage (N2V) (L=2)	Restage (N2V) (L=3)	MP2V	Restage (MP2V) (L=1)	Restage (MP2V) (L=2)	Restage (MP2V) (L=3)
ACM	Mi-F1	20%	81.27	**84.79**	82.96	81.19	73.91	82.65	**83.40**	81.59
		40%	82.66	**84.99**	83.79	81.22	75.66	83.42	**84.00**	81.76
		60%	82.75	**85.82**	83.40	80.10	74.32	**84.39**	84.14	80.53
		80%	83.51	**86.69**	84.08	81.34	77.61	**84.58**	83.71	79.10
	Ma-F1	20%	81.02	**84.24**	82.78	79.91	71.55	82.15	**83.02**	80.36
		40%	82.23	**84.86**	83.42	80.44	73.67	83.04	**83.76**	80.87
		60%	82.41	**85.54**	82.76	79.93	72.19	**83.99**	83.64	79.39
		80%	83.28	**86.48**	83.38	80.80	75.94	**84.29**	83.24	78.26
	Time (SpeedUp)		257 (6.9×)	74 (2.0×)	46 (1.2×)	**37 (1.0×)**	1251 (6.7×)	565 (3.0×)	257 (1.3×)	**185 (1.0×)**
AMiner	Mi-F1	20%	83.98	**85.61**	84.84	82.96	75.77	85.05	**85.93**	85.57
		40%	85.22	**86.29**	85.25	83.93	77.71	**86.47**	86.19	84.16
		60%	85.26	**87.09**	86.63	84.84	78.32	**86.94**	86.84	84.39
		80%	81.45	**87.66**	87.51	84.46	77.75	86.98	**87.36**	84.23
	Ma-F1	20%	73.29	**75.63**	73.90	69.07	55.42	**73.34**	72.89	64.41
		40%	74.49	**75.79**	73.45	69.13	58.27	**74.86**	72.83	65.25
		60%	74.62	**77.29**	75.53	71.05	58.65	**75.58**	74.19	65.34
		80%	73.52	**77.81**	77.49	70.53	58.17	**75.73**	75.38	66.36
	Time (SpeedUp)		1634 (22.0×)	315 (4.2×)	120 (1.6×)	**74 (1.0×)**	3250 (9.3×)	2069 (5.9×)	834 (2.3×)	**348 (1.0×)**
Freebase	Mi-F1	20%	66.03	67.22	65.86	**68.43**	64.76	65.21	**67.97**	63.14
		40%	68.10	**69.13**	69.08	68.32	65.62	**68.23**	67.94	63.93
		60%	68.36	**71.01**	70.15	69.08	64.49	**69.22**	67.34	63.35
		80%	66.70	**71.24**	70.53	70.24	63.23	**70.96**	67.10	63.23
	Ma-F1	20%	61.42	**62.30**	58.50	60.38	54.05	**59.57**	54.14	46.28
		40%	63.03	**63.97**	60.77	60.69	55.19	**62.60**	54.07	46.99
		60%	64.04	**66.21**	62.34	61.49	54.52	**63.49**	53.53	46.48
		80%	62.83	**66.62**	63.05	63.63	54.14	**65.86**	53.72	46.49
	Time (SpeedUp)		1358 (16.3×)	255 (3.0×)	123 (1.4×)	**83 (1.0×)**	7876 (77.2×)	1021 (10.0×)	249 (2.4×)	**102 (1.0×)**

when coarsen one times (L=1) has an 8% to 19% improvement over directly using MP2V on the original heterogeneous graph, which is likely due to the positive effect of the refinement method proposed in this paper. MP2V can only retain the semantics represented by a given meta-path, while the refinement method proposed in this paper enhances the representation within each layer when refining the top-level node representation back to the original heterogeneous graph, making up for the semantic deficiencies of MP2V.

