Expert Systems With Applications 187 (2022) 115823

Contents lists available at ScienceDirect

Expert Systems With Applications

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

HIN2Grid: A disentangled CNN-based framework for heterogeneous network learning

Ziyang Zhang [a], Chuan Chen [a,b,*], Yaomin Chang [a], Weibo Hu [a], Zibin Zheng [a,b], Yuren Zhou [a], Lei Sun [c]

[a] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China
[b] National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou 510275, China
[c] User Experience Center, Netease Games, Guangzhou 510665, China

ARTICLE INFO

Keywords:
Network embedding
Heterogeneous graph learning
Graph convolutional network
Data mining

ABSTRACT

Recently, graph convolutional networks (GCNs) have been applied to heterogeneous information network (HIN) learning and have shown promising performance. However, the performance of GCNs degrades attributed to the recursive propagation, which leads to an indistinguishable embedding for the distinctly heterogeneous node. Besides, the inherently coupled paradigm of GCNs limits their applications on large-scale graphs. In this paper, we tackle these problems by proposing a disentangled framework named Heterogeneous Information Network to Grid (HIN2Grid) for heterogeneous network learning. We innovatively design an effective and efficient strategy to transform the graph data into semantic-specific grid-like data, which can be effectively processed by convolutional neural networks (CNNs), thus explicitly overcoming the drawbacks of the inherent paradigm of GCNs. Such a CNN-based learning scheme also contributes to extracting more expressive features and consume less time and memory. We further propose dual attention mechanisms to capture the importance of various grid-like data and heterogeneous semantics, thus providing interpretability and robustness for HIN2Grid. We conduct experiments on four datasets and the results show that HIN2Grid significantly outperforms the state-of-the-art methods, gaining a improvement on node classification of about 2% to 10% and a 2 to 5 times promotion on running speed.

1. Introduction

In the real world, data often emerges as graph structures with a diversity of nodes and relations such as social networks and biological networks. Graph embedding, which aims to project nodes into a low-dimensional space while preserving structural information and properties of the network, has aroused attention in various fields (Chen, Li, Qian, Zheng, & Hu, 2020; Fei, Ren, & Ji, 2020; Huang, Chen, Ye, Hu, & Zheng, 2020; Tang, Chen, Cui, & Wei, 2019; Vo, Al-Obeidat, & Bagheri, 2020). After obtaining the embedding, they can be readily applied to various downstream tasks. Generally, graph-based applications can be categorized into node classification tasks or link prediction tasks, such as properties predictions (Ata, Fang, Wu, Li, & Xiao, 2017; Huang & Zitnik, 2020; Lin, Quan, Wang, Ma, & Zeng, 2020) and social recommendations (Chen, Jiang, et al., 2020; Chen, Wu, Hong, Zhang, & Wang, 2020; Ma et al., 2020).

So far, most graph embedding models have been designed for homogeneous graphs which have a single type of nodes and edges. However, the real-world data is rich in various types and relations, which is often presented as heterogeneous graphs. As shown in Fig. 1, the graph contains three types of nodes, i.e. author, paper and conference. Besides, the relations among these nodes are different. For example, the relation from author to paper is 'write' and the relationship between conference and paper is 'accept' or 'accepted'. In order to capture the rich semantic information, more and more heterogeneous graph embedding methods have been proposed. Roughly speaking, there are some meta-path based random walk methods (Dong, Chawla, & Swami, 2017; Fu, Lee, & Lei, 2017) inspired by DeepWalk (Perozzi, Al-Rfou, & Skiena, 2014), some methods (Tang, Qu, & Mei, 2015a; Xu, Wei, Cao, & Yu, 2017) base on the first-order and second-order similarity inspired by LINE (Tang et al., 2015), some methods (He et al., 2020; Hu, Li, Shi, Yang, & Shao, 2020) base on the message-passing framework (Gilmer, Schoenholz, Riley, Vinyals, & Dahl, 2017) that utilize graph convolutional networks (GCNs) to preserve local structural information. Among these methods, GCN-based methods achieve state-of-the-art performance in most cases

* Corresponding author.
E-mail addresses: zhangzy233@mail2.sysu.edu.cn (Z. Zhang), chenchuan@mail.sysu.edu.cn (C. Chen), changym3@mail2.sysu.edu.cn (Y. Chang), huwb7@mail2.sysu.edu.cn (W. Hu), zhzibin@mail.sysu.edu.cn (Z. Zheng), zhouyuren@mail.sysu.edu.cn (Y. Zhou), gzsunlei@corp.netease.com (L. Sun).

https://doi.org/10.1016/j.eswa.2021.115823
Received 19 February 2021; Received in revised form 20 May 2021; Accepted 27 August 2021
Available online 11 September 2021

, thereby drawing significant attention in both academic and industrial domains.

However, GCNs possess inherently coupled framework that repetitively propagates features along the edges among nodes during convolution, which limits its flexibility. Graph convolution can generally fall into spectral convolution and spatial convolution. Despite spatial-based models avoid certain complex computation such as the decomposition of Laplacian matrix existed in the spectral-based methods, they need to store all the intermediate embedding and suffer significant computational overhead due to the neighborhood expansion problem (Chiang et al., 2019), which leads to a great challenge for developing scalable GCN algorithms. In addition, when the graph convolutional layers are stacked too more, GCNs have a high risk of over-smoothing which leads to undistinguishable node embeddings. To address the above problems, various strategies have been introduced, including neighbor sampling (Hamilton, Ying, & Leskovec, 2017), layer sampling (Chen, Ma, & Xiao, 2018), and sub-graph training (Gao, Wang, & Ji, 2018). Although introducing these strategies can overcome aforementioned problems to some extent, they are still fettered to the inherent framework of GCN, i.e. repeatedly performing the complex aggregations among neighbors of nodes during graph convolution.

As to convolution operators, convolutional neural networks (CNNs) (LeCun, Bottou, Bengio, & Haffner, 1998) have achieved great performance on grid-like data (e.g. images) and are widely used in computer vision tasks and natural language processing tasks. With the trainable local filters, high-level information can be extracted automatically. The successful design of CNNs sheds light on improving directions on graph embedding. Intuitively, if CNNs can be applied to graphs, the aforementioned problems can be fully solved. There are two reasons: (1) the local receptive field and sharing weights of CNNs greatly reduce the number of parameters, and CNNs avoid the inflexibility of suffering recursive neighborhood expansion. Since CNNs have both faster speed and lower memory cost than GCNs, it is effortless to deal with large-scale graphs and dense graphs. (2) The CNN-based framework also avoids the over-smoothing problems of GCNs. Nevertheless, applying CNNs on graphs faces two severe challenges: (1) the nodes in graphs have no fixed number of neighbors and (2) there is no ranking information to put neighboring nodes in a certain order. Therefore, how to effectively transform the heterogeneous graphs to structured grids is the key problem.

In this paper, we propose a disentangled CNN-based framework for heterogeneous graph learning named Heterogeneous Information Network to Grid (HIN2Grid). Considering the heterogeneity of HINs, meta-paths are utilized to reconstruct adjacent relations and the original graph is divided into several semantic-specific subgraphs. To consider the scalability of HINs, an efficient strategy is proposed to transform graph data into grid-like data. Concretely, HIN2Grid selects the most influential first-order and second-order local neighbors for each node and combine them to structured grid by their degrees, without being restricted to complex pre-processing and comparison in some previous efforts. Then, CNNs are employed to operate on the generated grids so as to automatically extract effective information and speed up convolution while saving memory. Additionally, instead of separating the target node and its corresponding neighboring nodes during grid generation, a central fused mechanism is proposed to fuse their features into the grid. In this way, each row of the grid-like data contains the features from central nodes and their local structures which improves the expressiveness of the model. Lastly, HIN2Grid also designs dual attention mechanisms to obtain optimal combinations of various grid pixels and multifaceted semantic embeddings in a hierarchical manner, which enables the model to be robust and comprehensive.

