Relation Structure-Aware Heterogeneous Graph Neural Network

Shichao Zhu*‡, Chuan Zhou†‡, Shirui Pan§, Xingquan Zhu¶, Bin Wang‖
*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
†Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China
‡School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
§Faculty of Information Technology, Monash University, Melbourne, Australia
¶Dept. of Computer & Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, USA
‖Xiaomi AI Lab, Beijing, China
zhushichao@iie.ac.cn, zhouchuan@amss.ac.cn, shirui.pan@monash.edu, xzhu3@fau.edu, wangbin11@xiaomi.com

***Abstract*—Heterogeneous graphs with different types of nodes and edges are ubiquitous and have immense value in many applications. Existing works on modeling heterogeneous graphs usually follow the idea of splitting a heterogeneous graph into multiple homogeneous subgraphs. This is ineffective in exploiting hidden rich semantic associations between different types of edges for large-scale multi-relational graphs. In this paper, we propose Relation Structure-Aware Heterogeneous Graph Neural Network (RSHN), a unified model that integrates graph and its coarsened line graph to embed both nodes and edges in heterogeneous graphs without requiring any prior knowledge such as meta-path. To tackle the heterogeneity of edge connections, RSHN first creates a Coarsened Line Graph Neural Network (CL-GNN) to excavate edge-centric relation structural features that respect the latent associations of different types of edges based on coarsened line graph. After that, a Heterogeneous Graph Neural Network (H-GNN) is used to leverage implicit messages from neighbor nodes and edges propagating among nodes in heterogeneous graphs. As a result, different types of nodes and edges can enhance their embedding through mutual integration and promotion. Experiments and comparisons, based on semi-supervised classification tasks on large scale heterogeneous networks with over a hundred types of edges, show that RSHN significantly outperforms state-of-the-arts.**

***Index Terms*—heterogenous graph, coarsened line graph, graph neural network**

I. Introduction

The recent success of neural networks on non-Euclidean domain has impelled intensive research on graph embedding and its application to an increasing number of related domains, including chemical drug discovery [1], [2], knowledge graph embedding [3], [4] and etc. Graph Neural Networks (GNNs) in particular are effective techniques to learn graphs directly from the structural level for analyzing the underlying symbolic nature of graphs. To date, GNNs have achieved good results in many graph mining tasks [5], [6], but most of them focus on homogeneous graphs that assumed nodes and edges in the network are of the same type. In reality, heterogeneous graphs are ubiquitous and have immense application value [7], [8], because heterogeneous networks allow nodes and edges to have different types which are more realistic in characterize semantic relations between objects.

Different from homogeneous graphs, modeling heterogeneous graphs with GNNs typically suffers from two challenges:

- *Challenge 1: Rich semantic relations:* A heterogeneous graph has complex multi-type relations. We need to model and characterize semantic relations for large complex heterogeneous networks.
- *Challenge 2: Joint node and edge embedding:* Because both nodes and edge are playing important roles in heterogeneous networks, we need to jointly learn discriminative embedding for nodes and edges with maximum performance gain.

A handful of recent efforts, including HAT [9] and R-GCN [4], have studied heterogeneous graph embedding using graph neural networks. The basic idea of these models is to split a heterogeneous graph into multiple homogeneous subgraphs. HAT relies on meta-path [10] to extract subgraphs and then employs attention mechanism with graph neural network to embed heterogeneous graphs. However, for highly multi-relational data, meta-path based methods are not readily applicable, due to its high dependence and cost on constructing meta paths. R-GCN can be applied to highly multi-relational data that is primarily motivated as an adaption of previous works on GCN [11], and it splits heterogeneous graph to multiple subgraphs by building an independent adjacency matrix for each type of edge.

The above methods are typically node-centric graph neural networks that aggregate information from neighbor nodes to learn the representation and may partially address the *Challenge 2*. However, they are ineffective in exploiting rich semantic associations between different types of edges potentially hidden in a graph (*Challenge 1*). This is because although they take into account the different types of edges by introducing different matrices or weights, they still go along the idea of modeling homogeneous graphs. As a matter of fact, the neglected relation structure-aware information can further help to mine the underlying relational features between different types of edges, not just node-centric structural information.