In the experiments, the number of coarsening steps L has a positive effect on reducing the time consumption, and the more coarsening steps, the less time consumed. However, this is not linearly related. For example, the time consumption of coarsening three times on the Freebase and AMiner datasets is not much different from coarsening two times, which is likely because the community structure in the graph is relatively stable, and further coarsening does not significantly reduce the graph's scale. In Fig. 3, we further illustrate the positive impact of the number of coarsening steps on the compression of the graph. Our coarsening strategy aims to merge the most similar nodes in the graph, usually when coarsening once, the scale of the original graph is successfully reduced to less than 50%. However, as the number of coarsening steps increases (L=2, 3), the changes in the scale of the graph are relatively small. This may be because the community structure in the graph becomes stable, and nodes with topological differences will not converge due to the increase in the number of coarsening steps. Due to the relative stability of the graph scale, the time consumption of coarsening three times is not much different from that of coarsening twice on the three datasets. Excessive times of coarsening may lead to the loss of information in the graph, which is also confirmed in the experiments. The best results on the three datasets are mainly concentrated when coarsening one or two times. Notably, when coarsening once, although the scale of the network is significantly reduced, the results of the classification even slightly improved under the Restage framework, and the time cost is effectively reduced. This further proves that Restage is beneficial to enhance the scalability of existing representation learning methods on heterogeneous graph.

5.5 Quality of Restage

Restage not only enhances the scalability of existing graph representation learning methods but also improves their performance. In this section, we further demonstrate the excellent quality of Restage through comparisons with graph representation learning baseline methods on two downstream tasks, node classification and node clustering. For the node classification task, we adopt the same settings as in the previous section. For node clustering, we use the obtained node representations to train KMeans, using normalized mutual information (NMI) as the evaluation metric. Restage uses N2V as the embedding method for the top-level graph. The number of coarsening steps L is set to 1. As observed from the analysis in the previous section, significant graph scale reduction and relatively better results are achieved when the graph is coarsened once.

Node classification. Table 3 shows the results of node classification tasks on three datasets, Overall,

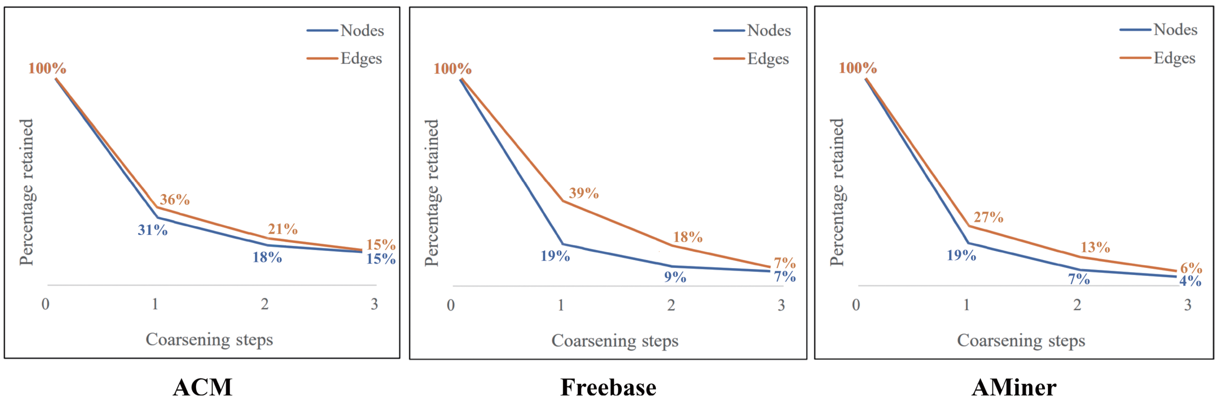

Fig. 3 Percentage of node and edge counts compared to the original graph as coarsening steps increase. The x-axis represents the number of coarsening steps, and the y-axis represents the percentage of the number of nodes (edges) at the current level compared to the number of nodes (edges) in the original graph.

Restage achieves the best or second-best results in most cases and significantly outperforms other baselines in terms of speed. The classical methods N2V and MP2V perform stably on all three datasets, but their sampling and shallow model learning strategies lead to higher time overheads. HAN uses node attributes and labels for semi-supervised learning, and its results are close to the best results on all three datasets. Baselines based on network schemes, namely NSHE and HeCo, have shown substantial competence across all three datasets, particularly on Freebase and AMiner. This indicates to some extent the advantage of network schemes in preserving local information for handling more complex interactions within the graph. However, this comes with a considerable time expense, as clearly exhibited by NSHE, whose training time even exceeds six days. In contrast, our Restage not only achieves similar, but in some cases even superior results, while utilizing the least amount of time. Compared with MILE, a hierarchical graph representation learning method in homogeneous networks, Restage achieves further speed improvements, which may be due to the higher edge folding efficiency brought by the relation structure-aware coarsening method adopted in this study. Moreover, Restage performs slightly better than MILE on Freebase and AMiner datasets, which can be attributed to the relation structure-aware refinement method.

