Knowledge-Based Systems 251 (2022) 109171

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Heterogeneous graph neural network for attribute completion

Kai Wang[a], Yanwei Yu[a,*], Chao Huang[b], Zhongying Zhao[c], Junyu Dong[a]

[a] College of Computer Science and Technology, Ocean University of China, Qingdao, Shandong 266100, China
[b] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong, China
[c] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong 266590, China

ARTICLE INFO

Article history:
Received 20 January 2022
Received in revised form 27 May 2022
Accepted 28 May 2022
Available online 16 June 2022

Keywords:
Network embedding
Attribute completion
Graph neural network
Heterogeneous networks
Attributed networks

ABSTRACT

Heterogeneous graphs consist of multiple types of nodes and edges, and contain comprehensive information and rich semantics, which can properly model real-world complex systems. However, the attribute values of nodes are often incomplete with many missing attributes, as the cost of collecting node attributes is prohibitively expensive or even impossible (*e.g.*, sensitive personal information). While a handful of graph neural network (GNN) models are developed for attribute completion in heterogeneous networks, most of them either ignore the use of similarity between nodes in feature space, or overlook the different importance of different-order neighbor nodes for attribute completion, resulting in poor performance. In this paper, we propose a general Attribute Completion framework for HEterogeneous Networks (AC-HEN), which is composed of feature aggregation, structure aggregation, and multi-view embedding fusion modules. Specifically, AC-HEN leverages feature aggregation and structure aggregation to obtain multi-view embeddings considering neighbor aggregation in both feature space and network structural space, which distinguishes different contributions of different neighbor nodes by conducting weighted aggregation. Then AC-HEN uses the multi-view embeddings to complete the missing attributes via an embedding fusion module in a weak supervised learning paradigm. Extensive experiments on three real-world heterogeneous network datasets demonstrate the superiority of AC-HEN against state-of-the-art baselines in both attribute completion and node classification. The source code is available at: https://github.com/Code-husky/AC-HEN.

1. Introduction

Many real-world complex applications are naturally represented in graph data structures, such as social networks, communication networks, brain neural networks and E-commerce networks. There has been a surge in interest in learning from these graph data, especially heterogeneous graphs. Heterogeneous graphs are composed of multiple types of nodes and edges, corresponding to describing various entities and their relationships in real-world complex systems. For instance, we can model citation network in ACM dataset as a heterogeneous network, where entities such as authors, papers and subjects are regarded as different types of nodes, and the relationships between them are represented as different edge types (*e.g.*, paper–paper, paper–author, paper–subject). Since heterogeneous graphs contain more comprehensive information and rich semantics, they can better model the real-world complex systems than conventional homogeneous graphs. Analyzing and mining what information in these networks has been a very active research topic for decades both in academia and industry.

Among various graph mining techniques, network embedding (or network representation learning), which projects high-dimensional network data to a low-dimensional continuous space while captures the intrinsic information of the network, is shown to be especially effective for various network analytical tasks [1–4]. Recently, graph neural networks (GNNs) [5,6] have demonstrated state-of-the-art performance and attracted considerable attention from researchers. To capture the rich neighborhood contexts, various GNN-based models have been developed to aggregate feature information from neighbor nodes, such as graph convolutional networks (GCN) [7], graph attention networks (GAT) [8], inductive graph learning (GraphSAGE) [9] and meta-path aggregated graph neural network (MA-GNN) [10]. To learn better node representations for various downstream tasks, all nodes' attributes are required in these GNN-based models. However, this is not always satisfied due to the problem of missing attributes. Absence of attributes in the real world is widespread. Some entity attributes require extremely expensive costs to obtain, while some entity attributes are almost impossible to obtain. For example, some users are unwilling to fill their gender while others do not disclose their age information in social networks or

* Corresponding author.
E-mail addresses: wangkai9521@stu.ouc.edu.cn (K. Wang), yuyanwei@ouc.edu.cn (Y. Yu), chaohuang75@gmail.com (C. Huang), zyzhao@sdust.edu.cn (Z. Zhao), dongjunyu@ouc.edu.cn (J. Dong).

https://doi.org/10.1016/j.knosys.2022.109171
0950-7051/© 2022 Elsevier B.V. All rights reserved.

