


ITC 1/52 Information Technology and Control Vol. 52 / No. 1 / 2023 pp. 199-214 DOI 10.5755/j01.itc.52.1.32390	Relation-Aware Weighted Embedding for Heterogeneous Graphs	
	Received 2022/09/27	Accepted after revision 2023/01/23
	 http://dx.doi.org/10.5755/j01.itc.52.1.32390	

HOW TO CITE: Hu, G., Pang, J. (2023). Relation-Aware Weighted Embedding for Heterogeneous Graphs *Information Technology and Control*, 52(1), 199-214. <http://dx.doi.org/10.5755/j01.itc.52.1.32390>

Relation-Aware Weighted Embedding for Heterogeneous Graphs

Ganglin Hu

College of Computer and Information Science, Southwest University, 400715 Chongqing, China;
e-mail: huganglin88@outlook.com

Jun Pang

Faculty of Science, Technology and Medicine & Interdisciplinary Centre for Security, Reliability and Trust,
University of Luxembourg, L-4364 Esch-sur-Alzette, Luxembourg

Corresponding author: huganglin88@outlook.com

Heterogeneous graph embedding, aiming to learn the low-dimensional representations of nodes, is effective in many tasks, such as link prediction, node classification, and community detection. Most existing graph embedding methods conducted on heterogeneous graphs treat the heterogeneous neighbours equally. Although it is possible to get node weights through attention mechanisms mainly developed using expensive recursive message-passing, they are difficult to deal with large-scale networks. In this paper, we propose R-WHGE, a relation-aware weighted embedding model for heterogeneous graphs, to resolve this issue. R-WHGE comprehensively considers structural information, semantic information, meta-paths of nodes and meta-path-based node weights to learn effective node embeddings. More specifically, we first extract the feature importance of each node and then take the nodes' importance as node weights. A weighted random walks-based embedding learning model is proposed to generate the initial weighted node embeddings according to each meta-path. Finally, we feed these embeddings to a relation-aware heterogeneous graph neural network to generate compact embeddings of nodes, which captures relation-aware characteristics. Extensive experiments on real-world datasets demonstrate that our model is competitive against various state-of-the-art methods.

KEYWORDS: Heterogeneous graphs, weighted random walks, graph neural networks, graph embedding.

1. Introduction

Heterogeneous graphs are widely used to construct and organize various complex systems in which the objects are interrelated or interacted, such as academic networks, e-commerce and social networks [20, 24]. A heterogeneous graph consists of many types of entities with different relationships among them.

As shown in Figure 1, a heterogeneous academic graph instance consists of four types of entities: authors (A), papers (P), venues (V) and organizations (O). There are five relationships among different types of entities: affiliating/affiliated-to ($O \rightarrow A/A \rightarrow O$), coauthor ($A \rightarrow A$), writing/written-by ($A \rightarrow P/P \rightarrow A$), citing/cited ($P \rightarrow P$) and publishing/published-in ($V \rightarrow P/P \rightarrow V$). Recently, one popular tool to analyze heterogeneous graphs is the meta-paths, which could better preserve relationship information between different types of entities. As shown in

Figure 1

A heterogeneous academic graph instance (a) and its graph schema (b). The red dotted lines represent coauthor relationships, and the blue dotted lines denote citation relationships

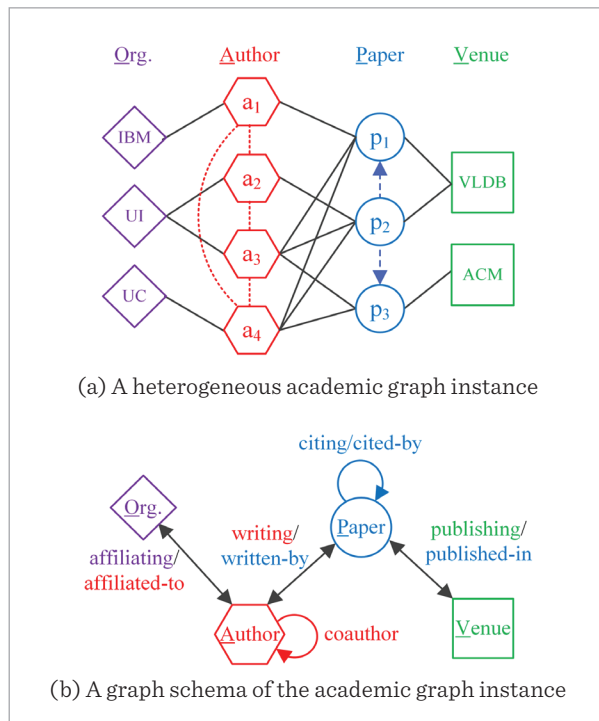


Figure 2

Examples of meta-paths for the heterogeneous academic graph instance. **(a) APA:** it means that authors collaborate on the same paper, and the examples of the meta-path instances are $a1-p1-a3$, $a3-p2-a4$, and $a3-p3-a4$, etc. **(b) APV:** it means that authors publish papers at a venue, and the examples of the meta-path instances are $a1-p1-VLDB$, $a3-p2-VLDB$, and $a4-p3-ACM$, etc. **(c) APVPA:** it means that authors publish papers on the same venue, and the examples of the meta-path instances are $a1-p1-VLDB-p2-a3$, $a3-p1-VLDB-p2-a4$, and $a4-p3-ACM-p3-a3$, etc. **(d) OAPVPAO:** it means that authors affiliated to the same or different organizations publish papers on the same venue, and the examples of the meta-path instances are $IBM-a1-p1-VLDB-p2-a3-UI$, $UI-a3-p1-VLDB-p2-a4-UC$, and $UC-a4-p3-ACM-p3-a3-UI$, etc.

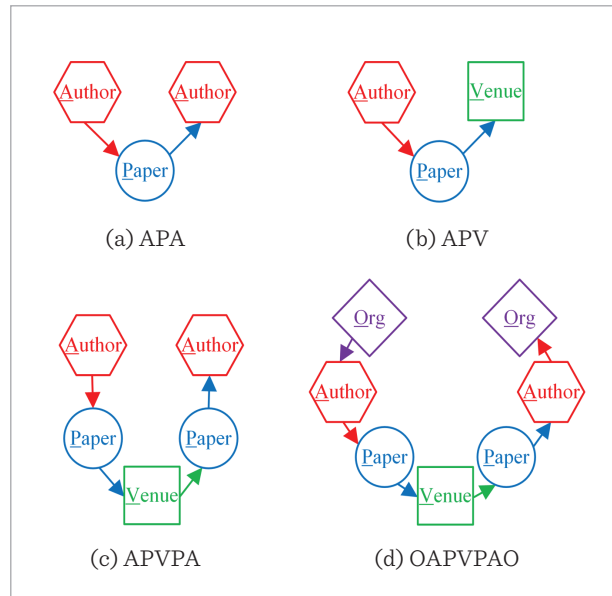


Figure 2, the meta-path author-paper-author (abbreviated as APA) indicates that two authors coauthor the same paper, while the meta-path APVPA conveys the information that two papers written by some authors are published in the same journal.

Heterogeneous graph embedding aims to map nodes of graphs into a low-dimensional vector space. The main goal of embedding learning is to improve the effectiveness of feature learning from a heterogeneous graph. The learned features can, in turn, improve tasks such as node classification [4, 30] and link prediction [1, 16, 29]. However, it is a challenge to achieve

this goal in heterogeneous graphs. Based on the number of neural network layers, the existing methods can be divided into two categories.

The first category consists of shallow models. For homogeneous networks, there exist a few classical models, such as DeepWalk [19], Line [25], and Node2Vec [8], which learn node embeddings using the Skip-gram model [17]. Metapath2vec [4], HERec [23] and HIN2Vec [5] are the three early works on heterogeneous graph embedding learning. However, these shallow models only consider the graph topology structure and cannot incorporate (heterogeneous) features of nodes [15] (e.g. text or image), resulting in inferior performance.

The second category consists of deep graph models basing on graph neural networks (GNNs) [15, 26], which extract the graph's structure and the heterogeneous nodes' features for graph embedding learning. Some of the GNNs were designed for node embedding learning in homogeneous graphs, including GCN [15], GraphSAGE [9] and GAT [26]. When dealing with heterogeneous graphs, the different types of nodes are divided into multiple-homogeneous subgraphs, losing some of the relational semantic information of heterogeneous graphs. Besides, several heterogeneous graph embedding methods have been designed based on GNNs. HetGNN [33] jointly considers node heterogeneous features encoding, type-based neighbours aggregation, and heterogeneous types combination. HetSANN [11] learns node embeddings in a heterogeneous graph by considering the heterogeneous structural information and each node's heterogeneous feature information. Specifically, the heterogeneous information is projected in low-dimensional entity spaces by means of an attention mechanism. HGT [13] is used to model web-scale heterogeneous graphs. To model heterogeneity, node- and edge-type dependent parameters are designed to characterize the heterogeneous attention over each edge, empowering HGT to maintain dedicated representations for different types of nodes and edges. Recently, R-HGNN [31] exploits the role of relations and collaboratively learns relation-aware node embeddings as well as semantic embeddings of relations. Different from these existing methods, in this paper, we propose model **R-WHGE**, a novel relation-aware weighted embedding model for heterogeneous graphs, which combines weighted random walks and weighted-based

GNN to learn node embeddings effectively.