Node clustering. We perform node clustering tasks on three datasets, with results displayed in Table 4. The running times of each algorithm remain consistent with

Table 3 Macro-F1 and Micro-F1 scores (%) and time consumption (seconds) for all baselines on node classification task.

Dataset	Metric	Train	N2V	MILE	MP2V	HERec	HAN	NSHE	HeCo	Restage
ACM	Mi-F1	20%	81.27	83.37	73.91	81.30	81.56	81.62	64.30	**84.79**
		40%	82.66	84.41	75.66	82.28	81.76	81.75	79.27	**84.99**
		60%	82.75	84.71	74.32	82.40	82.89	82.67	83.33	**85.82**
		80%	83.51	85.32	77.61	82.74	83.77	84.45	85.07	**86.69**
	Ma-F1	20%	81.02	83.94	71.55	81.07	80.81	80.85	52.32	**84.24**
		40%	82.23	84.75	73.67	82.16	81.43	81.18	75.49	**84.86**
		60%	82.41	85.12	72.19	82.31	82.18	82.48	81.47	**85.54**
		80%	83.28	85.62	75.94	82.76	83.30	84.36	83.77	**86.48**
	Time (SpeedUp)		257 (3.4×)	191 (2.5×)	1251 (16.9×)	342 (4.6×)	212 (2.8×)	1556 (21.0×)	481 (6.5×)	**74 (1.0×)**
AMiner	Mi-F1	20%	83.98	83.99	75.57	80.52	82.23	84.77	**85.95**	85.61
		40%	85.22	84.89	77.71	82.06	82.42	84.85	86.49	**86.69**
		60%	85.26	85.65	78.32	82.63	82.57	84.72	86.75	**87.09**
		80%	81.45	86.15	77.75	81.71	81.82	83.81	84.92	**87.66**
	Ma-F1	20%	73.29	68.74	55.42	67.69	66.09	71.61	74.28	**75.63**
		40%	74.49	70.57	58.27	69.75	66.84	71.85	75.09	**75.79**
		60%	74.62	72.12	58.65	69.72	66.16	71.95	75.14	**77.29**
		80%	73.52	73.64	58.17	68.29	65.40	71.11	71.98	**77.81**
	Time) (SpeedUp)		1634 (5.1×)	684 (2.1×)	3250 (10.3×)	412 (1.3×)	693 (2.2×)	> 6 days (/)	2736 (8.6×)	**315 (1.0×)**
Freebase	Mi-F1	20%	66.03	68.46	64.76	62.34	68.29	**70.01**	69.36	67.22
		40%	68.10	69.06	65.62	64.04	69.08	**70.04**	69.51	69.13
		60%	68.36	68.66	64.49	64.96	68.46	69.85	68.58	**71.01**
		80%	66.70	68.99	63.23	65.17	66.92	68.55	66.81	**71.24**
	Ma-F1	20%	61.42	59.73	54.05	57.69	60.79	**64.53**	61.58	62.30
		40%	63.33	61.14	55.19	59.83	61.77	**64.49**	62.05	63.97
		60%	64.04	61.05	54.52	61.17	62.11	65.41	62.23	**66.21**
		80%	62.83	62.20	54.14	61.78	61.07	64.92	60.93	**66.62**
	Time (SpeedUp)		1358 (9.6×)	773 (5.4×)	7876 (55.8×)	275 (1.9×)	263 (1.6×)	> 6 days (/)	9542 (67.6×)	**255 (1.0×)**

those in Table 3, as we simply apply the node representations learned by these baselines to different downstream tasks. Restage demonstrates superior performance on both the AMiner and Freebase datasets. On the ACM dataset, HeCo yields significantly higher clustering results than other baselines, likely due to its utilization of node attributes which are particularly apt for clustering tasks. However, as seen in Table 3, this method does not stand out in the classification experiment. Evaluating MILE's results in both classification and clustering, it consistently exhibits a robust performance, not far from the optimal results. This demonstrates the effectiveness of the hierarchical framework for graph representation learning methods in some extend. Restage, specifically designed for hierarchical heterogeneous graph representation, achieves better results with less time overhead.