To overcome the limitation of existing algorithms, in this

2374-8486/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDM.2019.00203

paper we propose Relation Structure-aware Heterogeneous Graph Neural Network (RSHN), a novel approach of modeling heterogeneous graph that takes into account multi-relation associations and enables implicit messages propagating among nodes in heterogeneous graphs. RSHN consists of two components: Coarsened Line Graph Neural Network (CL-GNN) and Heterogeneous Graph Neural Network (H-GNN). The former focuses on representing different types of edges with relational attention mechanisms that respect implicit associations of different types of edges (for *Challenge 1*), and the latter further enables neighbor node and edge messages propagating among nodes in heterogeneous graphs by coupling with edge type features (for *Challenge 2*).

Inspired by line graph [12], where each node represents an edge of graph and two nodes of line graph are adjacent iff their corresponding edges share a common endpoint in graph (as shown in Fig. 2), our coarsened line graph neural network employs an edge-to-node dual learning style for graph modeling that greatly reduces the size of line graph and still covers the relevance of different types of edges. Specifically, we build a coarsened line graph via random walks over original graph to alleviate such problems and mine potential associations between different types of edges. In the coarsened line graph, nodes represent different types of edges and weights on edges represent co-occurrence rates of two different types of edges. The construction of coarsened line graph is sketched in Fig. 1. We assume that the higher the co-occurrence rate of different types of edges, the more relevant they are. In other words, the coarsened line graph is developed to automatically excavate edges types that are highly correlated to each other. Based on the constructed coarsened line graph, CL-GNN

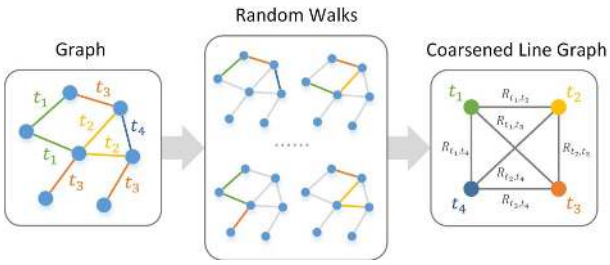

Fig. 1: An example of Coarsened Line Graph (right panel) construction from an input graph (left panel). Edges are color coded based on the types of edges ($t_1, t_2, \cdots$). Coarsened line graph uses edge types of the original graph as nodes. Edge weight in the coarsened line graph represents co-occurrences between two types of edges in random walks (R_{t_1,t_2} denotes co-occurrences between edge type t_1 and t_2 in random walks). This design tremendously reduces the number of nodes of the line graph, and also accurately captures edge relations.

module is designed to embed the coarsened line graph with a novel relational attention propagation layer. After that, H-GNN module takes edge types embedding learned by CL-GNN as input and integrates messages from neighbor nodes and edges to update central node. Both of the two modules are jointly learned and optimized in a unified framework to achieve optimal performance for heterogeneous graph modeling. The main contributions of the paper are as follows.

- To the best of our knowledge, this is the first endeavor exploring and integrating associations between different types of edges in heterogeneous graphs using graph neural network. In comparison, existing methods dissect heterogeneous networks as multiple homogeneous networks, so cannot fully excavate interactions and rich semantic relation between edges.
- We propose a unified model Relation Structure-Aware Heterogeneous Graph Neural Network (RSHN) that utilize graph structure and implicit relation structural information to simultaneously learn node and edge type embedding.
- The proposed model is independent of user-defined heuristics, such as meta-path, and is effective in dealing with a large number of complex relations.
- Experiments show that our model considerably outperforms all baselines for classification task on four large-scale multi-relational networks.

In the remainder of the paper, we first review related work in Section II, and then detail the proposed approach in Section III. Experiments and results are reported in Section IV, followed by the conclusion in Section V.

II. Related Work

Our research is related to (i) Graph neural networks (GNNs), (ii) Heterogeneous graph embedding, and (iii) Line graph.