In reality, each node has different importance [10, 18] in graphs. The importance of a node is an inherently objective matter, which depends on the node contents and its position in the graph. Specifically, the more important a node is, the more nodes will establish direct connections with it. The networks generated by this law are scale-free networks whose degree distribution of nodes follows a power law. For example, the degree distributions of e-commerce and academic networks are power-law. It is important to characterize the power-law feature of node degree distribution in scale-free networks.

Taking the importance aggregation of a heterogeneous academic network instance, shown in Figure 3, each author¹ equally gives his/her importance to papers according to the meta-path APA. Then the papers aggregate these importances in the step of AP. Papers written by more authors or more important authors have more importance. Similarly, each paper shares its importance equally with the authors, and then the authors aggregate the importances in the step of PA. As a result, authors who write more papers or whose papers have more importances [18] possess more importances. In the academic network example, Han and Sun have more importances since they write more papers. Similarly, GenClus has more importance than NetClus since GenClus is written by more authors or more important authors than NetClus. It can be seen that these nodes' importances are an essential feature of embedding learning, and the representation vectors of nodes without such importances are incomplete. However, these existing embedding methods are not able to cope with the importances of nodes.

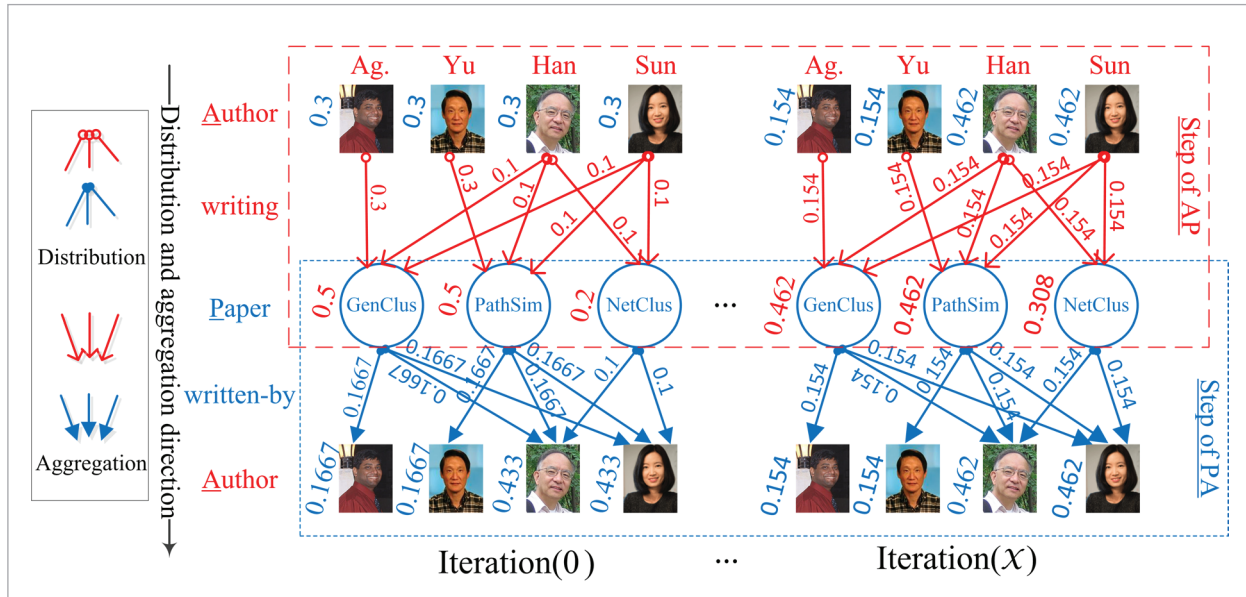
In order to integrate heterogeneous information considering the weight, some models [2, 11, 14, 16, 28] aggregate information by means of an attention mechanism in heterogeneous networks. But the attention mechanism is implemented through an expensive recursive message-passing mechanism and is hard to cope with large-scale networks.

To tackle the above-identified problems, in this paper, we propose R-WHGE, a novel relation-aware weighted heterogeneous graph embedding. Specifically, we first explore het-PageRank to generate node

1 <https://www.aminer.cn/ranks/home>

Figure 3

The weight aggregation for an academic network example according to the meta-path APA. Step of AP: each author gives its importance to nodes of papers equally, and then the papers aggregate their importances; Step of PA: each paper shares its importance equally with nodes of authors, and then authors aggregate their importances. Iteration(0) and Iteration(X) are the first and last convergent iterations in which the node's importances distribute and aggregate in the academic network according to the meta-path APA



importances, then combine it with meta-paths to generate weighted random walks sequences, which are fed to MetaPath2Vec to get the initial weighted node embeddings. Finally, the initial weighted node embeddings will be given to GNNs, a relation-aware heterogeneous graph neural network, to generate the final node embeddings. Extensive experiments are conducted on node classification tasks, and the results show that our approach outperforms existing methods consistently among four real-world heterogeneous graph datasets. The contributions of this paper are summarized as follows:

- We propose the algorithm het-PageRank to capture the importance of each node in a heterogeneous graph.
- Taking the importance of each node in a heterogeneous graph as node weights, we propose the algorithm of a weighted random walks-based embedding learning (WRWE) to better encode the relationship semantics among nodes into initial weighted node embeddings.
- Finally, R-HGNN, which received the initial weighted node embeddings, generates the multiple

relations-aware node embeddings for each node, which are fused into a compact node embedding.

The rest of the paper is organized as follows. Section 2 discusses related work. We provide some definitions and problem formulation in Section 3.

Section 4 presents in detail our proposed **R-WHGE** framework. We then show experimental results in Section 5 before concluding the paper in Section 6.

2. Related Work

In the last few years, a large number of graph embedding models have been proposed to learn node embedding in graphs efficiently. The methods of embedding learning are mainly shallow graph models and deep graph models. In this section, we briefly review related work in these two categories.

Shallow graph models. The shallow graph models can better extract multiple relational semantic features of heterogeneous graphs. Metapath2vec [4] uses a heterogeneous Skip-Gram to learn the node

embeddings in heterogeneous networks. HERec [23] thoroughly mines latent structure features of users and items using a random walk strategy based on meta-paths to derive more meaningful node sequences for network embedding. HIN2Vec [5] exploits rich semantics of relationships and the network structure details to learn nodes' embeddings using a random walk strategy based on meta-paths in heterogeneous networks. However, the above methods only learn graph structure features and do not consider heterogeneous features of nodes. Many graph datasets have rich, heterogeneous feature information, which can potentially be informative in the embedding encoding process.

Deep graph models. The deep graph model based on GNNs can fetch the graph structure and heterogeneous features of nodes. GCN [15] uses an efficient layer-wise propagation mechanism and learns local graph topology structure and node features based on the first-order approximation of spectral convolutions on graphs. GraphSAGE [9] generates embeddings leveraging topology structure and node feature information (e.g., text attributes). The topology structure features are fetched via sampling and aggregating features from node neighbourhoods. GAT [26] enables implicitly specifying different weights to different nodes in neighbourhoods of a node, leveraging masked self-attentional layers. RGCN [22] models relational data in knowledge graphs by employing specialized transformation matrices for each relation type, which is effective at encoding features of local and structured neighbourhoods. RSHN [34] builds an edge-centric coarsened line graph to describe relation semantic information and then integrates the graph and its coarsened line graph to embed both nodes and edges without requiring of meta-path. HGT [13] models Web-scale heterogeneous graphs by learning the features of different nodes and relations with type-specific parameters based on the architecture of the Transformer [32]. HetGNN [33] incorporates heterogeneous graph topology structure information and heterogeneous contents information of each node. MAGNN [6] aggregates features including node-attributes, intermediate semantic nodes along meta-paths and messages from multiple meta-paths. R-HGNN [31] exploits the relation-aware characteristics and learns node representations on heterogeneous graphs. Besides, quite a few other literature

reviews [3, 7, 27] provide a comprehensive, up-to-date review of the state-of-the-art of various graph embedding models and explain their differences. They cover early work on preserving graph structure and a new surge of incorporating heterogeneous features of nodes. The node-importance feature reflects the status and importance of nodes in the network. Nevertheless, none of them comprehensively consider the graph structure, heterogeneous features of nodes and node's importance when learning node embeddings.

3. Problem Definition

In this section, we present the necessary formal definitions and formulate our research problem.

Definition 1. (Heterogeneous Information Graph).

A heterogeneous graph (HG) is defined as $G = (V, E, T, R)$. Each node $v \in V$ represents an entity. $|V|$ is the total number of nodes in G . Each edge $e \in E$ ($e = (v, v')$) denotes a relation. $T = \{T_1, T_2, \dots, T_{|T|}\}$ denotes a set of entity type, where $|T|$ is the total number of entity type. $R = \{r_1, r_2, \dots, r_{|R|}\}$ denotes a set of relation types, where $|R|$ is the total number of relation types. Especially, $R(v)$ denotes a set of relation types that v is associated with. In a HG , every node and every edge are associated with their mapping functions $\phi: V \mapsto T$, $\psi: E \mapsto R$ and the inverse functions $\phi^{-1}: T \mapsto V$, $\psi^{-1}: R \mapsto E$, where $\ln |T| + |R| > 2$.