5.6 Performance of Restage on large-scale graph

AMiner_L is a subgraph extracted from the AMiner citation network, which includes about 400 000 nodes, 740 000 edges, and three types of nodes: papers, authors, and references. In this work, we use this dataset to analyze the application effect of the proposed Restage on the large-scale graph. We also adopt the node classification task and train a classifier with the obtained node representations at different ratios. We employ N2V as the representation learning method on the top-level graph and set L=1, 2, 3 as the three numbers of times of coarsening for comparison.

Table 4 NMI values (%) for all baselines on node clustering task.

Method	ACM	AMiner	Freebase
N2V	44.60	37.83	16.58
MILE	45.00	40.86	16.49
MP2V	27.24	34.16	18.88
HERec	15.76	15.84	4.04
HAN	40.62	26.73	18.13
NSHE	37.71	36.12	16.36
HeCo	**62.87**	33.36	20.14
Restage	51.81	**47.94**	**21.09**

Table 5 shows the results of the node classification task on the AMiner_L dataset. In this case, MILE's coarsest graph chooses N2V as the node embedding method, with two layers of coarsening. HAN is not included in the comparison because their required runtime environment exceeds memory limitations. NSHE is not included in the comparison due to training times exceeding six days. HeCo is not considered because the dataset does not include node attribute information. The proposed Restage achieves the best results when coarsened once, although its performance decreases relatively after three times of coarsening, its speed is improved. Restage, like MILE, applies N2V for embedding in the top-level graph. Compared to utilizing N2V directly on the original graph, its speed is increased by about eighteen times, and the classification performance remains basically the same.

5.7 Ablation analysis

The proposed Restage framework consists of two main components: relation structure-aware graph coarsening and relation structure-aware graph refinement. To demonstrate the contribution of each part to the framework, we design two variants, Restage$_{\text{com}}$ and Restage$_{\text{GCN}}$. Restage$_{\text{com}}$ does not account for the relation structures of heterogeneous graphs during the coarsening process. Instead, it employs community detection methods to partition the original graph into multiple blocks. It then computes the similarity of nodes of the same type within each block and merges the most similar nodes. Restage$_{\text{com}}$ implements the coarsening layer-by-layer in this manner. During the refining phase, Restage$_{\text{GCN}}$ does not take into account

Table 5 Node classification task on AMiner_L dataset.

Metric	Micro-F1 (%)				Macro-F1 (%)				Time (min)
Train	20%	40%	60%	80%	20%	40%	60%	80%	/
N2V	59.52	59.72	59.99	60.03	54.37	54.19	54.41	54.53	415 (18.86×)
MP2V	56.17	56.52	56.73	56.38	50.54	51.13	51.46	51.39	131 (5.94×)
HERec	52.55	52.91	53.07	53.52	45.52	46.03	46.52	47.13	289 (13.13×)
MILE	61.67	61.95	62.11	61.91	56.53	56.98	**57.38**	**57.43**	123 (5.59×)
Restage (L=1)	**61.84**	**62.24**	**62.33**	**62.05**	**56.55**	**57.01**	57.30	57.30	72 (3.27×)
Restage (L=2)	60.68	61.12	61.15	61.24	55.01	55.38	55.61	55.98	32 (1.45×)
Restage (L=3)	57.40	57.65	57.83	57.54	50.03	50.24	50.47	50.56	**22** (1.00×)

the relation structures. It utilizes GCN instead of RestGNN to fit the node embeddings at the top layer. Subsequently, it refines back to the original graph.

Table 6 shows the results of two ablation experiments on ACM dataset, where all three frameworks use N2V as the embedding method at the top-level graph. Restage, which considers the relation structure in both coarsening and refinement, performs better than the two variants, demonstrating the necessity of perceiving relation structures in hierarchical heterogeneous graph representation learning. This point is also corroborated by the comparison with MILE in Sec. 5.5. Moreover, $\text{Restage}_{\text{com}}$ incurs lower computational overhead compared to the other two frameworks. This is attributed to the utilization of community detection during the coarsening process, which significantly reduces the graph size. However, this also results in a notable performance decline, particularly at L=3.