Graph Neural Networks (GNNs). Graphs are ubiquitous in the real world that are considered to have rich semantical and structural information. Therefore, Graph Neural Networks (GNNs) have sprung up to utilize deep learning methods for graph data that learn the target nodes representation by propagating neighbor information via neural networks. And some surveys on GNNs have been proposed so far along with immense applications using GNNs as a tool [13]–[15]. For instance, in chemistry, GNNs are adopted to model molecules as graphs to excavate its unknown properties and discover new drugs [2]. In knowledge graphs, GNNs framework has achieved good performance for knowledge graph embedding tasks in terms of scalability and efficiency, such as entity classification [4]. However, most of these methods ignore the unique information of heterogeneous graphs, so we try to explore heterogeneous graphs modeling with GNNs in this paper.

Heterogeneous Graph Embedding. Heterogeneous graph is a kind of graph with multiple types of nodes and edges that is more ubiquitous in our real life. Heterogeneous graph embedding is proposed to embed heterogeneous graph into a low dimensional space while preserving the structure and property. The existing methods to this task can be roughly divided into shallow models [8], [16] and deep models [4], [7], [9]. Superior to shallow models, most of these deep models are based on graph neural networks, which have the effectiveness of deep feature exploration and some of them have achieved

state-of-the-art performance. As a result, this paper focuses on graph neural networks.

As we can see, there are a lot of methods mainly focused on preserving the meta-path [10] based structural information to embed heterogenous graph such as HAT [9], while these methods can only be used for datasets that have a small number of different types of edges that make it easy to build meta paths. For highly multi-relational data, R-GCN [4] is a representative work that divides different types of edges into independent adjacency matrices, which means that there are no direct correlations between them. In response to these limitations, we propose a more powerful model RSHN to capture deep relational structure-aware features for highly multi-relational data.

Line Graph. Line graph [12] is an edge-centric graph that represents the adjacency between edges of graph G, where each node of line graph represents an edge of G and two nodes of line graph are adjacent if and only if their corresponding edges share a common endpoint in G (see Fig. 2). There are also some works that adopt line graph, such as community detection [17] and traffic prediction [18]. It can be seen that line graph will be very large and cannot be constructed and used efficiently for large-scale graphs. Motivated by line graph and its limitation, we build a novel coarsened line graph with different types of edges as nodes that greatly reduces the size of line graph and still covers the relevance of different types of edges.

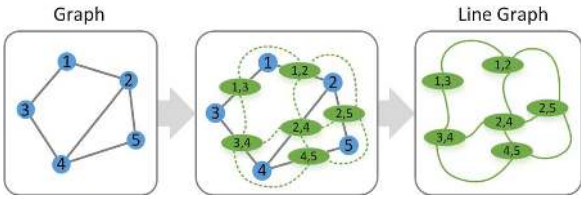

Fig. 2: An example of line graph, which converts the original graph into an edge-centric graph.

III. Proposed Approach

A. Notations and Problem Definition

In this section, we introduce some basic concepts and formalize the problem of heterogeneous graph embedding.

Definition 1. Heterogeneous Graph [19]. A heterogeneous graph is defined as a directed graph $G = (\mathcal{V}, \mathcal{E})$, in which $\mathcal{V}$ are the set of nodes, $\mathcal{E}$ are edges between them. Let $\mathcal{T} := \{1, 2, ...M\}$ denote the set of different edge types. We represent different types of edges in an adjacency matrix $A \in \mathbb{R}^{N \times N}$, $A_{vw} \in \{0\} \cup \mathcal{T}$, where $N := |\mathcal{V}|$ denotes the number of nodes, M denotes the number of different types of edges, A_{vw} are an integer-value that represents the type of directed edge from node v to w. $A_{vw} = 0$ means that there is no edge from node v to w. Heterogeneous graph embedding is to learn two mapping functions $f : \mathcal{V} \to \mathbb{R}^n$, $g : \mathcal{E} \to \mathbb{R}^m$ that project each node v and edge e into low dimensional vectors in spaces $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively.

Definition 2. Coarsened Line Graph. Given a heterogeneous graph G, the coarsened line graph $L(G) = (\mathcal{T}, \mathcal{R})$ is an undirected weighted graph, indicating the correlations between different types of edges, where $\mathcal{T}$ denotes different types $\{1, 2, ...M\}$ of edges from G, and $\mathcal{R}$ denotes the set of weighted edges. Coarsened line graph embedding will learn a type mapping function $h : \mathcal{T} \to \mathbb{R}^m$. In the coarsened line graph $L(G)$, each node $t \in \mathcal{T}$ represents one type of edge, and two nodes t_1, t_2 are linked by a weighted edge, where the weight R_{t_1,t_2} on edge indicates the co-occurrence rate of them.