In summary, we highlight our contributions as follows:

- We propose an effective and efficient strategy to transform graph data into grid-like data, and with the central fusion mechanism, the structured grids contain rich features from both central nodes and corresponding local structures.
- We propose a novel framework HIN2Grid, which employs CNNs to extract useful features automatically from the generated grid-like data. Owing to CNNs, HIN2Grid has advantages in computational efficiency and memory saving since it avoids the inflexibility of suffering recursive neighborhood expansion.
- We design grid-level and semantic-level attention mechanisms to capture the importance of different grid pixels and multifaceted semantic information, which improves the robustness and provides interpretability to the model.
- We evaluate HIN2Grid on four datasets, and extensive experiments demonstrate the superiority of HIN2Grid as compared to state-of-the-art models on accuracy and complexity in various tasks.

2. Literature review

2.1. HIN embedding

Network embedding refers to project nodes into a low dimensional space then they can be applied in various downstream tasks. So far, most network embedding methods have been designed for homogeneous graphs. Nevertheless, the data in the real world is highly complex with the existence of multiple types of objects and relations. Therefore, more and more HIN embedding methods are proposed in recent years. The main challenges in HIN embedding are rich semantics and variable graph scales. Instead of utilizing traditional embedding algorithms such as SkipGram (Mikolov, Chen, Corrado, & Dean, 2013), recent works designed various methods that enable models to contain both semantic and structural information. HAN (Wang, Ji, et al., 2019) adopts node- and semantic-level attentions to balance the importance among different nodes and meta-paths. RSHN (Zhu, Zhou, Pan, Zhu, & Wang, 2019) considers different relations based on coarsened line graphs and employs graph neural networks to model interactions between nodes and their neighbors. HGT (Hu, Dong, Wang, & Sun, 2020) utilizes node-level and edge-level parameters to obtain heterogeneous attention over each edge. Although these GCN-based models take heterogeneity into account, they have to make a balance in computational efficiency and accuracy since GCN is an internal coupling framework that limits their flexibility.

2.2. Graph convolutional network on large graphs

Graph convolutional networks (GCNs) aim to extend neural networks to deal with graph-structured data, which have shown great popularity in tackling graph analytics problems. However, GCN as a full-batch framework, matrix operations are performed over the whole graph, which leads to expensive computation and memory cost when applied to dense and large graphs. In order to solve the aforementioned problems, the mini-batch strategy is proposed in GraphSage (Hamilton et al., 2017). Since embedding updating is based on a small batch, the model can speed up convergence and reduce memory requirements. Nevertheless, it is still limited to high computational costs due to the neighborhood expansion problem. FastGCN (Chen et al., 2018) utilizes global importance sampling to tackle the problem of complex calculation, but it needs extra requirement for importance weight computation and becomes worse when GCN goes deep. VR-GCN (Ye, Li, Fang, Zang, & Wang, 2019) reduces the size of sampled neighbors of each node by variance reduction. Although the sample size is reduced, it needs extra space to store all the intermediate embeddings of all the nodes and thus resulting in high memory requirements. Cluster-GCN (Chiang et al., 2019) partition the original graph into several subgraphs by graph clustering techniques. However, the effectiveness of the model heavily depends on the selected graph clustering algorithm. LGCN (Gao et al., 2018) proposes a framework that utilizes a subgraph selection strategy to improve GCNs. In particular, LGCN finds neighbors for each target node and ranks them by their feature values to generate

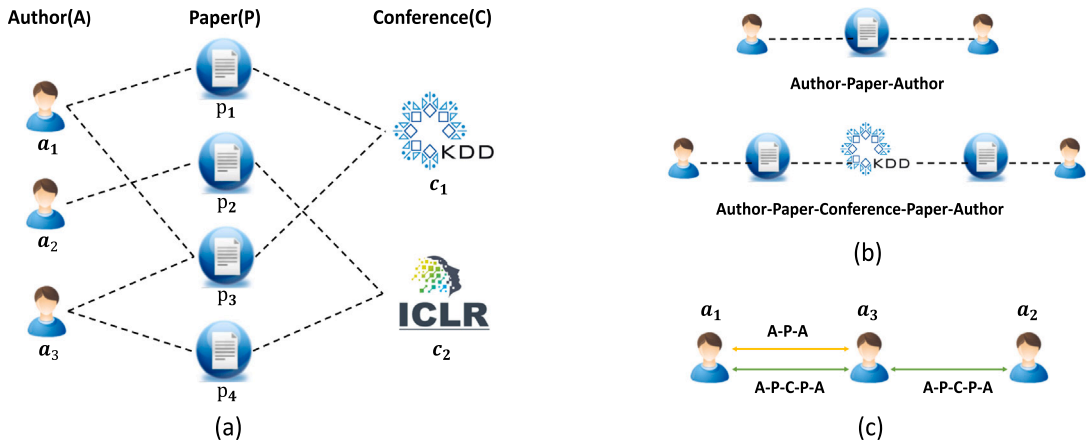

Fig. 1. (a) An example of heterogeneous graph which include three types of nodes: author, paper and conference. (b) Meta-paths. (c) Meta-path based neighbors.

grid-like data. Although LGCN achieves improvements, it suffers from heavy data-preprocessing, e.g. features ranking and selection, and the subgraph may introduce noise from long-distance positions. Although there are many related and similar methods, most of them are designed for homogeneous graphs. Hence, it is necessary to study the scalability of heterogeneous graph embedding algorithms.

2.3. Analysis on similar graph embedding models

In this section, we compare our work with existing HIN embedding models (Hu, Dong, Wang, & Sun, 2020; Schlichtkrull et al., 2018; Wang, Ji, et al., 2019; Wang, Zhang, et al., 2019; Zhu et al., 2019) base on graph convolutional networks and representative efficient methods (Chen et al., 2018; Chiang et al., 2019; Gao et al., 2018; Hamilton et al., 2017; Ye et al., 2019) for processing large graphs. As mentioned before, GCN-based HIN embedding models have a variety of ways to deal with the heterogeneity of graphs. However, they are suffered from the repeated computations and unnecessary data transfers in GCN training and inference (Jia et al., 2020). Although many sampling strategies have been proposed to speed up the process of convolution, they are still restricted by the neighborhood expansion problem or lead to more space requirements that make models inflexible.

Proceeding from this point, the proposed disentangled framework adopts an efficient strategy to transform graph data into grid-like data while preserving informative features of local structures. There are two advantages to this design. Firstly, the grid-like data can be effectively processed by CNNs, thus the proposed model has an advantage in efficiency and memory saving over GCN-based models. Secondly, it explicitly avoids the inflexibility of the graph convolution paradigm, so it is not restricted to complex neighborhood expansion and over smoothing problems. Besides, we also propose dual attention mechanisms to enhance the robustness and expressiveness of the proposed model. Experiments show that our work is revolutionary compared with the state-of-arts in efficiency and accuracy.

3. Preliminary

In this section, we formalize definitions and introduce backgrounds of HIN embedding.

Definition 1 (*Heterogeneous Information Network*)**.** A heterogeneous information network is composed of a node set V and a link set E which can be denoted as $g = (V, E)$. It along with the node type mapping function $\rho : V \to M$ and edge type mapping function $\varphi : E \to R$, where M and R denote the sets of the node and edge types, and $|M| + |R| > 2$.

Fig. 1(a) is a toy example of HIN on a citation network. The graph consists of multiple node types (i.e., author, paper and conference) and edge types (i.e., write and accept). In heterogeneous information network, two nodes are connected via different semantic paths, which are defined as meta-paths.

Definition 2 (*Meta-Path*)**.** A meta-path is defined as a sequence path between different types of nodes in the form which describes a composite relation $\mathrm{R} = R_1 \circ R_2 \circ \cdots \circ R_l$ between objects O_1 and O_l.

As shown in Fig. 1(b), two authors are connected with multiple meta-paths, i.e. Author–Paper–Author (APA) and Author–Paper–Conference–Paper–Author (APCPA). The two meta-paths represent different semantics. For example, the APA indicates the two authors share the same research direction, while APCPA means two authors may be interested in certain scientific areas.

Definition 3 (*Meta-Path based Neighbors*)**.** A node's meta-path based neighbors are a set of nodes connected with the node through meta-paths.

As we can see in Fig. 1(c), a_1 connects with a_3 bases on the meta-path Author–Paper–Author and a_2 is a meta-path based neighbor of a_3 on the meta-path APCPA. As discussed above, different meta-paths reflect different semantics. Therefore, the impact of different meta-paths on graph learning is ought to be considered.