5.8 Parameter analysis

In the process of relation structure-aware graph coarsening, it is necessary to merge the similarity of nodes under two kinds of relation structures as shown in Eq. (5). For this purpose, parameters ω_{IR} and ω_{AR} are set to represent the weights of IRs and ARs respectively. In the previous experiments, we set both parameters to 0.5, assuming they are equally important. In this section, we discuss the sensitivity of these two parameters. Figure 4 illustrates the results of node classification on the ACM dataset with different coarsening steps. We attempt to enumerate potential values for these two parameters, having combinations of $\omega_{\text{IR}}, \omega_{\text{AR}} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

As observed from Fig. 4, the F1 score under different weight combinations decreases as the number of coarsening iterations (L) increases. The decline is most significant when coarsening occurs three times (L=3). This is consistent with previous experimental results,

Table 6 Ablation analysis on ACM dataset.

Metric	Micro-F1 (%)				Macro-F1 (%)				Time (s)
Train	20	40	60	80	20	40	60	80	/
$\text{Restage}_{\text{com}}$ (L=1)	84.32	83.99	85.75	85.44	84.17	83.68	85.24	85.12	**67**
$\text{Restage}_{\text{GCN}}$ (L=1)	83.64	83.58	83.27	83.70	83.24	83.32	83.00	83.44	75
Restage (L=1)	**84.79**	**84.99**	**85.82**	**86.69**	**84.24**	**84.86**	**85.54**	**86.48**	74
$\text{Restage}_{\text{com}}$ (L=2)	81.25	82.91	81.71	82.33	80.42	82.14	80.71	81.74	**28**
$\text{Restage}_{\text{GCN}}$ (L=2)	82.24	82.62	81.84	82.33	81.45	82.12	81.22	81.93	52
Restage (L=2)	**82.96**	**83.79**	**83.40**	**84.08**	**82.78**	**83.42**	**82.76**	**83.38**	46
$\text{Restage}_{\text{com}}$ (L=3)	75.68	77.28	77.05	76.86	72.76	75.67	75.18	75.13	**10**
$\text{Restage}_{\text{GCN}}$ (L=3)	80.56	80.88	**80.97**	81.09	79.09	79.52	79.68	79.97	61
Restage (L=3)	**81.19**	**81.22**	80.10	**81.34**	**79.91**	**80.44**	**79.93**	**80.80**	37