In order to simplify the model and reduce feature, we aim at modeling different types of edges to represent edges, assuming that the edges of the same type have same feature. Specifically, edge features are extracted based on the following composite function.

$$g(e_{vw}) \quad = \quad h \circ F(e_{vw}) \tag{1}$$

where $F(e_{vw})$ extracts the type of edge e_{vw}, $h : \mathcal{T} \to \mathbb{R}^m$ is the type mapping function and $g : \mathcal{E} \to \mathbb{R}^m$ is the edge mapping function.

B. Modeling Heterogeneous Graph

The proposed Relation Structure-aware Heterogeneous Graph Neural Network (RSHN) contains two modules: Coarsened Line Graph Neural Network (CL-GNN) and Heterogeneous Graph Neural Network (H-GNN). The architecture is summarised in Fig. 3.

1) Coarsened Line Graph Neural Network (CL-GNN): Indeed, learning associations between different types of edges is an unsupervised problem. Our intuition is that edges sharing more common paths have a higher similarity, and hence should be learned jointly with co-occurrence rates. We build a coarsened line graph $L(G) = (\mathcal{T}, \mathcal{R})$, which greatly simplifies the traditional line graph and highlights the degree of correlations between nodes.

For any type t_i and t_j, the co-occurrence rate R_{t_i,t_j} is obtained by counting their occurrence frequency in a sampled batch of associated paths $P := \{p_1, p_2, \ldots, p_{bs}\}$ of graph G, where $bs := |P|$ denotes the $batch\_size$, and each associated path $p_k \in P$ with the length $step\_size$ (ss for short) is defined as a sequence of edges appearing in edge-based random walks $v_1^{(k)} \overset{e_1^{(k)}}{\rightsquigarrow} v_2^{(k)} \cdots \overset{e_{ss-1}^{(k)}}{\rightsquigarrow} v_{ss}^{(k)}$ over graph G, reflecting the associations between different types of edges. And then the co-occurrence rate R_{t_i,t_j} between type t_i and t_j can be counted according to

$$R_{t_i,t_j} = \frac{\#\{k = 1, ..., bs | t_i \text{ and } t_j \text{ appear together in } p_k\}}{bs} \tag{2}$$

where we say 't_i and t_j appear together in p_k' means that there exist at least two edges in path p_k with their types being t_i and t_j respectively. Similarly, node relationships can be introduced using the same way. It is worth noting here that the complexity