For example, as shown in Figure 1, we represent a heterogeneous academic graph instance and its graph schema, with the entity type $T = \{A, P, V, O\}$ wherein A represents authors, P represents papers, V represents venues, and O represents organizations, respectively. The nodes $V = \{IBM, UI, UC, a_1, a_2, a_3, a_4, p_1, p_2, p_3, VLDB, ACM\}$ are entities, and the set of entity relation types R is indicated by the relationship types of all edges of affiliating/affiliated-to ($O \rightarrow A/A \rightarrow O$), coauthor ($A \rightarrow A$), writing/ written-by ($A \rightarrow P/P \rightarrow A$), citing/cited ($P \rightarrow P$) and publishing/published-in ($V \rightarrow P/P \rightarrow V$), that is $R = \{O \rightarrow A, A \rightarrow O, A \rightarrow A, A \rightarrow P, P \rightarrow A, P \rightarrow P, V \rightarrow P, P \rightarrow V\}$. For instance, the node types of a_1 and p_1 can be calculated by $\phi(a_1) = A$ and $\phi(p_1) = P$. The node type of other nodes can be obtained similarly. The node-sets of type A and P can be obtained by the functions $\phi^{-1}(A) = \{a_1, a_2, a_3, a_4\}$ and $\phi^{-1}(P) = \{p_1, p_2, p_3\}$. Because the relation type of $p_1 \rightarrow a_1$, $p_1 \rightarrow a_3$ and $p_1 \rightarrow a_4$ belongs to $P \rightarrow A$ and the relation type of

$p_1 \rightarrow VLDB$ belongs to $P \rightarrow V$, so $R(p_1) = \{P \rightarrow A, P \rightarrow V\}$.

Definition 2. (Graph Schema). A graph schema is defined as $S_G = (T, R)$, which is a meta template for $G = (V, E, T, R)$ with the object type mappings $\phi: V \mapsto T$, and $\psi: E \mapsto R$. The graph schema specifies type constraints on both entities and relationships, guiding the semantics explorations of heterogeneous graphs. An example is shown in Figure 1.

Definition 3. (Meta-path). A meta-path \mathcal{P}_m is a sequence of object types that characterize heterogeneous information graphs. Given a graph schema $S_G = (T, R)$, a meta-path \mathcal{P}_m is the form of $T_1 \xrightarrow{r_1} T_2 \xrightarrow{r_2} \dots \xrightarrow{r_{L-1}} T_L$, abbreviated as $\mathcal{P}_m = T_1 T_2 \dots T_L$, wherein L denotes the length of a meta-path \mathcal{P}_m , $T_l \in \mathcal{P}_m$ and $T_l \in T$ ($1 \leq l \leq L$). $R = r_1 \circ r_2 \circ \dots \circ r_{L-1}$ describes a composite relation between node types T_1 and T_L , “ \circ ” denotes the composition operator on relations. All meta-paths are represented as $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{|\mathcal{P}|}\}$, $\mathcal{P}_m \in \mathcal{P}$, where $|\mathcal{P}|$ is the total number of meta-paths.

If there is a meta-path \mathcal{P}_m where $T_i = T_{L-i+1}$ and $R_i = R_{L-i}$ ($1 \leq i \leq \lfloor L/2 \rfloor$), we call the meta-path satisfies symmetry. We assume that the meta-paths in this paper are symmetric.

For example, as shown in Figure 2, we explore some meta-paths and their meta-path instances based on a given graph schema, wherein $\mathcal{P}_1 = APA$, $\mathcal{P}_2 = APV$, $\mathcal{P}_3 = APVPA$, $\mathcal{P}_4 = OAPVPAO$ and $\mathcal{P} = \{APA, APV, APVPA, OAPVPAO\}$.

Definition 4. (Meta-path Instance). A meta-path instance ρ is defined as a node sequence, following the meta-path \mathcal{P}_m , in the HG , as shown in Figure 2.

Definition 5. (Meta-path-based Neighbours). The meta-path-based neighbours N_v^{l+1} of a node v^l is the set of nodes that directly connect with node v^l via meta-path instances, wherein “ l ” indicates the entity-type of a node, and “ $l + 1$ ” indicates the next entity-type according to a meta-path \mathcal{P}_m . Especially, the neighbour nodes of a node v include itself when adjacent entities on a meta-path \mathcal{P}_m are of the same entity type (for example, when $\mathcal{P}_m = AA$, neighbour nodes $N_{a_1}^A = \{a_1, a_2, a_4\}$ of the node a_1 include a_1 itself).

Taking for example the meta-path $\mathcal{P}_m = APA$, the current node p_1 and the “ l ” of current node type, the mapping function ϕ computing the current node type is “ $l = \phi(p_1) = P$ ” and the neighbour nodes type is “ $l+1 = \phi(N_{p_1}^{l+1}) = A$ ”, wherein $N_{p_1}^A = \{a_1, a_2, a_3\}$ according to meta-path APA , as shown in Figure 1(a).

Definition 6. (Heterogeneous Graph Embedding).

Given a heterogeneous graph $G = (V, E, T, R)$, we aim to learn a mapping function $f: v \mapsto \vec{z}_v \in \mathbb{R}^d$ ($d \ll |V|$) for $\forall v \in V$. The purpose of the function f is to save the node similarity into a low dimensional feature vector \vec{z}_v . The log-likelihood objective is shown as follows:

$$\mathcal{L} = - \arg \min_f \sum_{v \in V} \sum_{u \in N_{\mathcal{R}}(v)} \log P(u|\vec{z}_v),$$

where $u \in N_{\mathcal{R}}(v)$ means u is the neighbour of node v and is visited by the strategy \mathcal{R} of a combination of weighted random walks and graph neural network in the heterogeneous graph G . The similarity between nodes u and v reflects the node’s similarity with a given graph’s structural properties and attribute features.

4. Our Method

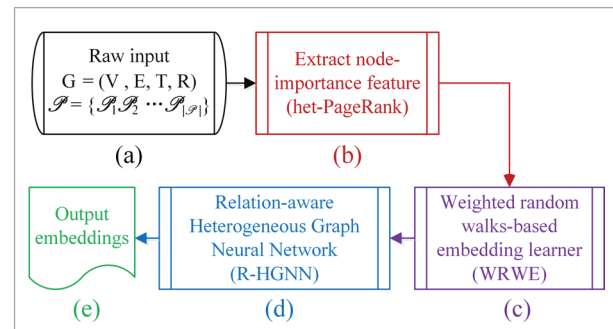
In this section, we first introduce the framework of the proposed model and then introduces each component step by step.

4.1. Framework of the Proposed Model

The framework of the proposed model **R-WHGE** is shown in Figure 4. **R-WHGE** consists of two com-

Figure 4

Overview of the framework R-WHGE. **(a) Raw input:** a heterogeneous information graph G and all meta-paths $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{|\mathcal{P}|}\}$, wherein $\mathcal{P}_m = T_1 T_2 \dots T_L$. **(b)** The algorithm of het-PageRank generates the importance of each node in G . **(c) WRWE:** Taking the importance of each node as node’s weights, node sequences are sampled by weighted random walks according to a meta-path \mathcal{P}_m and input these node sequences into MetaPath2Vec [4] which generates the weighted random walks-based embedding learning (WRWE). **(d)** A relation-aware heterogeneous graph neural network that catches relation-aware characteristics of neighbours to get compact embeddings. **(e)** The model output



ponents: weighted random walks-based embedding learning (**WRWE**) and relation-aware heterogeneous graph neural network (**R-HGNN**). Originally, an algorithm of het-PageRank is proposed to capture the importance of each node in a heterogeneous graph which is taken as each node's weight. The first component **WRWE** extracts spatial-topology features for each node (Section 4.2). The second component **R-HGNN** catches relational semantic features of heterogeneous neighbours and finally fuses the relation-aware node embeddings (Section 4.3).

4.2. WRWE

We propose the WRWE algorithm (as shown in Algorithm 1) to sample node sequences and generate the weighted random walks-based embeddings for each node v in heterogeneous graphs.

Node Weight Coefficient. First, we define a node's weighted vector $\vec{W}^{\mathcal{P}_m}$ consisting of the weighted coefficients of all nodes in a heterogeneous graph G .

In order to obtain the vector $\vec{W}^{\mathcal{P}_m}$, we update the PageRank [10, 18] algorithm according to a \mathcal{P}_m named *het-PageRank*. The algorithm of *het-PageRank* generates the importance [10, 18] of each node according to the \mathcal{P}_m in G . We take the importance of each node as node weights, and all the node's importance of the G compose the weighted vector $\vec{W}^{\mathcal{P}_m} = (w_1^{\mathcal{P}_m}, w_2^{\mathcal{P}_m}, \dots, w_{|V|}^{\mathcal{P}_m})$, where $|V|$ is the number of nodes in G , \mathcal{P}_m represents a meta-path, and $w_i^{\mathcal{P}_m}$ ($0 \leq i < |V|$) represents a node weight.