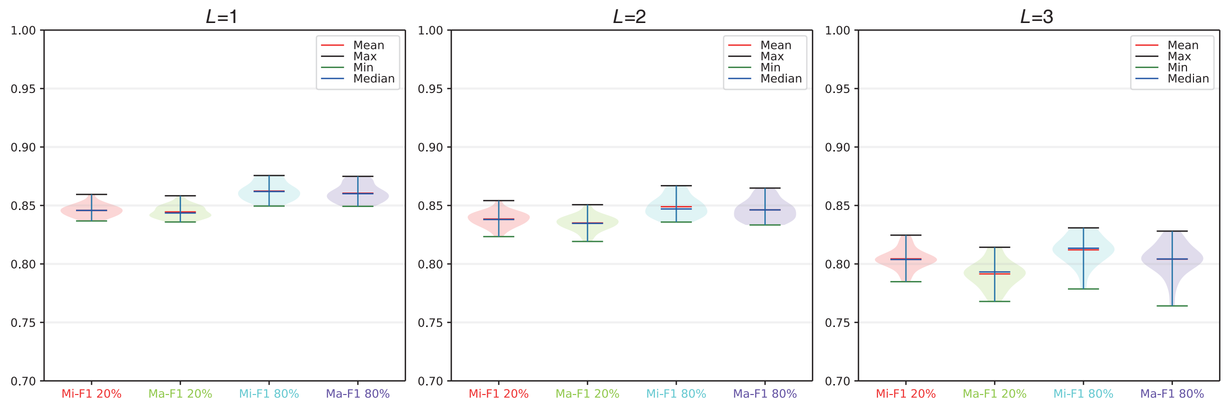

Fig. 4 Node classification results corresponding to different weight parameters ω_{IR}, ω_{AR} under different coarsening steps. The evaluation metrics are as follows, in order: Micro-F1 and Macro-F1 values using 20% of samples as training, and Micro-F1 and Macro-F1 values using 80% of samples as training. The combinations of the values of parameters ω_{IR} and ω_{AR} range between {0.1, 0.3, 0.5, 0.7, 0.9}, and the corresponding Micro-F1 and Macro-F1 values are listed in the violin plot.

where an increase in layers leads to a reduction in the graph's scale, which accelerates the model speed but compromising accuracy due to information loss during the coarsening process. Simultaneously, as L increases, the length (upper and lower bounds) of the violin plot gradually extends, indicating an increased model susceptibility to these two parameters ω_{IR} and ω_{AR}. This might be attributable to the gradual stabilization of the graph's community structure, causing the similarity of nodes under IRs to converge towards a certain value, thereby amplifying the impact of the weight ω_{IR}. Upon holistic analysis, the fluctuation range of the F1 score with the changes in these two parameters remains within approximately 0.06, which we deem acceptable. This variation range suggests a certain degree of robustness in the model.

6 Conclusion

In this paper, we propose a relation structure-aware hierarchical heterogeneous graph embedding framework, Restage, which allows existing representation learning methods to obtain the node embeddings of the original graph by a smaller-scale graph. Restage first designs a relation structure-aware coarsening method, calculating the similarity of nodes under different relation structures, and merging similar nodes. Then, in the top-level graph, any unsupervised graph representation method is allowed to use for embedding. Finally, a relation structure-aware refinement method is designed to transfer the node embeddings from the top-level graph to the original graph. During the refinement process, within each layer, the transferred node embeddings are reinforced utilizing the topology of the current layer graph. Experiments on three public datasets demonstrate that Restage enhance scalability for existing graph representation learning methods on heterogeneous graphs.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (Nos. 1876001, 61602003, and 61673020), the National High Technology Research and Development Program (No. 2017YFB1401903), the Provincial Natural Science Foundation of Anhui Province (No. 1708085QF156). We also acknowledge the High-Performance Computing Platform of Anhui University for providing computing resources.

References

[1] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han, Heterogeneous network representation learning: A unified framework with survey and benchmark, *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4854–4873, 2022.

[2] X. Wang, D. Bo, C. Shi, S. Fan, Y. Ye, and P. S. Yu, A survey on heterogeneous graph embedding: Methods, techniques, applications and sources, *IEEE Trans. Big Data*, vol. 9, no. 2, pp. 415–436, 2023.

[3] B. Perozzi, R. Al-Rfou, and S. Skiena, DeepWalk: Online learning of social representations, in *Proc. 20th ACM SIGKDD Int. Conf. Knowledge discovery and data mining*, New York, NY, USA, 2014, pp. 701–710.

[4] X. Wang, X. He, Y. Cao, M. Liu, and T. S. Chua, KGAT: Knowledge graph attention network for recommendation, in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, Anchorage, AK, USA, 2019, pp. 950–958.

[5] A. Xu, P. Zhong, Y. Kang, J. Duan, A. Wang, M. Lu, and C. Shi, THAN: Multimodal transportation recommendation with heterogeneous graph attention networks, *IEEE Trans. Intell. Transport. Syst.*, pp. 1–11, 2022.

[6] J. Liang, S. Gurukar, and S. Parthasarathy, MILE: A multi-level framework for scalable graph embedding, *Proc. Int. AAAI Conf. Web Soc. Medium.*, vol. 15, pp. 361–372, 2021.

[7] C. Deng, Z. Zhao, Y. Wang, Z. Zhang, and Z. Feng, GraphZoom: A multi-level spectral approach for accurate and scalable graph embedding, in *Proc. 8th Int. Conf. on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[8] S. Zhao, Z. Du, J. Chen, Y. Zhang, J. Tang, and P. S. Yu, Hierarchical representation learning for attributed networks, in *Proc. IEEE 38th Int. Conf. Data Engineering* (*ICDE*), Kuala Lumpur, Malaysia, 2022, pp. 2641–2656.