Problem setting: (Heterogeneous network embedding)

Heterogeneous network embedding aims to learn an embedding function $\tau : V \to R^d$ that maps the nodes $v \in V$ in the network into a low-dimensional space with $d \ll |V|$, while preserving local structural properties of nodes for various downstream tasks. In this work, we focus on the task of semi-supervised node classification.

4. The proposed model

In this section, we introduce the proposed framework for heterogeneous graph embedding. The overview of HIN2Grid is presented in Figs. 2 and 3. Firstly, taking the heterogeneity into account, we utilize meta-paths to transform the original heterogeneous graph into several semantic-specific homogeneous graphs. After that, influential neighboring nodes of each node are selected within its 2-hop neighbors according to their degree and combined as a structured grid. Then, we design a central fusion mechanism to enhance the expressive power of the grid by fusing features of the node and its influential neighbors. Next, the grid-like data is fed into CNNs thus the high-level information can be extracted automatically. Finally, grid-level attention and semantic-level attention are utilized to learn the weights of various grid pixels and semantic embeddings, thus provide the optimal combination of them to obtain the final embedding.

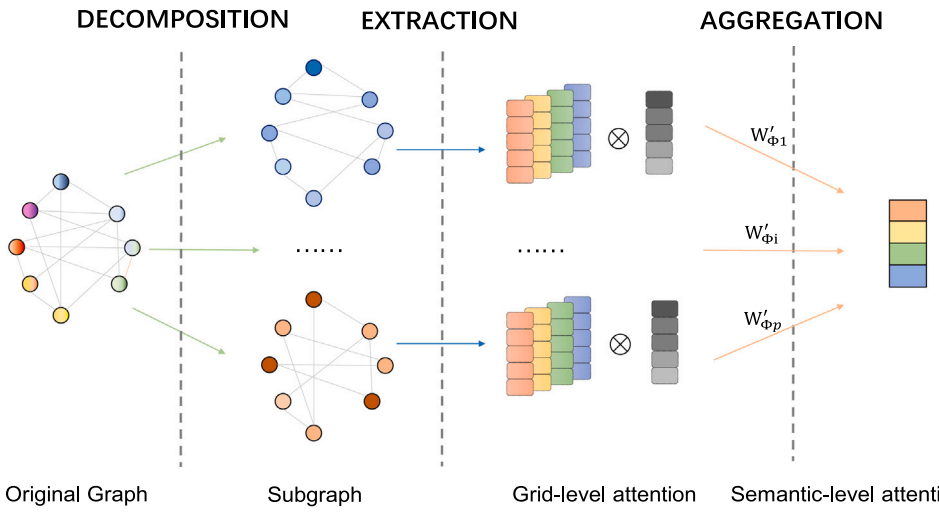

Fig. 2. An illustration of the framework for the proposed HIN2Grid. Firstly, the original graph is divided into semantic-specific homogeneous subgraphs by different meta-paths, and then the graph data is transformed into grid-like structures and fed into CNNs. Next, the extracted features are combined by attention mechanisms to obtain the final representation.

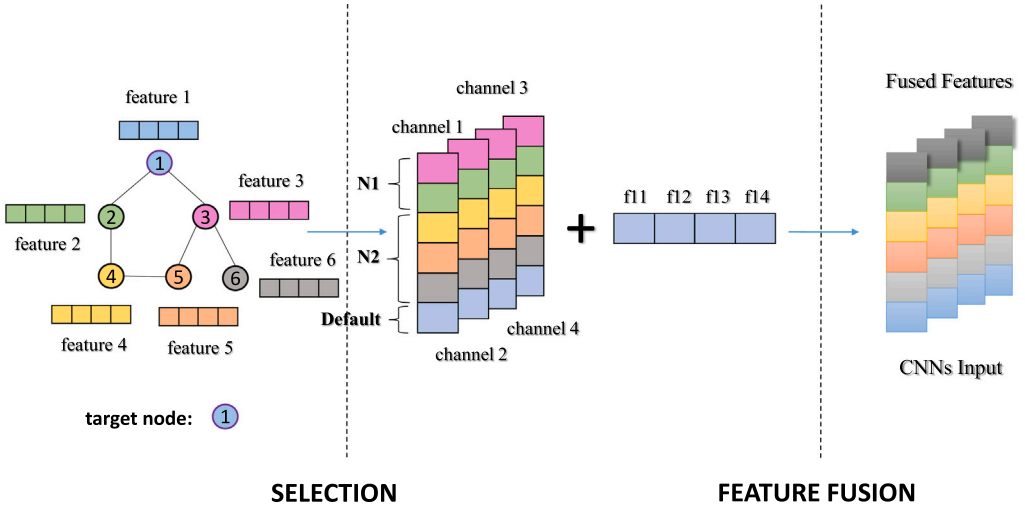

Fig. 3. Detailed process in the EXTRACTION in Fig. 2. Firstly, we select the most influential first-order and second-order neighbors for the target node (blue node, id 1) and rank them by degree, then assemble them to generate structured grids. Then, we fuse the features of grid-like data as well as the target node, and use them as the input of CNNs. In this plot, we set k=6 and the features of nodes are 4.

4.1. Generation of grid-like data

Due to the diversity of nodes and edges in HINs, heterogeneous graph structures have abundant and complex semantic information. In order to analyze the heterogeneity, we utilize the widely used semantic connection patterns, meta-path (Sun, Han, Yan, Yu, & Wu, 2011), to transform the original graph into several homogeneous subgraphs, thus various semantics can be fully explored.

Owing to the intrinsic non-Euclidean property of graphs, CNNs cannot be employed to these subgraphs directly since there is no fixed structure and node order existed in graphs. Therefore, a reasonable and efficient way should be designed to transform graphs into grid-like structures. Next, we discuss the two difficulties involved.

1. Which nodes are important to the target node?

According to the homophily hypothesis (Xu, et al., 2020, 2020; Yang & Leskovec, 2014), the closer the nodes are, the more likely they have similar properties. Therefore, the nodes close to the target ones have the priority to be selected. However, data in the real world is usually sparse, if only one-hop neighbors are selected for the node, the expressiveness of the model would be insufficient. On the basis of the analysis and experiments in Chen, Wei, Wang, and Guo (2019), Kipf and Welling (2016), the first-order and second-order neighboring nodes contain rich local structural information. Consequently, we select both the one-hop and two-hop nodes as the neighbors of target nodes. In this way, the receptive field of each node is expanded to improve the expressive power of the following constructed grids, and higher-order nodes are not considered to avoid the disturbance caused by noise.

2. How to transform the selected nodes into structured grid?

Before transforming the selected nodes into grid-like data, we should notice that there is no ranking information in the graphs. Some previous studies adopt feature ranking (Gao et al., 2018) or attention mechanism (Liu, Wang, & Ji, 2020) to fix orders of nodes. However, they are restricted to the heavy calculation that limits their flexibility. The study of influential research (Tangmunarunkit, Govindan, Jamin, Shenker, & Willinger, 2002; Zhao, Liu, Wang, Li, et al., 2017) indicates improving directions, which demonstrates that the degree of nodes reflects the influence of the node and topological properties of graphs. As a common character of graphs, the higher degree of the node, the

greater its influence. What is more, it is suitable and convenient for calculating the degree of nodes that provides the potential advantages of processing large and dense graphs. Consequently, the selected nodes are ranked by their degrees and assembled to construct structured grids.

Suppose the target node has f features, and k neighbors are selected to construct the structured grid, the dimension of the grid-like data is $k \times 1 \times f$ where f can be seen as the channels of the grid, and $k \times 1$ is the size of grid-like data in each channel. To better understand the construction process of grid-like data, we give a brief explanation in Fig. 3. Taking node 1 as an example, its influential neighboring nodes (i.e., first-order neighbors N_1 and second-order neighbors N_2) are selected and permutated according to their degree. Note that some graphs may be very sparse, if $|N_1| + |N_2| < k$, the default value of unfulfilled grid is set to features of target nodes to highlight themselves. Additionally, since the features of the node itself can best reflect its properties in the graph, we inject the features of the target node in the grids, thus the target node is coupled with the corresponding neighbors and the constructed grid contains rich information of the local structure. Concretely,

$$G = F_n \cdot \theta_{bias} + F_c \cdot (1 - \theta_{bias}), \tag{1}$$

where $F_n \in R^{(k\times 1\times f)}$ denotes the grid-like data combined by the features of selected neighbors and $F_c \in R^{(k\times 1\times f)}$ denotes the features expanded from the central node. θ_{bias} is the coefficient to balance the impact of two sides.