The first step of *het-PageRank* is to obtain a relational matrix set $\Delta^{\mathcal{P}} = (\delta^{\mathcal{P}_1}, \delta^{\mathcal{P}_2}, \dots, \delta^{\mathcal{P}_L})$ of G according to meta-paths \mathcal{P} . Each element $\delta(v_j, v_i, \mathcal{P}_m)$ of the $\delta^{\mathcal{P}_m}$ ($m \in [1, \mathcal{P}_m]$) is generated under the guidance of a meta-path \mathcal{P}_m and defined as follows:

$$\delta(v_j, v_i^l, \mathcal{P}_m) = \begin{cases} 1, & (v_j, v_i^l) \in E, \phi(v_j) = l + 1 \\ 0, & (v_j, v_i^l) \in E, \phi(v_j) \neq l + 1 \\ 0, & (v_j, v_i^l) \notin E \end{cases} \quad (1)$$

where l and $l+1$ indicate the current node (entity) type and next node (entity) type of the \mathcal{P}_m .

From Equation (1), $\delta(v_j, v_i^l, \mathcal{P}_m) = 0$ denotes that nodes v_i^l and v_j do not directly connected under a metapath \mathcal{P}_m and $\delta(v_j, v_i^l, \mathcal{P}_m) = 1$ means that nodes

v_i^l and v_j are neighbours under a \mathcal{P}_m . So the relational matrix $\delta^{\mathcal{P}_m}$ reflects the neighbour's relationship among nodes according to a \mathcal{P}_m . The matrix $\delta^{\mathcal{P}_m}$ is a symmetric matrix because the \mathcal{P}_m is symmetric (refer to Definition 3) in our algorithm *het-PageRank*. For example, Table 1 is the relational matrix δ^{APA} according to the heterogeneous academic graph instance (as shown in Figure 1) and the $\mathcal{P}_m = APA$ (as shown in Figure 2).

Table 1

A relational matrix δ^{APA} of the heterogeneous academic graph instance (as shown in Figure 1(a)) based on the meta-path APA (as shown in Figure 2). We can see that the meta-path-based neighbours (Definition 5) of a_3 are p_1, p_2, p_3 , the meta-path-based neighbours of p_1 are $\{a_1, a_3, a_4\}$, the meta-path-based neighbours of p_2 are $\{a_2, a_3, a_4\}$ and so on

entity	a_1	a_2	a_3	a_4	p_1	p_2	p_3
a_1	0	0	0	0	1	0	0
a_2	0	0	0	0	0	1	0
a_3	0	0	0	0	1	1	1
a_4	0	0	0	0	1	1	1
p_1	1	0	1	1	0	0	0
p_2	0	1	1	1	0	0	0
p_3	0	0	1	1	0	0	0

The second step of *het-PageRank* is to obtain the weighted vector $\vec{W}^{\mathcal{P}_m} = (w_1^{\mathcal{P}_m}, w_2^{\mathcal{P}_m}, \dots, w_{|V|}^{\mathcal{P}_m})$ of G according to a \mathcal{P}_m , wherein the $w_i^{\mathcal{P}_m}$ is the weight of a node v_i ($1 \leq i \leq |V|$). The vector $\vec{W}^{\mathcal{P}_m}$ is iteratively computed using the following Equation (2):

$$\vec{W}_{n+1}^{\mathcal{P}_m} = \|\delta^{\mathcal{P}_m} \vec{W}_n^{\mathcal{P}_m}\|_p, \text{ until } |\vec{W}_{n+1}^{\mathcal{P}_m} - \vec{W}_n^{\mathcal{P}_m}| \leq \epsilon, \quad (2)$$

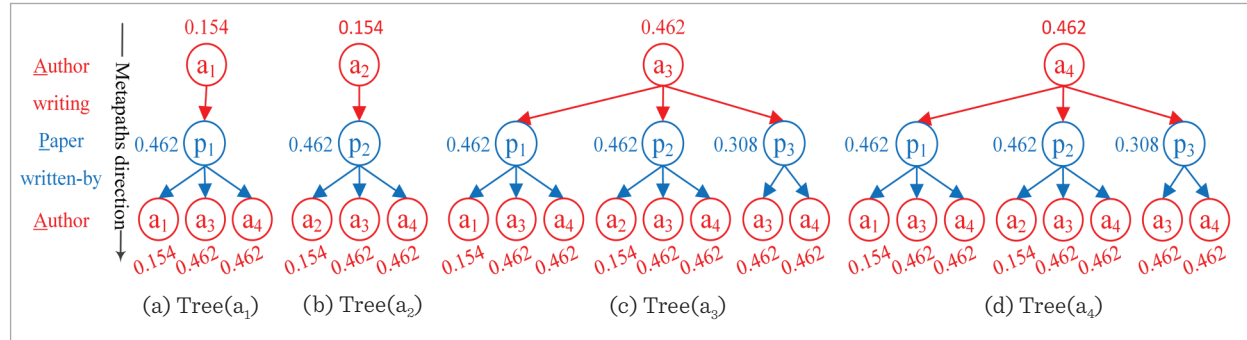
where $\epsilon = \frac{1}{10^{20}}$ denotes a hyperparametric constant and $p = 2$ is a parameter of the L2-normalization. We initialize $\vec{W}_1^{\mathcal{P}_m} = (0.3, 0.3, \dots, 0.3)$ for the first iteration and the $n, n+1$ denote two adjacent iterations.

Sampling Quantity. Before discussing the sampling quantity of meta-path instances, we first define the concepts of sampling trees and sampling forests. A sampling tree can be constructed as follows.

- 1 Given a meta-path $\mathcal{P}_m = T_1 T_2 \dots T_L$, the first node $v^1 \in \phi^{-1}(T_1)$ is chosen as the root node of the sampling tree.

Figure 5

A meta-path-based (APA) sampling forest



- 2 If the type index l of the node v^l in the \mathcal{P}_m is less than L , then we take the meta-path-based neighbours N_v^{l+1} (see Definition 5, and Equation (1)) as child nodes of the node v^l , otherwise the iteration terminates.
- 3 The child nodes N_v^{l+1} of the v^l are taken as the new root nodes. Go back to step 2 and process for another iteration guided by the \mathcal{P}_m .

For example, in the heterogeneous academic graph instance (see Figure 1) according to a meta-path $\mathcal{P}_m = T_1 T_2 T_3 = \text{APA}$, the root nodes can be obtained by $\phi^{-1}(T_1) = \phi^{-1}(A) = \{a_1, a_2, a_3, a_4\}$. By querying the relational matrix δ^{APA} shown in Table 1, the sampling trees of $Tree(a_1)$, $Tree(a_2)$, $Tree(a_3)$ and $Tree(a_4)$ can be constructed (refer to Figure 5). Taking $Tree(a_3)$ (refer to Figure 5c) as an example, $\{p_1, p_2, p_3\}$ are the child node-set of a_3 by querying Table 1. In a same manner, we can find the child node-set $\{a_1, a_3, a_4\}$ of p_1 , the child node-set $\{a_2, a_3, a_4\}$ of p_2 and the child node-set $\{a_3, a_4\}$ of p_3 . Finally, all the sampling trees of $Tree(a_1)$, $Tree(a_2)$, $Tree(a_3)$ and $Tree(a_4)$ form a meta-path-based (APA) sampling forest.

The sampling quantity of meta-path instances, briefly named sampling quantity, refers to the number of paths from the root node v^1 to the leaf node in a sampling tree $Tree(v^1)$. The sampling quantity $\varpi^{\mathcal{P}_m}(v^1)$ of meta-path instances can be expressed as follows.

$$\varpi^{\mathcal{P}_m}(v^1) = \lceil \kappa \cdot w_{v^1}^{\mathcal{P}_m} \rceil, \quad (3)$$

$$v^1 \in \phi^{-1}(T_1) \text{ and } \mathcal{P}_m = T_1 T_2 \dots T_L,$$

where \mathcal{P}_m denotes a meta-path, v^1 represents the starting node of a meta-path or the root-node of a

$Tree(v^1)$, $w_{v^1}^{\mathcal{P}_m}$ indicates the root-node weights of a $Tree(v^1)$, κ is an amplification coefficient, and $\lceil \cdot \rceil$ means upwards to the nearest integer.