[9] Y. Lu, C. Shi, L. Hu, and Z. Liu, Relation structure-aware heterogeneous information network embedding, *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 1, pp. 4456–4463, 2019.

[10] C. Shi, Y. Lu, L. Hu, Z. Liu, and H. Ma, RHINE: Relation structure-aware heterogeneous information network embedding, *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 433–447, 2022.

[11] S. Zhu, C. Zhou, S. Pan, X. Zhu, and B. Wang, Relation structure-aware heterogeneous graph neural network, in *Proc. IEEE Int. Conf. Data Mining* (*ICDM*), Beijing, China, 2019, pp. 1534–1539.

[12] J. Tang, M. Qu, and Q. Mei, PTE: Predictive text embedding through large-scale heterogeneous text networks, in *Proc. 21th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Sydney, Australia, 2015, pp. 1165–1174.

[13] Y. Dong, N. V. Chawla, and A. Swami, metapath2vec: Scalable representation learning for heterogeneous networks, in *Proc. 23rd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Halifax, Canada, 2017, pp. 135–144.

[14] T.-Y. Fu, W.-C. Lee, and Z. Lei, HIN2Vec: Explore meta-paths in heterogeneous information networks for representation learning, in *Proc. 2017 ACM on Conf. Information and Knowledge Management*, Singapore, Singapore, 2017, pp. 1797–1806.

[15] M. Defferrard, X. Bresson, and P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in *Proc. 30th Int. Conf. Neural Information Processing Systems*, Barcelona, Spain, 2016, pp. 3844–3852.

[16] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, Heterogeneous graph attention network, in *Proc. World Wide Web Conference*, San Francisco, CA, USA, 2019, pp. 2022–2032.

[17] X. Wang, Y. Zhang, and C. Shi, Hyperbolic heterogeneous information network embedding, *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 1, pp. 5337–5344, 2019.

[18] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, Heterogeneous information network embedding for recommendation, *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 357–370, 2019.

[19] T. N. Kipf and M.Welling, Semi-supervised classification with graph convolutional networks, in *Proc. 6th Int. Conf. on Learning Representations*, Toulon, France, 2017.

[20] P. Veličkovit'c, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, Graph attention networks, in *Proc. 6th Int. Conf. on Learning Representations*, Vancouver, Canada, 2017.

[21] W. L. Hamilton, R. Ying, and J. Leskovec, Inductive representation learning on large graphs, in *Proc. 31st Int. Conf. Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 1025–1035.

[22] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, Modeling relational data with graph convolutional networks, in *Lecture Notes in Computer Science*, R. D. Deshpande and R. D. Deshpande, eds. Cham, Switzerland: Springer, 2018, pp. 593–607.

[23] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, Representation learning for attributed multiplex heterogeneous network, in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, Anchorage, AK, USA, 2019, pp. 1358–1368.

[24] X. Fu, J. Zhang, Z. Meng, and I. King, MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding, in *Proc. The Web Conf. 2020*, Taipei, China, 2020, pp. 2331–2341.

[25] J. Zhao, X. Wang, C. Shi, Z. Liu, and Y. Ye, Network schema preserved heterogeneous information network embedding, in *Proc. 29th Int. Joint Conf. on Artificial Intelligence*, Yokohama, Japan, 2020, pp. 1366–1372.

[26] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, Simplifying graph convolutional networks, in *Proc. 36th Int. Conf. on Machine Learning*, Long Beach, CA, USA, 2019, pp. 6861–6871.

[27] E. Rossi, F. Frasca, B. Chamberlain, D. Eynard, M. Bronstein, and F. Monti, SIGN: Scalable inception graph neural networks, arXiv preprint arXiv: 2004.11198, 2020.

[28] C. Sun and G. Wu, Scalable and Adaptive graph neural networks with self-label-enhanced training, arXiv preprint arXiv: 2104.09376, 2021.

[29] W. Zhang, Z. Yin, Z. Sheng, Y. Li, W. Ouyang, X. Li, Y. Tao, Z. Yang, and B. Cui, Graph attention multi-layer perceptron, in *Proc. 28th ACM SIGKDD Conf. Knowledge Discovery and Data Mining*, Washington, DC, USA, 2022, pp. 4560–4570.