4.2. Feature transformation

After obtaining the structured grid, HIN2Grid employs CNNs to extract significant features. Specifically, given a constructed grid, CNNs with kernel size s is utilized to extract features, which can be written as follows:

$$x_u = Conv(p(G)), \tag{2}$$

where G is the grid-like data, p denotes transformation function, e.g., MLP. $Conv(\cdot)$ denotes a 1-D CNN to extract features, in which the stride is set to 1 and without padding strategy. x_u is the obtained embedding that contains high-level information in the grid.

4.3. Grid-level attention mechanism

In GAT (Veličković et al., 2018), different neighbors show different importance in node embedding. Similarity, we assume that different pixels in the grid contribute differently to node learning in HIN2Grid. To explore the importance of each pixel, we adopt grid-level attention mechanism to improves the expressive power and robustness of the model. Due to the high variance of data in graphs, we extend our mechanism to multi-head attention to stabilize the learning process. The improved embedding is shown as follows:

$$x'_u = x_u + (\frac{1}{h}\sum_{i=1}^{h} Att_i) \cdot x_u, \tag{3}$$

where $Att_i \in R^{((k-s+1)\times 1\times f)}$ denotes the attention matrix, h is the number of attention heads. Additionally, we employ two-layer nonlinear transformations to map the embedding to a low dimensional space:

$$x'_\phi = g(x'_u), \tag{4}$$

where x'_ϕ is the node embedding under the meta-path ϕ. Given the meta-path set $\{\phi_1, \phi_2, \ldots, \phi_p\}$, we can get p groups of semantic-specific embeddings, denoted as $\{x'_{\phi_1}, x'_{\phi_2}, \ldots, x'_{\phi_p}\}$.

4.4. Semantic-level attention mechanism

As mentioned before, different meta-paths reveal different semantic information. After obtaining p groups of semantic embeddings, HIN2Grid utilizes semantic-level attention to fuse various semantics to obtain comprehensive embeddings. To learn the attention weights of meta-paths, the node embedding is transformed through a nonlinear transformation, is shown as follows:

$$x^*_{\phi_i} = \sigma(W \cdot x'_{\phi_i} + b), \tag{5}$$

where W is a learnable weight matrix, b is a bias vector and $\sigma(\cdot)$ denotes the activation function. Furthermore, we calculate the similarity between embeddings and the attention vector a, and then average them to obtain the attention weight of each meta-path:

$$w_{\phi_i} = \frac{1}{|V|}\sum_{j\in V} a^T \cdot x^{*,j}_{\phi_i}, \tag{6}$$

where a is a learnable vector and $x^{*,j}_{\phi_i}$ is the semantic specified embedding of node j under the meta-path ϕ_i. The weight of meta-path can be obtained by normalizing the attention weights of all meta-paths:

$$w^*_{\phi_i} = \frac{exp(w_{\phi_i})}{\sum_{i=1}^{p} exp(w_{\phi_i})}. \tag{7}$$

The weight $w^*_{\phi_i}$ reflects the importance of meta-path ϕ_i, i.e., a more important meta-path should have a higher weight. Hence the attention mechanism provides the interpretability to the model. With the learned weights, semantic embeddings are fused together to obtain the final representation of the node:

$$Z = \sum_{i=1}^{p} w^*_{\phi_i} \cdot x^*_{\phi_i}, \tag{8}$$

where $Z \in R^{1\times C}$, C denotes the number of classes of nodes in the dataset. Suppose N_L denotes the training set, for each $l \in N_L$ the real label and predicted label is Y^l and Z^l respectively. For model optimization, we minimize the cross-entropy loss with l_2 regularization defined as follows:

$$L = -\sum_{l\in N_L}\sum_{i=1}^{C} Y^l_i \cdot log(Z^l_i) + \lambda \sum w^2_{att}, \tag{9}$$

where w_{att} denotes learnable parameters in the attention matrix in Eq. (3), λ is the constraint coefficient of regularization term. The detail of the calculation process is illustrated in Algorithm 1.

Algorithm 1 The algorithm of HIN2Grid

Input: An HIN graph $g = (V, E)$,
meta path set $\{\phi_1, \phi_2, \ldots, \phi_p\}$,
node features $\{F_i, i \in V\}$,
grid dimension K, fusion coefficient θ_{bias},
regularization coefficient λ;
Output: Final embedding Z,
semantic-level attention weights w_{as};
1: Select neighbors for each node and rank them by degree;
2: Generate grid-like data G via Eq. (1);
3: **for** $\phi_i \in \{\phi_1, \phi_2, ..., \phi_p\}$ **do**
4: Learn the updated and improved embedding by Eq. (2) and Eq. (3);
5: Learn semantic-specific node embedding via Eq. (4);
6: **end for**
7: Calculate the weights of meta-paths w_{as} via Eq. (7);
8: Calculate the final embedding Z via Eq. (8);
9: Calculate Cross-Entropy L via Eq. (9);
10: Update parameters in HIN2Grid;
11: **Return** Z, w_{as};

5. Complexity analysis

In this section, we analyze and compare the time and memory complexity of the proposed disentangled CNN-based framework and GCN .

5.1. Complexity of graph convolution networks

Suppose we are given a graph $g = (V, E, A, X)$, where $V = \{v_1, \ldots, v_N\}$ denotes the vertex set with N nodes and E denotes the edge set. $A \in R^{N\times N}$ is the adjacency matrix and $X \in R^{N\times F}$ is the feature matrix. In each layer of GCN, the embedding of each node are updated by aggregating the embeddings of neighbors of the node:

$$Z^{(l)} = \sigma(\widetilde{A}X^{(l-1)}W^{(l)}), \tag{10}$$

where $\widetilde{A}$ is the regularized adjacency matrix, $W^{(l)} \in R^{F_l\times F_{l+1}}$ and $X^{(l-1)} \in R^{N\times F_l}$ are the learnable transformation matrix at the lth layer and node embeddings at the $(l-1)$th layer, respectively, and $X^{(0)} = X$. For simplicity and without affecting the analysis, we set $F_1 = \cdots = F_l = F$ and ignore the heterogeneity of graphs.

For the time complexity of the information aggregation in (10), $X^{(l)}_* = \widetilde{A}X^{(l)}$ cost $O(|E|F)$ and $X^{(l)}_* W^{(l)}$ costs $O(NF^2)$ in time. Hence for L-layer GCN, the total time complexity is $O(L|E|F + LNF^2)$. For the memory complexity, L-layer GCN need to store node embeddings and transformation matrices in L layers, which in total leads to $O(LNF + LF^2)$ in memory.

5.2. Complexity of the proposed framework

The proposed framework utilizes CNNs to extract significant information from grid-like data. Different from GCN suffers from the inflexibility of full-batch training, the proposed HIN2Grid can utilize mini-batch training algorithms to improve the training speed and memory requirement. Suppose we are given a convolutional kernel $k_s \in R^{S\times 1}$, minibatch size B and structured grid with the size of $K \times 1 \times F$ for each node, where K denotes the number of neighbors we selected for each node. For a fair comparison, here we also do not consider the heterogeneity. The time complexity of the convolution is $O(B(K-S)SF^2)$, and the time complexity of grid-level attention is $O(BH(K-S)F)$, where H denotes the number of attention heads. With regard to the memory complexity, storing parameters and node embeddings requires $O(BF+SF^2+H(K-S)F)$ in memory. We summary the complexities of the two models in Table 1.

5.3. Complexity comparison

From the analysis above, we can observe that the complexity of GCN highly depends on the number of nodes and edges in the graph. Before comparison, it is worth noting that the hyper-parameters in our method are far less than the number of features and nodes in the graph, i.e., $S \approx H \approx K \ll F \ll N$. Therefore, our method has the advantage in computing efficiency and memory saving over GCN on dealing with large and dense graphs, especially when GCN has many layers.

In this work, we regard the construction of structured grids as part of data preprocessing. Consider the heterogeneity of graphs, the semantic-level attention is utilized and its complexity is $O(BF^2)$. Besides, HIN2Grid can be easily parallelized since the convolution and attention mechanisms can be parallelized across nodes and meta-paths, respectively.