As the weight of a meta-path starting node v^1 is obtained by aggregating neighbour-node weights iteratively in a sampling tree $Tree(v^1)$ according to Equation (2), and the weight of v^1 can reflect the importance of $Tree(v^1)$, we acquire the sampling quantity of meta-path instances by the weight multiple of v^1 . All sampling quantities of a sampling forest are defined as the following set:

$$\Omega^{\mathcal{P}_m} = \{\varpi^{\mathcal{P}_m}(v_1), \varpi^{\mathcal{P}_m}(v_2), \dots, \varpi^{\mathcal{P}_m}(v_{|\phi^{-1}(T_1)|})\}, \quad (4)$$

$$v_i \in \phi^{-1}(T_1).$$

Taking Figure 5 as an example, we compute the sampling quantity of each sampling tree in the sampling forest as follows:

$$\begin{aligned} \Omega^{\text{APA}} &= \{\varpi^{\text{APA}}(a_1), \varpi^{\text{APA}}(a_2), \varpi^{\text{APA}}(a_3), \varpi^{\text{APA}}(a_4)\} \\ &= \{\lceil \kappa \cdot w_{a_1}^{\text{APA}} \rceil, \lceil \kappa \cdot w_{a_2}^{\text{APA}} \rceil, \lceil \kappa \cdot w_{a_3}^{\text{APA}} \rceil, \lceil \kappa \cdot w_{a_4}^{\text{APA}} \rceil\} \\ &= \{\lceil \kappa \cdot 0.154 \rceil, \lceil \kappa \cdot 0.154 \rceil, \lceil \kappa \cdot 0.462 \rceil, \lceil \kappa \cdot 0.462 \rceil\}. \end{aligned}$$

If $\kappa = 17$, the sampling quantity of each sampling tree is calculated as $\{\lceil 2.618 \rceil, \lceil 2.618 \rceil, \lceil 7.854 \rceil, \lceil 7.854 \rceil\} = \{3, 3, 8, 8\}$.

Transition Probability of Meta-path-based Random Walks. According to a \mathcal{P}_m , a transition probability matrix $\mathcal{A}^{\mathcal{P}_m}$ consists of the transition probabilities between every two nodes of a graph G under certain conditions. We define $\mathcal{A}^{\mathcal{P}} = \{\mathcal{A}^{\mathcal{P}_1}, \mathcal{A}^{\mathcal{P}_2}, \dots, \mathcal{A}^{\mathcal{P}_{|\mathcal{P}|}}\}$, $\mathcal{A}^{\mathcal{P}_m} \in \mathcal{A}^{\mathcal{P}}$. When random walks travel to a node v_i^l , the transition probability is defined as follows:

$$p(v_j|v_i^l, \mathcal{P}_m) = \begin{cases} \frac{w_{v_j}^{\mathcal{P}_m}}{\sum_{v_k \in N^{l+1}(v_i^l)} w_{v_k}^{\mathcal{P}_m}}, & (v_j, v_i^l) \in E \text{ and } \phi(v_j) = l+1 \\ 0, & (v_j, v_i^l) \in E \text{ and } \phi(v_j) \neq l+1 \\ 0, & (v_j, v_i^l) \notin E \end{cases} \quad (5)$$

where \mathcal{P}_m is a meta-path, l and $l+1$ are entity types in the \mathcal{P}_m , $N^{l+1}(v_i^l)$ (refer to Definition 5) denotes the meta-path-based neighbours of node v_i^l , and $w_{v_j}^{\mathcal{P}_m}$ (refer to Equation (2)) is the weight of node $v_j \in N^{l+1}(v_i^l)$.

For example, when random walks travel to a node p_2^P , P is the entity type of node p_2 , as shown in Figure 5, the meta-path-based neighbours of node p_2^P are $N^{l+1}(v_i^l) = N^A(p_2^P) = \{a_2, a_3, a_4\}$, and the probabilities of transition to the subsequent nodes are computed as follows:

$$\begin{aligned} p(v_j|v_i^l, \mathcal{P}_m) &= p(a_2|p_2^P, APA) \\ &= \frac{w_{a_2}}{w_{a_2}+w_{a_3}+w_{a_4}} = \frac{0.154}{0.154+0.462+0.462} = 0.142, \\ p(a_3|p_2^P, APA) &= \frac{w_{a_3}}{w_{a_2}+w_{a_3}+w_{a_4}} = 0.428, \\ p(a_4|p_2^P, APA) &= 0.428. \end{aligned}$$

WRWE. Given the sampling quantity of meta-path instances and the transition probability of meta-path-based random walks, the algorithm of WRWE is presented in Algorithm 1.

Algorithm 1 has three parameters \mathcal{P} , $\phi^{-1}(T_1)$, and L . $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{|\mathcal{P}|}\}$ is a set where in each element \mathcal{P}_m ($1 \leq m \leq |\mathcal{P}|$) is a meta-path $\mathcal{P}_m = T_1, T_2, \dots, T_L$ defining the entity types and relational semantics between nodes of random walks. $\phi^{-1}(T_1)$ is a starting node-set of random walks-based paths. L is the length of the meta-path \mathcal{P}_m , the length of the sample meta-path instances, and the length of random walks-based paths. Shown in Algorithm 1, the algorithm of WRWE outputs *embedding_dict* (line 18 of Algorithm 1), which is an initial weighted embedding-set of nodes and is input to **R-HGNN** (shown in Section 4.3). The *embedding_dict* is generated by the model *MetaPath2Vec* [4] (line 17), which is a graph embedding model based on random walks for a heterogeneous graph.

The first loop (line 3) selects a meta-path \mathcal{P}_m and ensures the consistency of sampling node sequences and the meta-path node types. Line 4 is used to com-

Algorithm 1: WRWE

Input: $G = (V, E, T, R)$: a heterogeneous graph;
 $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{|\mathcal{P}|}\}$: all meta-paths for G;
 $\phi^{-1}(T_1)$: a starting node-set of meta-path instances;
 L : a length of meta-path $\mathcal{P}_m \in \mathcal{P}$;

Output: *embedding_dict* representing the node embedding-set and inputted into graph neural network;

```

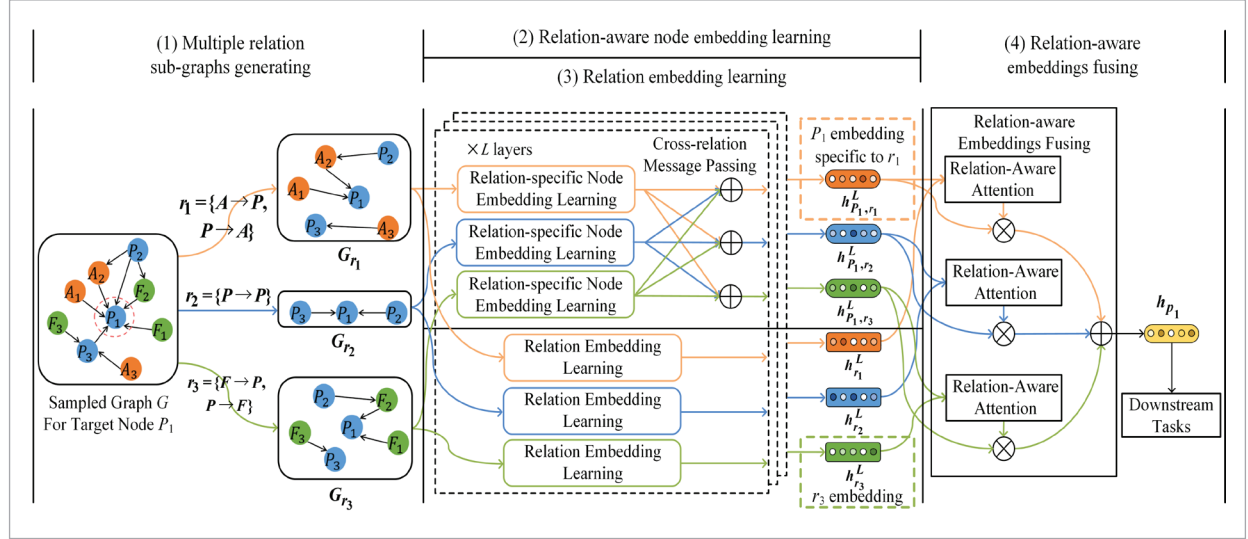
1  $X = []$ ;
2  $embedding\_dict = \{\}$ ;
3 for  $\mathcal{P}_m \in \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{|\mathcal{P}|}\}$  do
4   Compute the transition probability matrix  $\mathcal{A}^m$ , each element of
   which is the  $p(v_j|v_i^l, \mathcal{P}_m)$ , shown in Eq. 5;
5   Calculate all sampling quantities of a sampling forest  $\Omega^{\mathcal{P}_m} =$ 
    $\{\varpi^{\mathcal{P}_m}(v_1), \varpi^{\mathcal{P}_m}(v_2), \dots, \varpi^{\mathcal{P}_m}(v_{|\phi^{-1}(T_1)|})\}$  by Eq. 3;
6   for  $\varpi^{\mathcal{P}_m}(v) \in \{\varpi^{\mathcal{P}_m}(v_1), \varpi^{\mathcal{P}_m}(v_2), \dots, \varpi^{\mathcal{P}_m}(v_{|\phi^{-1}(T_1)|})\}$  do
7     for  $j \in \{1, \dots, \lceil \varpi^m(v) \rceil\}$  do
8        $\rho = v$ ;
9       for  $l \in \{1, \dots, L-1\}$  do
10        Select next node  $v' \in N^{l+1}(v^l)$  according to the
        transition probability matrix  $\mathcal{A}^{\mathcal{P}_m}$ ;
11        Add the  $v'$  to a sample-path sequence  $\rho$ ;
12      end
13       $X[\mathcal{P}_m].append(\rho)$ ;
14    end
15  end
16 end
17 Add the  $X$  to MetaPath2Vec and output the embedding_dict;
18 return(embedding_dict);
```

pute a transition probability matrix $\mathcal{A}^{\mathcal{P}_m}$ according to Equation (5). Line 5 calculates the sampling quantity according to Equation (3). The second loop (line 6) traverses every sampling quantity $\varpi^{\mathcal{P}_m}(v)$ (Equation (3)) in the sampling forest $\Omega^{\mathcal{P}_m}$ (Equation (4)). In line 8, we take the node $v (v \in \phi^{-1}(T_1))$ as the first node of every sample meta-path instances ρ . The third and fourth loops are used to sample meta-path instances for every meta-path according to the transition probability matrix $\mathcal{A}^{\mathcal{P}_m}$ (Equation (5)). In line 17, the sample-path set X is fed to *MetaPath2Vec* to get the initial weighted node embedding *embedding_dict*, which will be given to GNNs.