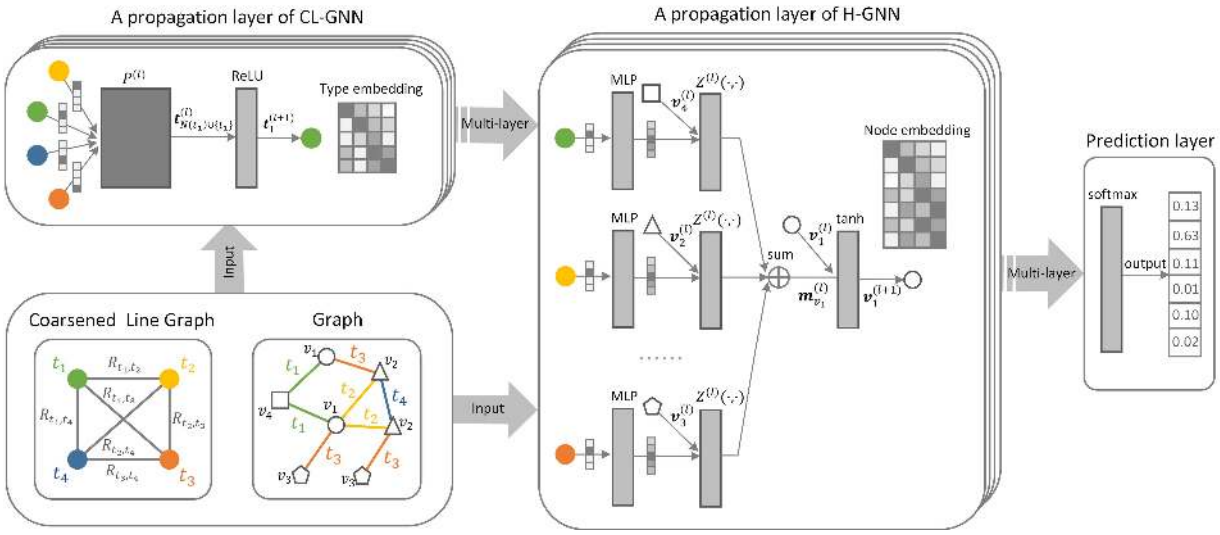

Fig. 3: Schematic of RSHN architecture. (i) Coarsened line graph $L(G)$ is constructed by random walks over graph G; (ii) Taking coarsened line graph $L(G)$ as input, CL-GNN module is designed to learn different types of edges; (iii) Taking original heterogeneous graph G and different types of edges embedding as input, H-GNN module aims to represent nodes with message passing layers; (iv) The prediction layer is applied row-wise on the output of H-GNN's last layer for classification task.

of simple counting statistics is higher than that of random walks and thus it is not efficient here.

CL-GNN module is designed based on Attention Graph Neural Network (AGNN) [20] to learn a type mapping function $h : \mathcal{T} \rightarrow \mathbb{R}^m$ on coarsened line graph $L(G) = (\mathcal{T}, \mathcal{R})$, which adopts attention mechanism over neighbors to learn which neighbors are more relevant and weigh their contributions accordingly. For every propagation layer $l \in \{0, 1, ..., K-1\}$, it convolves type features $t^{(l)}$ using graph propagation matrix $P^{(l)}$ according to

$$\boldsymbol{t}_i^{(l+1)} = \sigma\left(\sum_{t_j \in \mathcal{N}(t_i) \cup \{t_i\}} \left(P^{(l)}_{t_i,t_j} \cdot \boldsymbol{t}_j^{(l)}\right)\right) \quad (3)$$

$$h(t_i) = \boldsymbol{t}_i^{(K)} \quad (4)$$

where σ is an ReLU activation, the propagation matrix $P^{(l)} \in \mathbb{R}^{M \times M}$ is an attention-guided propagation matrix computed as (5) with a trainable parameter $\beta^{(l)}$ at layer l. We use one-hot encoding to initialize type embedding $\boldsymbol{t}^{(0)}$. After K layers, CL-GNN will obtain the final type features $\boldsymbol{t}^{(K)}$.

$$P^{(l)}_{t_i,t_j} = \frac{\exp(\beta^{(l)} \cdot R_{t_i,t_j})}{\sum_{t_k \in \mathcal{N}(t_i) \cup \{t_i\}} \exp(\beta^{(l)} \cdot R_{t_i,t_k})} \quad (5)$$

2) Heterogeneous Graph Neural Network (H-GNN): H-GNN, a novel variant of the Message Passing Neural Network (MPNN) framework [2], aims to learn a node mapping function $f : \mathcal{V} \rightarrow \mathbb{R}^n$ on heterogeneous graph $G = (\mathcal{V}, \mathcal{E})$, where every layer l's forward pass is defined in terms of message function $Z^{(l)}$ and node update function $U^{(l)}$. During each layer $l \in \{0, 1, ..., K'-1\}$, node features $\boldsymbol{v}^{(l)}$ are updated based on messages $\boldsymbol{m}_v^{(l+1)}$ according to

$$\boldsymbol{m}_v^{(l+1)} = \sum_{w \in \mathcal{N}(v)} Z^{(l)}\left(\boldsymbol{w}^{(l)}, h_\Theta\left(g(e_{vw})\right)\right) \quad (6)$$

$$= \sum_{w \in \mathcal{N}(v)} Z^{(l)}\left(\boldsymbol{w}^{(l)}, h_\Theta\left(h\left(F(e_{vw})\right)\right)\right) \quad (7)$$

$$\boldsymbol{v}^{(l+1)} = U^{(l)}\left(\boldsymbol{v}^{(l)}, \boldsymbol{m}_v^{(l+1)}\right) \quad (8)$$