6. Experiments

In this section, we evaluate the HIN2Grid in four datasets. We further analyze the model in accuracy, efficiency and parameter sensitivity.

6.1. Datasets

We conduct experiments on four datasets, including ACM (with three types of nodes: author, paper and subject), DBLP (paper, author, conference, term), IMDB (movie, actor, director) and PUBMED (gene, disease, chemical, species). We organize them into heterogeneous graphs and summarize them in Table 2.

1. The DBLP dataset we constructed consists of 4057 authors, 8789 terms, 14328 papers, 20 conferences. The authors in this datasets belong to four classes, including information retrieval, data mining, machine learning, and database. The authors' research areas are labeled by the conferences they submitted and their features correspond to elements of a bag-of-words represented of keywords.
2. The IMDB dataset we constructed consists of 4780 movies, 2269 directors and 5841 actors. Features of Movie are obtained from a bag-of-words of plots. The movies belongs to three categories: drama, comedy and action.
3. The ACM dataset we constructed consists of 3025 papers, 56 subjects and 5835 authors. Features of Paper are calculated from a bag-of-words of keywords. The papers are divided into three areas, including data mining, database and wireless communication.
4. The PUBMED dataset we constructed consists of genes, diseases, chemicals, and species from PubMed.[1] The features of nodes are the elements of 200-dim embedding computed by word2vec. The diseases are labeled for eight categories and each labeled disease has one label.

6.2. Baselines

To verify the effectiveness of the proposed model, we compare HIN2Grid with the following state-of-the-art models. The baselines are divided into three categories: homogeneous graph embedding models LINE (Tang et al., 2015) and DeepWalk (Perozzi et al., 2014); random-walk based HIN embedding model HIN2Vec (Fu et al., 2017); GCN-based HIN embedding model HAN (Wang, Ji, et al., 2019), R-GCN (Schlichtkrull et al., 2018) and HGT (Hu, Dong, Wang, & Sun, 2020).

1. LINE utilizes the first- and second-order proximity to preserve structural properties of the graph.
2. DeepWalk is a random-walk based model that employs Skip-Gram algorithm to generate node embedding.
3. HIN2Vec learns the latent embedding of nodes by conducting multiple prediction training tasks jointly.
4. HAN adopts both node- and semantic-level attention to guide embedding generation.
5. R-GCN utilizes relational graph convolutional networks to model the relations among the nodes.
6. HGT designs node-type and edge-type parameters to obtain the heterogeneous attention on each edge.

6.3. Parameter settings

For the proposed HIN2Grid, we use Adam (Kingma & Ba, 2014) to optimize the model and randomly initialize parameters. Besides, we set the learning rate to 0.01, weight decay to 0.0005, the dropout rate to 0.4, the number of grid-level attention heads to 10, coefficient of regularization term λ to 0.1, kernel size s to 2, the dimension of the hidden embeddings of two nonlinear transformations in Eq. (4) to 128 and 16, respectively. In the proposed model, $tanh(\cdot)$ is utilized as the

[1] https://www.ncbi.nlm.nih.gov/pubmed/.

Table 1
Time and memory complexity.

	GCN		Proposed framework	
Time complexity	Feature aggregation $O(L\|E\|F)$	Transformation $O(LNF^2)$	Convolution $O(B(K-S)SF^2)$	Grid-level attention $O(BH(K-S)F)$
Memory complexity	Embedding storage $O(LNF)$	Weight storage $O(LF^2)$	Embedding storage $O(BF)$	Weight storage $O(SF^2+H(K-S)F)$

Table 2
Statistics of four datasets.

Dataset	Relations	Nodes	Avg. Degree	Features	Edges	Labels	Meta-Paths
DBLP	Paper-Author Paper-Conf Paper-Term	4780	2904.6	334	8799281	4	APA APCPA APTPA
IMDB	Movie-Actor Movie-Director Movie-Year	4057	195.2	1232	917178	3	MAM MDM MYM
ACM	Paper-Author Paper-Subject	3025	740.2	600	2221657	3	PAP PSP
PUBMED	Disease-Gene Disease-Disease Disease-Chemical Disease-Species	20163	305.9	200	119819	8	DGD DD DCD DSD

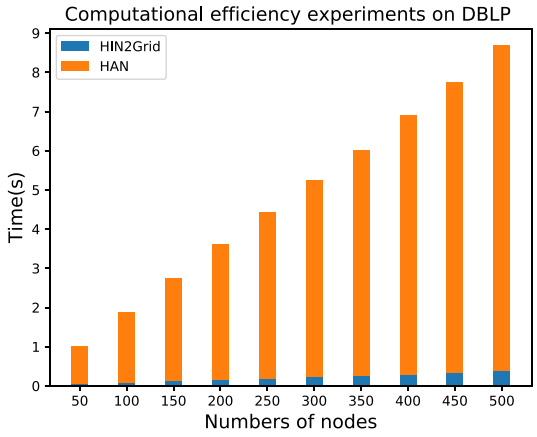

Fig. 4. Computational efficiency experiments on DBLP. For clearly displaying, the number of nodes in the training phase is set from 50 to 500.

activation function. The selection of the remaining hyperparameters K and θ_{bias} depend on the statistical characteristics of datasets, we discuss them in Section 6.6. For random walk based models, negative samples per node are set to 5, walk length is set to 50, walks per node are set to 20, the window size is set to 5. For other compared methods, we report the results by re-running the released code with suggested hyper-parameters. The nodes are randomly sampled and split into training set and test set by a ratio of 0.8:0.2, then evaluated by Micro F1 and Macro F1 on the test set. For each model, we report the average performance on 10 repeated processes.

6.4. Classification

To evaluate the embeddings learned from the above algorithms, we compare HIN2Grid to state-of-the-art models on node classification and report the Macro-F1 and Micro-F1 in Table 3.

As we can observe from the table, as graph embedding models for homogeneous graphs, LINE and DeepWalk ignore the heterogeneity of the network thus perform worse than HIN embedding models. Broadly speaking, GCN-based models such as HAN and R-GCN usually achieve better performance because the models aggregate both the features and structural information at each graph convolutional layer. By contrast to the above models, HIN2Grid significantly outperforms all the baselines on four datasets and achieves maximum relative improvements of 9.2% to the state-of-art HAN. The improvements derive from both constructed grid-like data contains rich features of central nodes as well as their influential neighbors, and high-quality node embedding with effective information extracted by CNNs.

To further compare the HIN2Grid and HAN, and verify the statistical robustness of our experimental setup, we calculate confidence intervals via bootstrapping and report the p-values of a paired t-test between the two models. From the last line in Table 3, it is obvious that the improvements of HIN2Grid over HAN are statistically significant with paired t-test at $p < 0.05$ on all datasets. Overall, the experimental results demonstrate the effectiveness of HIN2Grid.

6.5. Computational efficiency

In this subsection, we evaluate the proposed model on computational efficiency. According to the previous discussion, GCN-based models are inflexible due to the repeated and redundant neighborhood expansion. To verify this, we compare the proposed model with the HIN- and GCN-based model HAN, which also uses meta-paths and attention mechanism to explore the heterogeneity of graphs. In order to make a fair comparison, we set the same number of training epochs, i.e. 100 epochs, and the same embedding dimension of hidden layers for the two models, and record the corresponding average time per epoch. As to the memory usage, it includes the memory needed for training the model. Experimental results are reported in Table 4.

From the table, we observe that the proposed model has better computational efficiency and much more memory saving than HAN, indicating the effectiveness of the proposed framework that can speed up the training progress and saving memory to enhance the scalability of the model. Concretely, we gain improvement of mostly 5 times faster and 1/3 memory saving than HAN. For further study, we conduct extra experiments on DBLP. For clear illustration, the number of nodes in the training phase is set from 50 to 500. We take the size of the whole training set as the batch size of HAN and the proposed model for fair comparison and keep K fixed. Experimental results are shown in Fig. 4. From the plot, we can observe that the proposed model is faster than HAN, and as the number of nodes increases, the time gap between the two models becomes larger and larger, which verifies the advantage of HIN2Grid in efficiency.

Table 3
Experimental results on node classification.