4.3. Relation-aware Heterogeneous Graph Neural Network (R-HGNN)

The initial weighted node embeddings generated by the WRWE (shown in Algorithm 1) are input into a relation-aware heterogeneous graph neural network

Figure 6

Framework of R-HGNN for Learning Node Embedding h_{P_1} 

(*R-HGNN*) [31]. *R-HGNN* is a GNNs [21], which is used to further learn the relational semantic features of heterogeneous neighbours by propagation mechanism, then outputs final node embedding for each node $v \in G$. *R-HGNN* learns node embeddings in four steps (shown in Fig. 6): (1) multiple relations sub-graphs generating, (2) relation-aware node embedding learning, (3) relation embedding learning and (4) relation-aware embedding fusing.

Multiple Relations Sub-graphs Generating. The heterogeneous graph G is decomposed into multiple relations-specific sub-graphs G_r ($r \in R$) based on the relation types R .

Taking the sampled graph $G = (V, E, T, R)$ as an example (shown in Figure 6(1)), the relation type R of target node P_1 is shown by $R = \{r_1, r_2, r_3\}$, where $r_1 = \{A \rightarrow P, P \rightarrow A\}$, $r_2 = \{P \rightarrow P\}$ and $r_3 = \{F \rightarrow P, P \rightarrow F\}$. The sub-graph set is $G_r = \{G_{r_1}, G_{r_2}, G_{r_3}\}$.

Relation-aware Node Embedding Learning. Relation-aware node embedding learning is implemented by relation-specific node embedding learning and cross-relation message passing. Relation-specific node embedding learning is realized by a dedicated graph convolution module designed to learn node embeddings from each relation-specific sub-graph G_r ($r \in R$). Cross-relation message passing amends relation-specific node embedding by considering the interaction effects of the different relations associated with the node.

Taking Figure 6 as an example, according to the sub-graphs G_{r_1} , G_{r_2} and G_{r_3} , P_1 node embeddings h_{P_1, r_1}^L , h_{P_2, r_2}^L and h_{P_3, r_3}^L are obtained by relation-aware node embedding learning.

Relation Embedding Learning. The target node's relation embeddings are learned layer-wise by the propagation mechanism, which can be formalized as

$$\mathbf{h}_{\psi(e)}^l = \mathbf{W}_{\psi(e), upd}^l \mathbf{h}_{\psi(e)}^{l-1} + \mathbf{b}_{\psi(e), upd}^l, \quad (6)$$

where $\mathbf{W}_{\psi(e), upd}^l$ and $\mathbf{b}_{\psi(e), upd}^l$ are the trainable parameters to update the embeddings \mathbf{h} for relation $\psi(e)$ at layer l .

In Figure 6, $h_{r_1}^L$, $h_{r_2}^L$ and $h_{r_3}^L$ are the relation embeddings of node P_1 . The $\psi(e)$ of node P_1 indicates one of, $\psi(A_2 \rightarrow P_1) = r_1$, $\psi(A_1 \rightarrow P_1) = r_1$, $\psi(P_3 \rightarrow P_1) = r_2$, $\psi(P_2 \rightarrow P_1) = r_2$, $\psi(F_2 \rightarrow P_1) = r_3$ and $\psi(F_1 \rightarrow P_1) = r_3$.

Relation-aware Embeddings Fusing. $\mathcal{R}(v)$ (refer to Definition 1) is a relation type set that node v associates with, and L means aggregating information to node v from multi-hop neighbours by stacking L layers GNNs. So a relation-aware embedding fusing aggregates the relation-aware node embeddings $h_{v, r}^L$ ($r \in \mathcal{R}(v)$) into a compact embedding via the following equations,

$$\gamma_{v, r} = \frac{\exp\left(\text{LeakyReLU}\left(\left(\mathbf{V}_r \mathbf{h}_{v, r}^L\right)^\top \mathbf{E}_r \mathbf{h}_r^L\right)\right)}{\sum_{r' \in \mathcal{R}(v)} \exp\left(\text{LeakyReLU}\left(\left(\mathbf{V}_{r'} \mathbf{h}_{v, r'}^L\right)^\top \mathbf{E}_{r'} \mathbf{h}_{r'}^L\right)\right)}, \quad (7)$$

$$h_v = \sum_{r \in \mathcal{R}(v)} \gamma_{v,r} \cdot V_r h_{v,r}^L \quad (8)$$

where $\gamma_{v,r}$ denotes the learned importance of relation r to final node embedding h_v . V_r is the transformation matrix of node embedding $h_{v,r}^L$ and E_r is the transformation matrix of relation embedding $h_r^L (r \in \mathcal{R}(v))$.

Taking Figure 6 as an example, $R(P_1) = \{r_1, r_2, r_3\}$, $h_{v,r}^L \in \{h_{E_1,r_1}^L, h_{E_1,r_2}^L, h_{E_1,r_3}^L\}$ and $h_r^L \in \{h_{r_1}^L, h_{r_2}^L, h_{r_3}^L\}$.

4.4. Loss Function

After finishing the ultimately model training, we feed the learned node embeddings into a classifier (e.g., a single-layer neural network) to predict the node classification. The purpose of the classifier is to minimize the following cross-entropy loss:

$$Loss = - \sum_{v \in V_{label}} \sum_{c=1}^C y_{v,c} \log \hat{y}_{v,c} \quad (9)$$

where C is the node class set, V_{label} represents the set of labelled nodes, $y_{v,c}$ denotes the ground truth, and $\hat{y}_{v,c}$ denotes the predicted value of the classifier for node v .

5. Experiments

This section describes the datasets and baseline models and evaluates the performance of the proposed method by experiments on node classification tasks.

5.1. Datasets

We consider the following real-world graph datasets: a small-scale dataset (IMDB), two large-scale datasets, OAG (OAG-Venue, OAG-L1-Field) and OGB, shown in Table 2.

The Internet Movie Database (IMDB)² is the world's most popular and authoritative data source for movie, TV and celebrity content. To test our model, we extract a subset of IMDB and construct a heterogeneous graph containing three types of entities (movies (M), directors (D) and actors (A)) and two types of relations (a movie "is directed by" a director and a movie "is acted by" actors). The dataset is split by the random split strategy following reference [28].

The Open Academic Graph (OAG)³ is the largest pub-

² <https://developer.imdb.com>

³ www.openacademic.ai/oag

Table 2

Statistics of the real-world heterogeneous graph datasets

Name	Datasets		OAG		OGB
	--	IMDB	Venue	L1-Field	MAG
Nodes	Movie (M): 4,076	Paper (P): 166,065	Paper (P): 119,483	Paper (P): 736,389	
	Director (D): 1,999	Author (A): 510,189	Author (A): 510,189	Author (A): 1,134,649	
	Actor (A): 5,069	Field (F): 45,717	Venue (V): 6,934	Field (F): 59,965	
		Institution (I): 9,079	Institution (I): 9,079	Institution (I): 8,740	
Edges	MD: 4,076	PA: 477,676	PA: 340,959	PA: 7,145,660	
	MA: 12,228	PP: 851,644	PP: 329,703	PP: 5,416,271	
		PF: 1,700,497	PV: 119,483	PF: 7,505,078	
		AI: 612,872	AI: 612,872	AI: 1,043,998	
Split Strategy	Random Split	Time-based Split	Time-based Split	Time-based Split	
Split Sets	Train: 817	Train: 106,058	Train: 81,071	Train: 629,571	
	Validation: 407	Validation: 24,255	Validation: 16,439	Validation: 64,879	
	Test: 2,852	Test: 35,752	Test: 21,973	Test: 41,939	
meta-paths	MDM, MAM	PPP, APA, AIA, PFP	APA, AIA, PVP, PPP	PPP, APA, AIA, PFP	

licly available heterogeneous academic graph, consisting of more than 178 million nodes and 2.236 billion edges. All papers in OAG are associated with their publication dates, spanning from 1900 to 2019.