$$= \sigma\left(W_0^{(l)}\boldsymbol{v}^{(l)} + W_1^{(l)}\boldsymbol{m}_v^{(l+1)}\right) \quad (9)$$

$$f(v) = \boldsymbol{v}^{(K')} \quad (10)$$

where e_{vw} is the edge from node v to w, $g(\cdot)$ is the edge mapping function that equals to the composite function $h \circ F(\cdot)$, where h function is learned from CL-GNN. h_Θ denotes a differentiable function with parameter Θ, such as MLPs, which is used to project edge features into node features space, *i.e.* $h_\Theta(\cdot) \in \mathbb{R}^n$. In sum, $N(v)$ denotes the set of first-order neighbours of node v, *i.e.* $N(v) = \{w | v \neq w \vee A_{vw} \neq 0\}$. The message function $Z^{(l)}$ is an integration process that can be typically implemented in several operations, including concatenation, subtraction or multiplication, and produces a message vector $\boldsymbol{m}_v^{(l+1)} \in \mathbb{R}^n$. In update function $U^{(l)}$, σ is an element-wise activation function, $W_0^{(l)}$ and $W_1^{(l)}$ are learned weight matrices. The initial node embedding $\boldsymbol{v}^{(0)}$ for each node v is randomly initialized and after K' layers, H-GNN will get the final node features $\boldsymbol{v}^{(K')}$.

Intuitively, RSHN can capture information about multi-level neighbor nodes and correlation of different types of edges at varying depth by stacking multiple layers of H-GNN and CL-GNN.

C. Model Training

Our approach combines heterogeneous graph with its coarsened line graph to effectively couple the representations of nodes and different types of edges. We apply them to semi-supervised classification task.

At H-GNN's layer l, message function $Z^{(l)}$ is implemented as subtraction that refers to translational distance model [21] and it has achieved the best result in tasks. Our forward model then takes following form as a instance of message function $Z^{(l)}$ and node update function $U^{(l)}$:

$$\boldsymbol{m}_v^{(l+1)} = \sum_{w \in \mathcal{N}(v)} \left(\boldsymbol{w}^{(l)} - h_\Theta \left(g(e_{vw}) \right) \right) \quad (11)$$

$$\boldsymbol{v}^{(l+1)} = \tanh \left(W_0^{(l)} \boldsymbol{v}^{(l)} + W_1^{(l)} \boldsymbol{m}_v^{(l+1)} \right) \quad (12)$$

And then passing through a *softmax* function, defined as $\text{softmax}(h_v) = \frac{1}{\mathcal{Z}} \exp(h_v)$ with $\mathcal{Z} = \sum_v \exp(h_v)$, is applied row-wise on the output of H-GNN's last layer. Both H-GNN and CL-GNN are trained jointly by minimizing the cross-entropy loss over all labeled examples:

$$\mathcal{L} = - \sum_{l \in \mathcal{Y}_L} \sum_{f=1}^{F} Y_{lf} \ln X_{lf} \quad (13)$$

where $\mathcal{Y}_L$ is the set of node indices that have labels, X_{lf} is the f-th entry of the network output for i-th labeled node and Y_{lf} denotes its ground truth label.

D. Complexity Analysis

The proposed model is efficient. Before training RSHN, we construct the coarsened line graph via edge-based random walks with the complexity $O(batch\_size \cdot step\_size)$, which are dependent on a specific dataset. And the time complexity of CL-GNN is $O(MR_{clg}C_{clg})$, where M is the number of different types of edges, R_{clg} and C_{clg} are the numbers of row and column of coarsened line graph transformation matrix, respectively. The time complexity of H-GNN is $O(NR_gC_g)$, where N is the number of nodes, R_g and C_g are the numbers of row and column of graph transformation matrix, respectively. The overall complexity is linear to the number of nodes and different types of edges.