Dataset	ACM		IMDB		DBLP		PUBMED	
Metrics	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
DeepWalk	0.853	0.849	0.520	0.520	0.868	0.870	0.109	0.139
LINE-1st	0.943	0.933	0.439	0.440	0.919	0.920	0.159	0.174
LINE-2st	0.960	0.960	0.637	0.627	0.872	0.870	0.117	0.233
HIN2Vec	0.836	0.840	0.426	0.427	0.870	0.880	0.359	0.384
R-GCN	0.796	0.800	0.585	0.587	0.877	0.880	0.482	0.512
HAN	0.947	0.947	0.585	0.587	0.930	0.930	0.508	0.535
HGT	0.946	0.947	0.617	0.613	0.919	0.920	0.330	0.370
HIN2Grid	0.984±.008	0.983±.008	0.639±.022	0.64±.021	0.949±.004	0.950±.005	0.529±.018	0.558±.02
+Impv.	+3.9%	+3.8%	+9.2%	+9.0%	+2.0%	+2.1%	+4.1%	+4.3%
p-value	1.29E−56	1.48E−58	1.23E−34	2.81E−32	1.96E−43	1.20E−44	2.58E−37	8.25E−33

Table 4
Experimental results on computational efficiency.

Dataset	ACM		DBLP		IMDB		PUBMED	
Hin2Grid	Time **0.19 s**	Memory **1.52 M**	Time **0.44 s**	Memory **3.32 M**	Time **0.23 s**	Memory **1.24 M**	Time **0.27 s**	Memory **3.13 M**
HAN	Time 0.52 s	Memory 3.16 M	Time 2.07 s	Memory 8.17 M	Time 0.32 s	Memory 2.32 M	Time 0.51 s	Memory 9.22 M

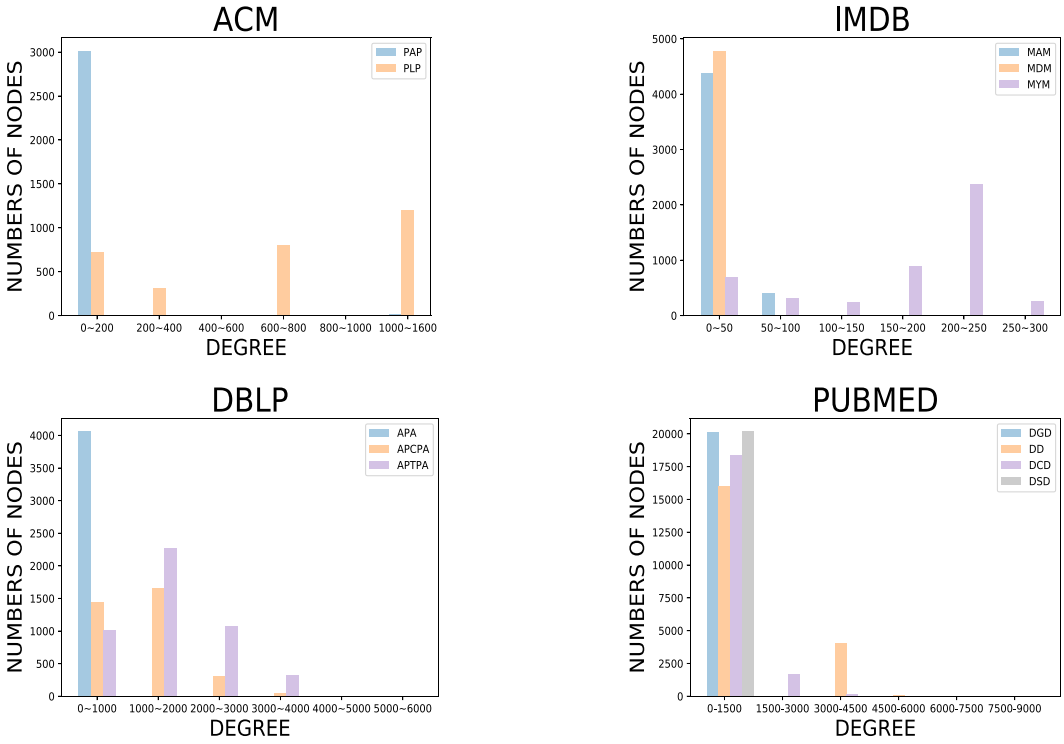

Fig. 5. Degree distribution of all nodes of target node type on four datasets. Here the neighboring nodes are based on all meta-paths.

6.6. Model analysis

6.6.1. Impact of K

The hyperparameter K controls the size of the grid. As mentioned before, influential first-and second-order neighbors of target nodes are selected to construct structured grids. From Fig. 6 we can see that the performance of the model increases first and then decreases. It may probably because that small grids constructed with few neighboring nodes contain less informative features, and also, larger K may introduce pointless features to deteriorate the performance of the model.

Degree information is shown in Table 2 and Fig. 5, we can observe that graphs from ACM, IMDB and PUBMED datasets are relatively sparse, i.e. their average degrees are 195.2, 740.2, 305.9, respectively. In order to increase the robustness and effectiveness of the model, a small K should be chosen. Whereas the DBLP dataset contains rich and dense links, has a high average degree than other datasets, i.e. 2904.6, meaning that most nodes have lots of neighbors. To enable the constructed grid to contain more information of local structures, a relatively large K is appropriate for it. It is worth noting that the K in our experiments is far less than the average degree of the nodes in the dataset, which also demonstrates the efficiency of HIN2Grid. Another point worth highlighting is that although K=40 for the sparse PUBMED dataset looks larger than the other two sparse datasets, there are far more nodes in the PUBMED than in the other two datasets, i.e., more than 20 thousand, so a 'large' K is set for PUBMED. In general, the best

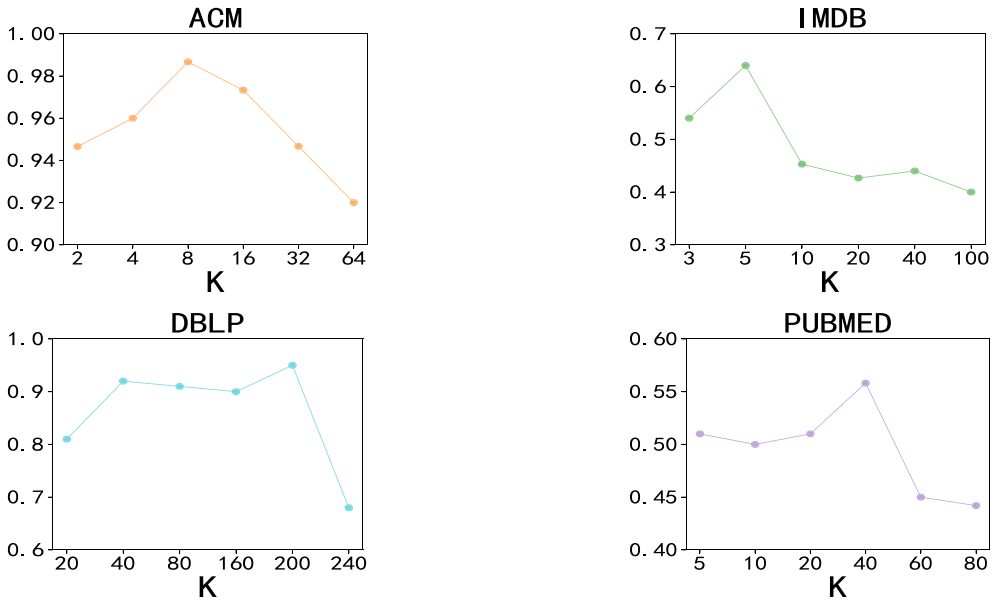

Fig. 6. Analysis of hyperparameter K on four datasets.

size of grids is different and varies from dataset to dataset. For selecting K, it should not only consider the sparsity of graphs but also consider the size of the dataset, e.g. the number of nodes.

6.6.2. Impact of θ_{bias}

The coefficient θ_{bias} controls the proportion of features from the central node and its neighbors in the constructed grid. We test the effect of it in four datasets and report the results in Fig. 7. We can observe that with the increase of the θ_{bias}, the performances raise first and then start to drop. On one hand, the model should aggregate features of neighbors to extract local information. On the other hand, the main features, i.e. features of the central node, are also essential for the constructed grid-like data. In particular, when θ_{bias} is set to 0 and 1, the generated grids only contains features from neighbors and central node, respectively. From the figure, we can observe that in these two settings, the performance of the model decreased sharply, which reflects the necessity of the proposed central fusion mechanism. Notably, the optimal θ_{bias} almost differs in different datasets, which indicates that the balance point between the features of the central node and its neighbors varies in different tasks.