We construct two domain-specific subgraphs from OAG [13, 31] (OAG-Venue and OAG-L1-Field). OAG-Venue [31] is a heterogeneous graph in the Computer Science (CS) domain and consists of four types of entities (paper (P), authors (A), fields (F) and institutions (I)), four types of entity-relations (a paper “is written by” an author, a paper “cites” a paper, a paper “has a topic of” a field of study and an author “is affiliated with” an institution). OAG-L1-Field [31] is another heterogeneous graph in the CS domain, which contains four types of entities (papers (P), authors (A), venues (V) and institutions (I)) and four types of entity-relations (a paper “is written by” an author, a paper “cites” a paper, a paper “is published in” a venue and an author “is affiliated with” an institution). The split strategy in OAG-L1-Field is the same as in OAG-Venue.

The Open Graph Benchmark (OGB)⁴ is a collection of realistic, large-scale, and diverse benchmark datasets for machine learning on graphs. The OGB [12] includes OGB-PRODUCTS, OGB-PROTEINS, OGB-ARXIV, OGB-PAPERS100M and OGB-MAG, where OGB-MAG is a heterogeneous academic graph extracted from the Microsoft Academic Graph (MAG), consisting of four types of entities (paper (P), authors (A), fields (F) and institutions (I)). Four types of directed relations in OGB-MAG are the same in OAG-Venue. For experiments, we use MAM and MDM as meta-paths on IMDB, PPP, APA, AIA and PFP as meta-paths on OAG-Venue and OGB-MAG, and APA, AIA, PVP and PPP as meta-paths on OAG-L1-Field.

⁴ <https://ogb.stanford.edu>

5.2. Baselines

R-WHGE is an improved model based on R-HGNN [31], so I developed the WRWE algorithm (refer to Algorithm 1) based on R-HGNN to extract the heterogeneous graphs’ structural, semantic and node importance features. To validate the performance of our approach, we leverage R-HGNN and MetaPath2Vec as baselines to compare with our model based on the data of R-HGNN (refer to Table 2).

R-HGNN learns node embeddings on heterogeneous graphs at a fine-grained level by considering relation-aware features. MetaPath2Vec [4] formalizes meta-path-based random walks to generate the node-sequence set, which is input into a heterogeneous skip-gram model to perform node embeddings of heterogeneous graphs.

5.3. Evaluation Metrics and Parameter Settings

Following [28], the dataset of IMDB is split into training sets, validation sets and testing sets with the ratio of 2:1:7 by the random split strategy. Following [12], the split strategy on OGB-MAG is based on the paper published years, where papers published before 2018, in 2018 and after 2018 are divided into training sets, validation sets and testing sets. Following [13], the split strategy on OAG-Venue and OAG-L1-Field is based on the paper published years. Papers published before 2015, between 2015 and 2016, and after 2016 are divided into training, validation and testing sets. For experiments, we evaluate the effectiveness and efficiency of our model on the node classification task. It involves the prediction of the category of a movie (IMDB), the field that a paper belongs to (OAG-L1-

Table 3

Comparisons with different methods on the node classification task

Methods	-- Datasets Metrics	-- IMDB	OAG		OGB
			Venue	L1-Field	MAG
MetaPath2Vec	Accuracy	0.6334	0.2437	0.5673	0.4643
	Macro-F1	0.6345	0.2045	0.3655	0.2644
R-HGNN	Accuracy	0.6419	0.2872	0.5897	0.5191
	Macro-F1	0.6413	0.2579	0.3981	0.3213
R-WHGE	Accuracy	0.6475	0.2916	0.6032	0.5329
	Macro-F1	0.6439	0.2618	0.4135	0.3392

Field) and the published venue of a paper (OGB-MAG and OAG-Venue). We input the learned node embeddings into a classifier to get the final predictions, shown in Equation (9). We take Accuracy and Macro-F1 as evaluation metrics. The better models have higher evaluation metrics. We run all the models ten times and report the averaged performance in Table 3.

5.4. Experimental Results

The Accuracy and Macro-F1 over four datasets are shown in Table 3, from which several conclusions can be summarized.

R-WHGE outperforms the baselines on the four datasets. The improvements are more significant on large-scale datasets. The main reason for the improvements is that the R-WHGE considers the weight of nodes when extracting structural features.

Specifically, R-WHGE achieves better effectiveness

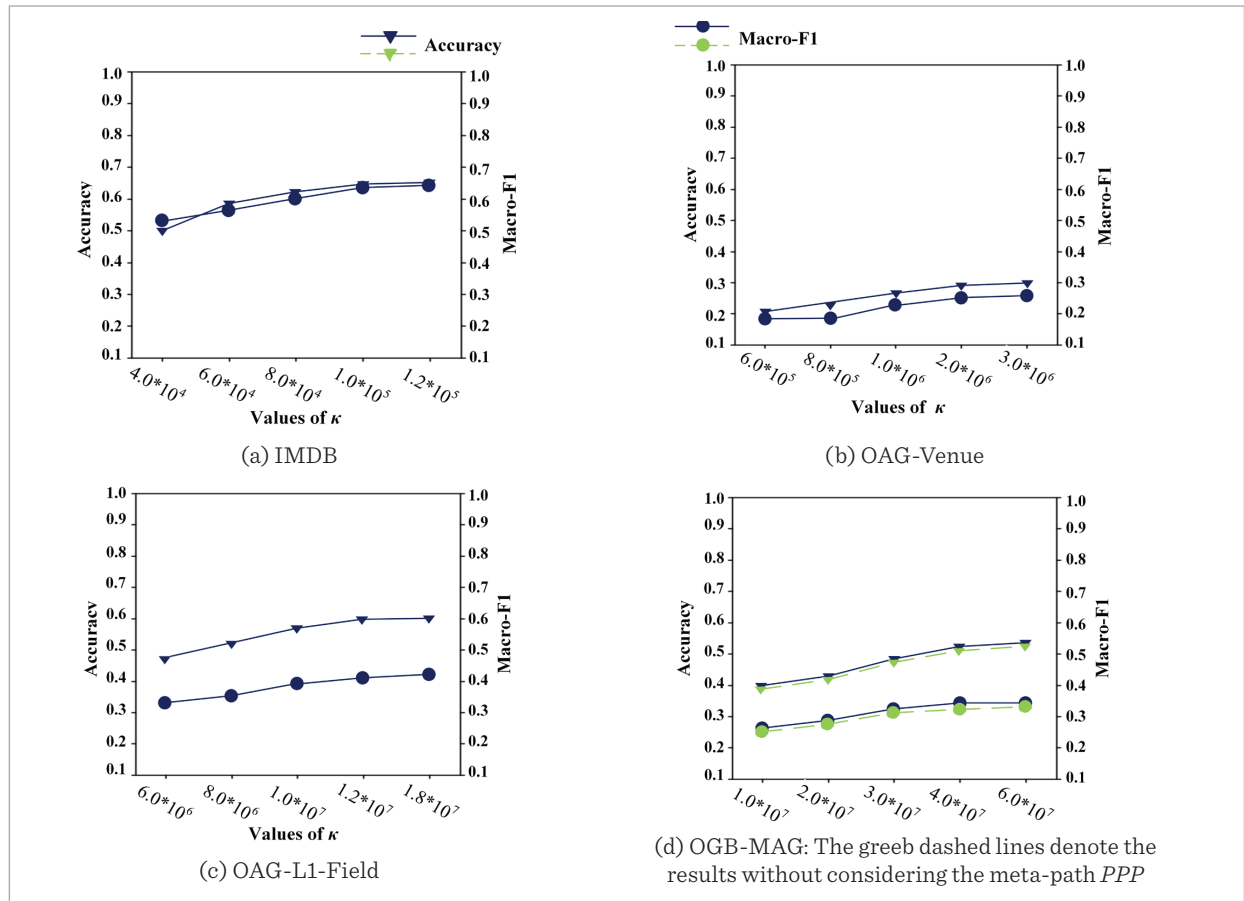
over R-HGNN because R-WHGE fetches structural and semantic features of graph and node's importance features by MetaPath2Vec with weighted random walks and generates the initial weighted node embedding as initial input to GNNs. In summary, these results demonstrate the superiority of R-WHGE, as it considers the heterogeneity and weight of nodes.

5.5. Parameter Sensitivity Analysis

We further perform parameter sensitivity analysis in this section, and the results are summarized in Figure 7. Specifically, we estimate how different the values of κ can affect the classification results. According to Equation (3), the sampling quantity of meta-path instances is proportional to κ . As Shown in Figure 7, we can see that the performances of the *R-WHGE* grow with the increment of the values of κ . It can obtain the best performances when the val-

Figure 7

Results of parameter sensitivity analysis



ues of κ continue to increase to critical values. After that, the performances increase very slowly or even stop growing with an increase of κ . In the subgraph (d) OGB-MAG of Figure 7, subtracting meta-path PPP has a slight influence on the classification effect of node-type Page, under the premise that all nodes of graph OGB-MAG can be trained in the experiment.