IV. Experiments

A. Benchmark Datasets

For semi-supervised classification task, we consider four benchmark heterogeneous graph datasets: AIFB, MUTAG, BGS, and AM. AIFB [22] describes the AIFB research institute in terms of its staff, research groups, and publications. We try to predict the affiliation for people in AIFB research institute. MUTAG [1] describes the interactions between complex molecules that can be classified as isMutagenic or not. BGS [23] describes information about relations between named rock units that can be classified as hasLithogenesis or not. AM [24] describes connections and details of artifacts in Amsterdam Museum, each of which has one type property. The statistics of datasets are shown in Table I.

[1] http://dl-learner.org

TABLE I: A summary of the benchmark Datasets. For each dataset, we report the number of nodes, number of node types, number of labeled nodes, number of edges, and number of edge types, respectively.

Datasets	Nodes	Types	Labeled	Edges	Types
AIFB	8,285	4	178	29,043	45
MUTAG	23,644	2	340	74,227	23
BGS	333,845	2	146	916,199	103
AM	1,666,764	11	1,000	5,988,321	133

B. Baselines

We compare our model to several state-of-the-art baselines, including the shallow network embedding methods and GNN-based methods. The shallow models include Hand-designed feature extractors (Feat) [25], Weisfeiler-Lehman kernels (WL) [26] and RDF2Vec embeddings [27]. We select two models Graph Attention Network (GAT) [28] and Relational Graph Convolutional Network (R-GCN) [4] as GNN-based baselines. We implement two versions of our model including RSHN and RSHN$_{\neg CL}$, in order to reflect the effectiveness of the coarsened line graph. RSHN$_{\neg CL}$ is a variant of RSHN, which removes the CL-GNN module and represents different types of edges with one-hot vectors.

C. Experimental Settings

We perform training and full-batch optimization of all baselines on the same training set (80%) and test set (20%). We use the Adam [29] optimizer with a learning rate of 0.01 and l_2 penalty on weights $5 \cdot 10^{-4}$. For all baselines, we use the same hidden units as shown below. We use 2-layer H-GNN coupled with 1-layer RLG-NN with 8 hidden units on AIFB and MUTAG datasets, and 2-layer H-GNN coupled with 2-layer RLG-NN with 16 hidden units on BGS dataset. For large dataset AM, we use 2-layer H-GNN coupled with 2-layer RLG-NN with 16 hidden units.

For the construction of coarsened line graph, the $batch\_size$ and $step\_size$ of random walks depend on different datasets. We tune the number of $batch\_size$ in $\{100, 200, ..., 1200\}$ and $step\_size$ in $\{3, 4, 5\}$. Based on the results, we find that performance on different datasets will converge before 1000 batch sizes and 4 step sizes, and hence use the same setting $batch\_size = 1000$ and $step\_size = 4$ on four datasets for simplicity.

D. Results

This task is to classify the node in heterogeneous graph into one type that belongs to. Using accuracy as a protocol to evaluate the performance of classification task, our model has achieved the best performance comparing all baselines. In AIFB, MUTAG, BGS and AM datasets, our model improves upon R-GCN by a margin of 2.78%, 2.94%, 10.34% and 1.01%, respectively. Without incorporating coarsened line graph, RSHN$_{\neg CL}$ ranks after RSHN. Compared with RSHN, the accuracy of classification of RSHN$_{\neg CL}$ drops by 3.61% on average. The results demonstrate the effectiveness of the

coarsened line graph by excavating latent associated features between different types of edges. The specific statistics are shown in Tabel II. Test performances of shallow baselines Feat, WL and RDF2Vec are reported on the train/test splits from [27].

TABLE II: Classification accuracy comparison (%).

Method	AIFB	MUTAG	BGS	AM
Feat	55.55	77.94	72.41	66.66
WL	80.55	80.88	86.20	87.37
RDF2Vec	88.88	67.20	87.24	88.33
GAT	91.67	72.06	66.32	67.30
R-GCN	94.44	79.41	82.76	89.39
RSHN$_{\neg CL}$	94.44	77.94	89.66	86.58
RSHN	**97.22**	**82.35**	**93.10**	**90.40**

V. Conclusions

In this paper we studied graph embedding for heterogeneous networks with many types of edge connections. We argued that existing methods mainly dissect heterogeneous networks as multiple homogeneous networks, failing to capture rich semantic interactions/relations between different types of edges. Accordingly, we proposed relation structure-aware heterogeneous graph neural network (RSHN), which first builds edge-centric coarsened line graph to excavate and exploit edge relations highly correlated to each other. By coupling different types of edges, H-GNN further takes into account hidden relation structural information and enables implicit message propagating between nodes for effective node and edge embedding learning. Experiments confirm that RSHN significantly outperforms all baselines for node classification in heterogeneous networks with rich node types and relationships.