6.6.3. Impact of grid-level attention

During the grid-like data construction, the influential neighbors are selected for each node and assembled together to generate the structured grid. However, the pixels of the grid contribute differently to node learning. Consequently, grid-level attention with multi-head mechanism is proposed to consider the importance of the pixels and stabilize the learning process. Fig. 8 shows the experiments we conduct on the grid-level attention mechanism. From the figure we observe that the model with attention mechanism achieves maximum relative improvements of 23% in IMDB, demonstrating that the effectiveness of the grid-level attention.

6.6.4. Impact of semantic-level attention

With semantic-level attention, the proposed HIN2Grid can learn the importance of meta-paths for the specific task automatically. To verify the effectiveness of it, we take IMDB and PUBMED as examples and report the attention weights in Fig. 10. From the figure, we observe that the attention weights of meta-paths in the PUBMED dataset have little difference. Among the meta-paths in PUBMED, the weight of DGD (Disease–Gene–Disease) has the highest value, in other words, themes of gene and disease are most closely related. In IMDB datasets, the importance of MDM (Movie–Director–Movie) is the highest among all meta-paths and the weight of MYM (Movie–Year–Movie) is the lowest one. It indicates that the styles of movies directed by the same director would be similar while the movie genres in the same year are difficult to be distinguished. What is more, Fig. 9 reports the experimental results on semantic-level attention. The figures show the advantages of the semantic attention mechanism which can improve the performance of the model significantly, about 5% to 50%. From the analysis above, attention mechanisms cannot only bring interpretability to the experimental results but also help the model to achieve great improvements.

.

Table 5
Experimental results on neighborhood selection.

Dataset	2-hop	3-hop	4-hop
DBLP	**0.89**	0.86	0.83
ACM	**0.96**	**0.96**	0.95
PUBMED	**0.47**	0.45	0.44
IMDB	**0.50**	0.49	0.47

6.6.5. Impact of neighborhood selection strategy

In order to explore the influence of neighborhood selection on the HIN2Grid, we use neighboring nodes within hops range in $\{2, 3, 4\}$ to construct the grid respectively. Besides, due to the sparsity of datasets is different, we set large enough K for four datasets to ensure that long-distant neighbors (e.g. fourth-order neighbors) of each node can be selected. As shown in Table 5, with the increase of the neighborhood order, the accuracy of the model declines, proving the stability and effectiveness of the selection strategy of HIN2Grid. Besides, although selecting higher-order neighboring nodes to construct grid-like data sometimes has little impact on the accuracy of the model, it takes more time to search for long-range nodes inevitably, which would affect the efficiency of the model.

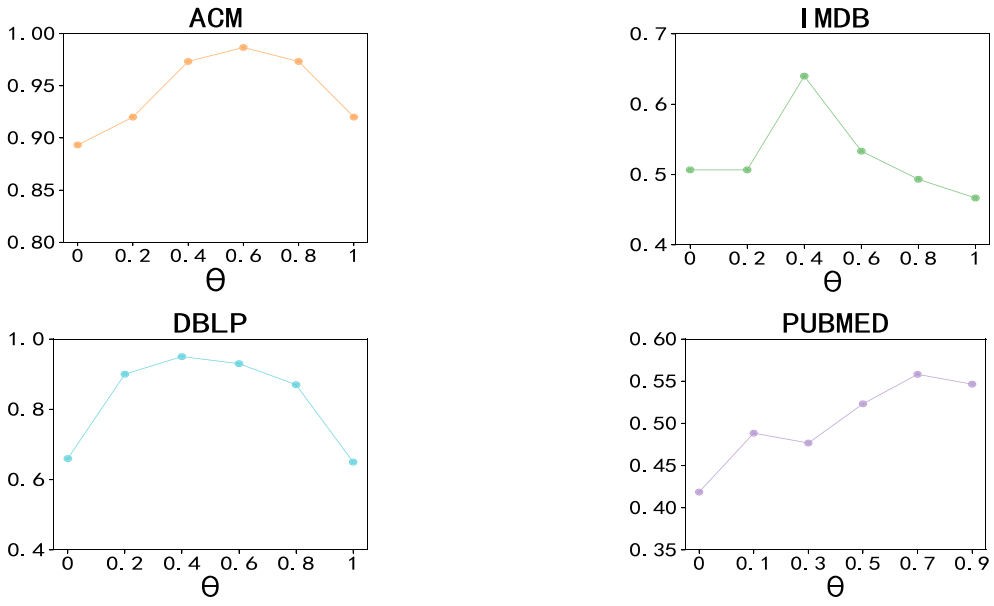

Fig. 7. Analysis of hyperparameter θ on four datasets.

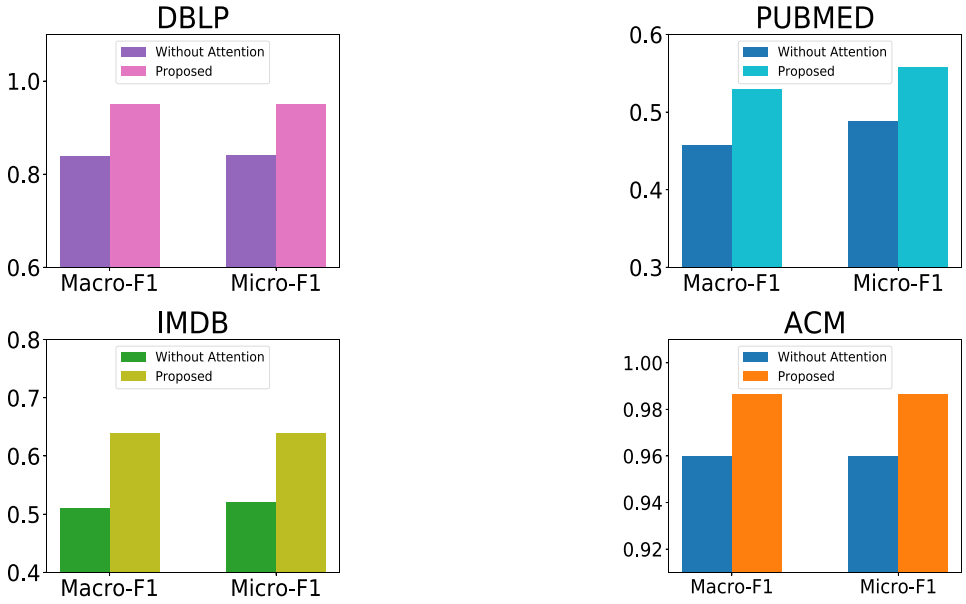

Fig. 8. Experiments of grid-level attention on four datasets.

7. Conclusion

In this paper, we propose a disentangled CNN-based framework named HIN2Grid for heterogeneous network embedding. Taking the heterogeneity and scalability of HINs into account, HIN2Grid firstly decomposes the original graph into several semantic-specific homogeneous subgraphs by meta-paths, then transforms them into grid-like data and takes it as the input to efficient CNNs. In addition, we also propose dual attention mechanisms to provide interpretability and robustness for the model. Extensive experiments on four datasets demonstrate that the proposed model has superior performance to baselines. In addition to its excellent accuracy, it also avoids the inherently coupled paradigm which leads to redundant and repeat computation in GCN-based models. As a result, HIN2Grid improves computational efficiency and reduces memory cost.

Although experimental results demonstrate the superiority of HIN2Grid over baselines, there is still room for improvement. In the proposed framework, the hyper-parameter K is fixed for all the nodes in the graph. However, when it comes to the graph that meets unusual settings, e.g., half nodes in the graph have lots of neighbors and the other half have few neighbors, the hyper-parameter may be hard to set. For future work, we will refine our framework by exploring other methods to involve certain hyper-parameters in the training phase of the model so that the appropriate values of them can be obtained automatically. Moreover, we will also explore the adoption of the proposed HIN2Grid for other graph-based applications such as recommendation and fraud detection.