[30] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, HARP: Hierarchical representation learning for networks, *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, pp. 2127–2134, 2018.

[31] G. Fu, C. Hou, and X. Yao, Learning topological representation for networks via hierarchical sampling, in *Proc. Int. Joint Conf. Neural Networks* (*IJCNN*), Budapest, Hungary, 2019, pp. 1–8.

[32] A. K. Bhowmick, K. Meneni, M. Danisch, J. L. Guillaume, and B. Mitra, LouvainNE: Hierarchical Louvain method for high quality and scalable network embedding, in *Proc. 13th Int. Conf. Web Search and Data Mining*, Houston, TX, USA, 2020, pp. 43–51.

[33] E. Ranjan, S. Sanyal, and P. Talukdar, ASAP: Adaptive structure aware pooling for learning hierarchical graph representations, *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, pp. 5470–5477, 2020.

[34] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec, Hierarchical graph representation learning with differentiable pooling, in *Proc. 32nd Int. Conf. Neural Information Processing Systems*, Montréal, Canada, 2018, pp. 4805–4815.

[35] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, Fast unfolding of communities in large networks, *J. Stat. Mech.*, vol. 2008, no. 10, p. P10008, 2008.

[36] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, *Phys. Rev. E*, vol. 69, no. 2, p. 026113, 2004.

[37] M. E. J. Newman, Modularity and community structure in networks, *Proc. Natl. Acad. Sci. U. S. A.*, vol. 103, no. 23, pp. 8577–8582, 2006.

[38] B. Hu, Y. Fang, and C. Shi, Adversarial learning on heterogeneous information networks, in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, Anchorage, AK, USA, 2019, pp. 120–129.

[39] X. Li, D. Ding, B. Kao, Y. Sun, and N. Mamoulis, Leveraging meta-path contexts for classification in heterogeneous information networks, in *Proc. IEEE 37th Int. Conf. Data Engineering* (*ICDE*), Chania, Greece, 2021, pp. 912–923.

[40] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, Freebase: A collaboratively created graph database for structuring human knowledge, in *Proc. 2008 ACM SIGMOD Int. Conf. Management of data*, Vancouver, Canada, 2008, pp. 1247–1250.

[41] H. Wan, Y. Zhang, J. Zhang, and J. Tang, AMiner: Search and mining of academic social networks, *Data Intell.*, vol. 1, no. 1, pp. 58–76, 2019.

[42] A. Grover and J. Leskovec, node2vec: Scalable feature learning for networks, in *Proc. 22nd ACM SIGKDD Int.*

Conf. Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 855–856.

[43] X. Wang, N. Liu, H. Han, and C. Shi, Self-supervised heterogeneous graph neural network with co-contrastive learning, in *Proc. 27th ACM SIGKDD Conf. Knowledge Discovery & Data Mining*, Virtual Event, Singapore, Singapore, 2021, pp. 1726–1736.

[44] T. Mikolov, K. Chen, G. Corrado, and J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv: 1301.3781, 2013.

Huanjing Zhao received the MEng degree in computer science from Anhui University, Hefei, China, in 2019. He is currently a PhD candidate at Anhui University. His research interests include graph representation learning and social network analysis.

Pinde Rui received the MEng degree in computer science from Anhui University, Hefei, China, in 2022. His research interests include heterogeneous network and network representations learning.

Jie Chen received the PhD degree in computer science from Anhui University in 2014. She is now an associate professor in School of Computer Science and Technology, Anhui University. Her current research interests include quotient space theory and granular computing.

Shu Zhao received the PhD degree in computer science from Anhui University, Hefei, China, in 2007. She is currently a professor with School of Computer Science and Technology, Anhui University. Her current research interests include quotient space theory, granular computing, social networks, and machine learning.

Yanping Zhang received the PhD degree from Anhui University, Hefei, China, in 2005. She is currently a professor with Anhui University. She is a principal investigator of 973 projects and the leader of the National Natural Science Foundations of China and Anhui Province. Her main research interests include computational intelligence, quotient space theory, artificial neural networks, intelligent information processing, and machine learning.