Acknowledgment

This work was supported by the National Key Research and Development Program of China (No. 2016YFB0801003), the NSFC (No. 61872360), the Youth Innovation Promotion Association CAS (No. 2017210), and by the US National Science Foundation (NSF) through Grants IIS-1763452 and CNS-1828181. C. Zhou is the corresponding author.

References

[1] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Advances in neural information processing systems*, 2015, pp. 2224–2232.

[2] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1263–1272.

[3] T. Hamaguchi, H. Oiwa, M. Shimbo, and Y. Matsumoto, "Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach," *arXiv preprint arXiv:1706.05674*, 2017.

[4] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference*. Springer, 2018, pp. 593–607.

[5] S. Pan, R. Hu, S.-f. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *arXiv preprint arXiv:1901.01250*, 2019.

[6] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in *IJCAI*, 2019, pp. 1907–1913.

[7] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 119–128.

[8] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *arXiv preprint arXiv:1610.09769*, 2016.

[9] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," *The World Wide Web Conference*, 2019.

[10] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[11] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[12] J. L. Gross and J. Yellen, *Graph Theory and Its Applications, 2nd ed*. Chapman and Hall/CRC, 2006.

[13] Y. Zhang, Y. Xiong, X. Kong, S. Li, J. Mi, and Y. Zhu, "Deep collective classification in heterogeneous information networks," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2018, pp. 399–408.

[14] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun, "Graph neural networks: A review of methods and applications," *arXiv preprint arXiv:1812.08434*, 2018.

[15] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *arXiv preprint arXiv:1901.00596*, 2019.

[16] H. Jiang, Y. Song, C. Wang, M. Zhang, and Y. Sun, "Semi-supervised learning over heterogeneous information networks by ensemble of meta-graph guided random walks." in *IJCAI*, 2017, pp. 1944–1950.

[17] Z. Chen, L. Li, and J. Bruna, "Supervised community detection with line graph neural networks," 2018.

[18] X. Wang, C. Chen, Y. Min, J. He, B. Yang, and Y. Zhang, "Efficient metropolitan traffic prediction based on graph recurrent neural network," *arXiv preprint arXiv:1811.00740*, 2018.

[19] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *Acm Sigkdd Explorations Newsletter*, vol. 14, no. 2, pp. 20–28, 2013.

[20] K. K. Thekumparampil, C. Wang, S. Oh, and L.-J. Li, "Attention-based graph neural network for semi-supervised learning," *arXiv preprint arXiv:1803.03735*, 2018.

[21] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in neural information processing systems*, 2013, pp. 2787–2795.

[22] S. Bloehdorn and Y. Sure, "Kernel methods for mining instance data in ontologies," in *The Semantic Web*. Springer, 2007, pp. 58–71.

[23] G. K. D. De Vries, S. De Rooij *et al.*, "A fast and simple graph kernel for rdf." *DMoLD*, vol. 1082, 2013.

[24] V. De Boer, J. Wielemaker, J. Van Gent, M. Hildebrand, A. Isaac, J. Van Ossenbruggen, and G. Schreiber, "Supporting linked data production for cultural heritage institutes: the amsterdam museum case study," in *Extended Semantic Web Conference*. Springer, 2012, pp. 733–747.

[25] H. Paulheim and J. Fümkranz, "Unsupervised generation of data mining features from linked open data," in *Proceedings of the 2nd international conference on web intelligence, mining and semantics*. ACM, 2012, p. 31.

[26] G. K. D. De Vries and S. de Rooij, "Substructure counting graph kernels for machine learning from rdf data," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 35, pp. 71–84, 2015.

[27] P. Ristoski and H. Paulheim, "Rdf2vec: Rdf graph embeddings for data mining," in *International Semantic Web Conference*. Springer, 2016, pp. 498–514.

[28] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.