6. Conclusion and Future Work

This paper has presented a practical graph embedding framework R-WHGE, which can easily incorporate heterogeneous topology features, relationship characteristics among heterogeneous nodes, and weights of heterogeneous nodes into graph embedding. To acquire weights of heterogeneous nodes, the algorithm of het-PageRank based on meta-paths is designed to get the importance of the node's location in the network, and then these importances are used as the node's weights. Utilizing the node's weights, we propose the algorithm of weighted random walks, which

can extract topological and relation semantics features by fetching the meta-path instances. The initial weighted node embeddings are generated by MetaPath2Vec based on weighted random walks, which is the input of the algorithm of R-HGNN. R-HGNN aggregates neighbour features considering the surrounding specific relations of each node and gets the final relation-aware node embeddings. Experimental results show the effectiveness of embedding vectors in node classification. In the future, we plan to develop methods of automatically learning meta-paths and R-WHGE to exploit the heterogeneous structures more comprehensively. We will also study the applications of R-WHGE for other tasks, such as link prediction and recommendation.

Acknowledgement

This paper is funded by the National Natural Science Foundation of China (62032019, 61732019, 61672435), and the Capacity Development Grant of Southwest University (SWU116007).

References

1. Ai, B., Qin, Z., Shen, W., Li, Y. Structure Enhanced Graph Neural Networks for Link Prediction, 2022. <https://doi.org/10.48550/arXiv.2201.05293>
2. Busbridge, D., Sherburn, D., Cavallo, P., Hammerla, N. Y. Hammerla. Relational graph Attention Networks. ICLR, 2019, 964. <https://doi.org/10.48550/arXiv.1904.05811>
3. Cai, H., Zheng, V. W., Chang, K. C. C. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2018, 30(9), 1616-1637. <https://doi.org/10.1109/TKDE.2018.2807452>
4. Dong, Y., Chawla, N. V., Swami, A., Claims, A. I. Metapath2vec: Scalable Representation Learning for Heterogeneous Networks. The 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, 135-144. <https://doi.org/10.1145/3097983.3098036>
5. Fu, T.-Y., Lee, W. C., Zhen, L. Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM'17), 2017. <https://doi.org/10.1145/3132847.3132953>
6. Fu, X., Zhang, J., Meng, Z., King, I., Claims, A. I. MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding. WWW ,20: Proceedings of the Web Conference 2020, 2020, 2331-2341. <https://doi.org/10.1145/3366423>
7. Goyal, P., Ferrara, E. Graph Embedding Techniques, Applications, and Performance: A Survey. Knowledge Based Systems, 2018, 151, 78-94. <https://doi.org/10.1016/j.knosys.2018.03.022>
8. Grover, A., Leskovec, J. Node2vec: Scalable Feature Learning for Networks. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, vol. 2016, 855-864. <https://doi.org/10.1145/2939672.2939754>
9. Hamilton, W. L., Ying, R., Leskovec, J. Inductive Representation Learning on Large Graphs. 31st Conference on Neural Information Processing Systems (NIPS), 2017. <https://doi.org/10.48550/arXiv.1706.02216>
10. Haveliwala, T. H. Topic-sensitive Pagerank: A Context-sensitive Ranking Algorithm for Web Search. IEEE Transactions on Knowledge and Data Engineering, 2003, 15, 784-796. <https://doi.org/10.1109/TKDE.2003.1208999>

11. Hong, H., Guo, H., Lin, Y., Yang, X., Li, Z., Ye, J. An Attention-based Graph Neural Network for Heterogeneous Structural Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, 34, 4132-4139. <https://doi.org/10.1609/aaai.v34i04.5833>
12. Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., Leskovec, J. Open Graph Benchmark: Datasets For Machine Learning On Graphs. *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020. <https://doi.org/10.48550/arXiv.2005.00687>
13. Hu, Z., Dong, Y., Wang, K., Sun, Y. Heterogeneous Graph Transformer. *Proceedings of the Web Conference 2020 (WWW 2020)*, 2020, 2704-2710. <https://doi.org/10.1145/3366423.3380027>
14. Ji, H., Shi, C., Wang, B. Attention Based Meta Path Fusion for Heterogeneous Information Network Embedding. *PRICAI 2018: Trends in Artificial Intelligence*, 2018, 11012, 348-360. https://doi.org/10.1007/978-3-319-97304-3_27
15. Kipf, T. N., Welling, M. Semi-supervised Classification with Graph Convolutional Networks. *ICLR*, 2017. <https://doi.org/10.48550/arXiv.1609.02907>
16. Li, X., Shang, Y., Cao, Y., Li, Y., Liu, Y. Type-aware Anchor Link Prediction Across Heterogeneous Networks Based on Graph Attention Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, 34(01), 47-155. <https://doi.org/10.1609/aaai.v34i01.5345>
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. *NIPS'13: Proceedings of the 26th International Conference on Neural Information Processing Systems*, 2013, 2, 3111-3119. <https://doi.org/10.48550/arXiv.1310.4546>
18. Page, L., Brin, S., Motwani, R., Winograd, T. The Pagerank Citation Ranking: Bringing Order to the Web. *Stanford Digital Libraries Working Paper*, 1998, 5766. https://doi.org/10.1007/978-3-642-04417-5_3
19. Perozzi, B., Al-Rfou, R., Skiena, S. Deepwalk: Online Learning of Social Representations. *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*, 2014, 701-710. <https://doi.org/10.1145/2623330.2623732>
20. dos Santos, L., Piwowski, B., Denoyer, L., Gallinari, P. Representation Learning for Classification in Heterogeneous Graphs with Application to Social Networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2018, 12(5), 1-33. <https://doi.org/10.1145/3201603>
21. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, 2009, 20(1), 61, 61-80. <https://doi.org/10.1109/TNN.2008.2005605>
22. Schlichtkrull, M., Kipf, T.N., Bloem, P., Berg, R.v.d., Titov, I., Welling, M. Modeling Relational Data with Graph Convolutional Networks. *The Semantic Web*, 2018, 10843, 593-607. https://doi.org/10.1007/978-3-319-93417-4_38
23. Shi, C., Hu, B., Zhao, W. X., Yu, P. S. Heterogeneous Information Network Embedding for Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2019, 31(2), 357-370. <https://doi.org/10.1109/TKDE.2018.2833443>
24. Shi, C., Li, Y., Zhang, J., Sun, Y., Yu, P. S. A Survey of Heterogeneous Information Network Analysis. *IEEE Transactions on Knowledge & Data Engineering*, 2017, 29(1), 17-37. <https://doi.org/10.1109/TKDE.2016.2598561>
25. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q. Line: Large-scale Information Network Embedding. *Proceedings of the 24th international Conference on World Wide Web (WWW'15)*, 2015, 1067-1077. <https://doi.org/10.1145/2736277.2741093>
26. Veličković, P., Cucurull G., Casanova A., Romero A., Liò P., Bengio, Y. Graph Attention Networks. *International Conference on Learning Representations (ICLR 2018)*, 2018. <https://doi.org/10.48550/arXiv.1710.10903>
27. Wang, X., Bo, D., Shi, C., Fan, S., Ye, Y., Yu, P. S. A Survey on Heterogeneous Graph Embedding: Methods, Techniques, Applications and Sources. *IEEE Transactions on Big Data*, 2022, 1-1. <https://doi.org/10.1109/TBDATA.2022.3177455>
28. Wang, X., Ji, H., Shi, C., Wang, B., Cui, P., Yu, P., Ye, Y. Heterogeneous Graph Attention Network. *The World Wide Web Conference (WWW'19)*, 2019, 2022-2032. <https://doi.org/10.1145/3308558.3313562>
29. Xu, S., Yang, C., Shi, C., Fang, Y., Guo, Y., Yang, T., Zhang, L., Hu, M. Topic-aware Heterogeneous Graph Neural Network for Link Prediction. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM'21)*, 2021, 2261-2270. <https://doi.org/10.1145/3459637>
30. Yang, Y., Guan, Z., Li, J., Zhao, W., Cui, J., Wang, Q. Interpretable and Efficient Heterogeneous Graph Convolutional Network. *IEEE Transactions on Knowledge and Data Engineering*, 2021. <https://doi.org/10.1109/TKDE.2021.3101356>
31. Yu, L., Sun, L., Du, B., Liu, C., Lv, W., Xiong, H. Heterogeneous Graph Representation Learning with Relation Awareness. *IEEE Transactions on Knowledge*

- and Data Engineering, 2022. <https://doi.org/10.1109/TKDE.2022.3160208>
32. Yun, S., Jeong, M., Kim, R., Kang, J., Kim, H.J. Graph transformer networks. 33rd Annual Conference on Neural Information Processing Systems (NeurIPS 2019), 2019, 32. <https://doi.org/10.48550/arXiv.1911.06455>
33. Zhang, C., Song, D., Huang, C. HetGNN: Heterogeneous Graph Neural Network. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD,19), 2019, 793-803. <https://doi.org/10.1145/3292500>
34. Zhu, S., Zhou, C., Pan, S., Zhu, X., Wang, B. Relation structure-aware heterogeneous graph neural network. The 19th IEEE International Conference on Data Mining (ICDM-19). IEEE (2019), 2019, 1534-1539. <https://doi.org/10.1109/ICDM.2019.00203>



This article is an Open Access article distributed under the terms and conditions of the Creative Commons Attribution 4.0 (CC BY 4.0) License (<http://creativecommons.org/licenses/by/4.0/>).