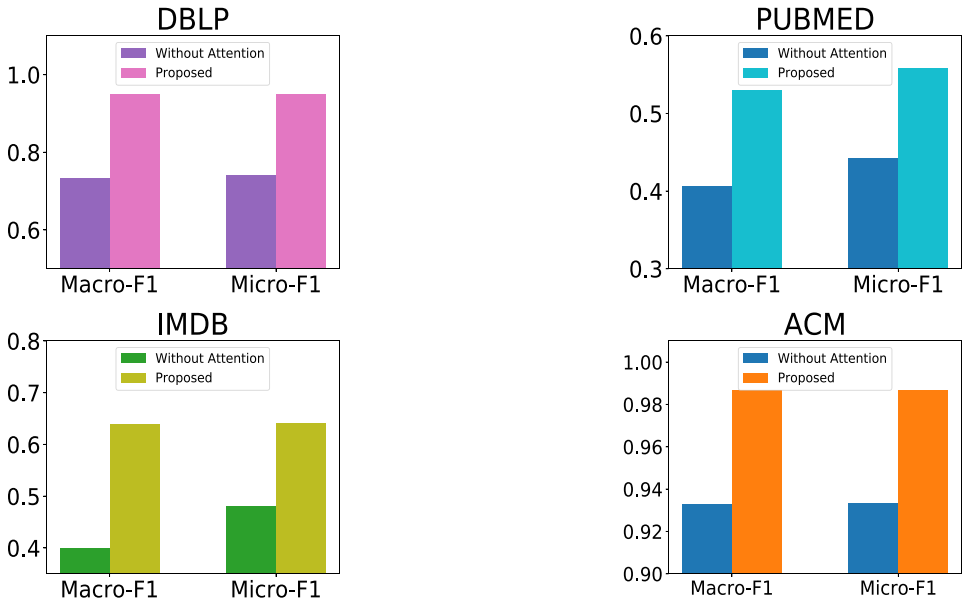

Fig. 9. Experiments of semantic-level attention on four datasets.

Fig. 10. Semantic-level attention values on two datasets.

CRediT authorship contribution statement

Ziyang Zhang: Conceptualization, Methodology, Data curation, Writing - original draft, Visualization. **Chuan Chen:** Supervision, Validation, Writing - review & editing. **Yaomin Chang:** Writing - review & editing. **Weibo Hu:** Formal analysis, Supervision. **Zibin Zheng:** Project administration. **Yuren Zhou:** Writing - review & editing. **Lei Sun:** Formal analysis.

Declaration of competing interest

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

Acknowledgments

The research is supported by the Key-Area Research and Development Program of Guangdong Province (No. 2020B010165003), the National Natural Science Foundation of China (62176269, 11801595), the Guangdong Basic and Applied Basic Research Foundation (2019A1515011043). This work is also supported by the UX center, Netease Games.

References

Ata, S. K., Fang, Y., Wu, M., Li, X. L., & Xiao, X. (2017). Disease gene classification with metagraph representations. *Methods*, *131*, 83–92.

Chen, J., Jiang, C., Wang, C., Zhou, S., Feng, Y., Chen, C., et al. (2020). Cosam: An efficient collaborative adaptive sampler for recommendation. arXiv preprint arXiv:2011.07739.

Chen, C., Li, Y., Qian, H., Zheng, Z., & Hu, Y. (2020). Multi-view semi-supervised learning for classification on dynamic networks. *Knowledge-Based Systems*, Article 105698.

Chen, J., Ma, T., & Xiao, C. (2018). Fastgcn: Fast learning with graph convolutional networks via importance sampling. arXiv preprint arXiv:1801.10247.

Chen, Z. M., Wei, X. S., Wang, P., & Guo, Y. (2019). Multi-label image recognition with graph convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5177–5186).

Chen, L., Wu, L., Hong, R., Zhang, K., & Wang, M. (2020). Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 27–34).

Chiang, W. L., Liu, X., Si, S., Li, Y., Bengio, S., & Hsieh, C. J. (2019). Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 257–266).

Dong, Y., Chawla, N. V., & Swami, A. (2017). metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 135–144).

Fei, H., Ren, Y., & Ji, D. (2020). Boundaries and edges rethinking: An end-to-end neural model for overlapping entity relation extraction. *Information Processing & Management*, *57*, Article 102311.

Fu, T. y., Lee, W. C., & Lei, Z. (2017). Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on conference on information and knowledge management* (pp. 1797–1806).

Gao, H., Wang, Z., & Ji, S. (2018). Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 1416–1424).

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. arXiv preprint arXiv:1704.01212.

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

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020). Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval* (pp. 639–648).

Hu, Z., Dong, Y., Wang, K., & Sun, Y. (2020). Heterogeneous graph transformer. In *Proceedings of the web conference 2020* (pp. 2704–2710).

Hu, L., Li, C., Shi, C., Yang, C., & Shao, C. (2020). Graph neural news recommendation with long-term and short-term interest modeling. *Information Processing & Management*, *57*, Article 102142.

Huang, J., Chen, C., Ye, F., Hu, W., & Zheng, Z. (2020). Nonuniform hyper-network embedding with dual mechanism. *ACM Transactions on Information Systems (TOIS)*, *38*, 1–18.

Huang, K., & Zitnik, M. (2020). Graph meta learning via local subgraphs. In *Advances in neural information processing systems (vol. 33)*.

Jia, Z., Lin, S., Ying, R., You, J., Leskovec, J., & Aiken, A. (2020). Redundancy-free computation for graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 997–1005).

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *Computer Science*.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324.

Lin, X., Quan, Z., Wang, Z. J., Ma, T., & Zeng, X. (2020). Kgnn: Knowledge graph neural network for drug-drug interaction prediction. In *IJCAI*.

Liu, M., Wang, Z., & Ji, S. (2020). Non-local graph neural networks. arXiv preprint arXiv:2005.14612.

Ma, C., Ma, L., Zhang, Y., Sun, J., Liu, X., & Coates, M. (2020). Memory augmented graph neural networks for sequential recommendation. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 5045–5052).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701–710).

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., & Welling, M. (2018). Modeling relational data with graph convolutional networks. In *European semantic web conference* (pp. 593–607). Springer.

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

Tang, X., Chen, L., Cui, J., & Wei, B. (2019). Knowledge representation learning with entity descriptions, hierarchical types, and textual relations. *Information Processing & Management*, *56*, 809–822.

Tang, J., Qu, M., & Mei, Q. (2015). Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1165–1174).

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web* (pp. 1067–1077).

Tangmunarunkit, H., Govindan, R., Jamin, S., Shenker, S., & Willinger, W. (2002). Network topology generators: Degree-based vs. structural. *ACM SIGCOMM Computer Communication Review*, *32*, 147–159.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In *International conference on learning representations*.

Vo, D. T., Al-Obeidat, F., & Bagheri, E. (2020). Extracting temporal and causal relations based on event networks. *Information Processing & Management*, *57*, Article 102319.

Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., et al. (2019). Heterogeneous graph attention network. In *The world wide web conference* (pp. 2022–2032).

Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., et al. (2019). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 968–977).

Xu, B., Huang, J., Hou, L., Shen, H., Gao, J., & Cheng, X. (2020). Label-consistency based graph neural networks for semi-supervised node classification. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval* (pp. 1897–1900).

Xu, L., Wei, X., Cao, J., & Yu, P. S. (2017). Embedding of embedding (eoe) joint embedding for coupled heterogeneous networks. In *Proceedings of the tenth ACM international conference on web search and data mining* (pp. 741–749).

Yang, J., & Leskovec, J. (2014). Overlapping communities explain core–periphery organization of networks. *Proceedings of the IEEE*, *102*, 1892–1902.

Ye, R., Li, X., Fang, Y., Zang, H., & Wang, M. (2019). A vectorized relational graph convolutional network for multi-relational network alignment. In IJCAI (pp. 4135–4141).

Zhao, X., Liu, F., Wang, J., Li, T., et al. (2017). Evaluating influential nodes in social networks by local centrality with a coefficient. *ISPRS International Journal of Geo-Information*, *6*, 35.

Zhu, S., Zhou, C., Pan, S., Zhu, X., & Wang, B. (2019). Relation structure-aware heterogeneous graph neural network. In *2019 IEEE international conference on data mining* (pp. 1534–1539). IEEE.