E-commerce networks. Since more types of nodes are involved, this problem is more serious in heterogeneous networks, and thus it is usually impossible to obtain all the attributes of all types of nodes. Many studies [2,10,11] have shown that the attribute information of nodes has a great impact on various network analytical tasks for heterogeneous networks. Therefore, the missing attributes will significantly affect the performance of GNN-based models.

In this work, we are interested in completing the missing attributes of nodes in GNN-based model for heterogeneous networks. Towards this end, previous studies have made significant efforts on attribute completion with heterogeneous graph neural networks. For example, HGAT [12] leverages heterogeneous graph learning in a semi-supervised manner to complete users' missing attribute features. AGCN [13] joints item recommendation and attribute inference tasks and adaptively updates network embedding by considering both ground-truth attributes and the predicted attributes, which is guided by supervised information to facilitate the two tasks. HGNN-AC [14] first employs pre-trained existing heterogeneous network embedding models to get nodes' representation vectors. Then it uses the topology of the graph as guidance to complete the missing attributes for target nodes from the directly connected attributed nodes by weighted aggregation with attention mechanism. However, these existing methods still have three key limitations. *First*, most methods only consider the use of network topological structure for node attribute completion, but ignore the direct use of similarity between nodes in the feature space for attribute filling. In many cases, the attributes of the nodes are not completely absent, but partially missing. The similarity in the feature space can effectively help fill in the attributes, but sometimes GNN models on heterogeneous networks cannot achieve this, because nodes of the same type do not necessarily have edges. For example, in DBLP, there is no direct connection between authors. *Second*, although multi-layer GNN networks consider high-order neighbors, they ignore that the importance of different-order neighbors for attribute completion is different. It is obvious that neighbors of different orders must have different effects, and many studies have also shown that first-order neighbors are likely to have more contributes [14,15]. *Third*, in fact, GNN models are usually set to two or three layers, that is, higher-order neighbors are not actually considered for attribute completion. For example, HGNN-AC only uses the first-order neighbors to perform aggregation of attributes for no-attribute nodes. Nevertheless, high-order neighbors can be used as an effective supplement to first-order neighbors, especially heterogeneous networks where there are no edges between same typed nodes.

To tackle the aforementioned challenges, we propose **A**ttribute **C**ompletion framework for **HE**terogeneous **N**et-works (AC-HEN), which solves the problem of missing attributes of some types of nodes in heterogeneous networks by a learning manner. Specifically, AC-HEN consists of feature aggregation, structure aggregation and multi-view embedding fusion modules. AC-HEN first obtains the feature embeddings for target nodes by weighted aggregating their k nearest neighbor (kNN) in feature space. Then AC-HEN uses structure aggregation to learn the node embeddings considering both the first-order and high-order network structure, which distinguishes different contributions of different neighbor nodes by conducting weighted aggregation. Finally, the multi-view embeddings are used to complete the missing attributes via the embedding fusion module in a weak supervised learning paradigm. Our extensive experiments on three real-world heterogeneous networks show that our proposed AC-HEN is significantly better than state-of-the-art GNN-based models on both attribute completion and node classification tasks.

The key contributions of this work are summarized as follows:

- We propose a universal GNN-based attribute completion framework for heterogeneous networks, considering feature aggregation in feature space and first- and high-order neighbor aggregation in network structure space.
- We propose an attention-based high-order neighbor aggregation, which explicitly explores the different effects of high-order neighbor nodes on attribute completion for target nodes instead of multi-layer GNN model.
- We perform extensive experiments on three real-world heterogeneous networks to demonstrate the superiority of our proposed model when competing with state-of-the-art baselines. Furthermore, ablation studies also verify the rationality of our designed sub-modules.

2. Related work

Homogeneous Network Embedding. Network embedding is to embed the nodes in the graph to a low dimension space, in which the node embeddings preserve graph topology and node attribute similarities, such as random walk based methods [1], matrix factorization based approaches [16,17] and deep neural network models [18]. For GNNs, GCN [7] performs convolutional operations on graph-structured data and has recently achieved appealing performance in a variety of tasks ranging from node classification to link prediction. Graph Attention Network (GAT) [19] extends the convolutional operations in GCN with masked self-attention layers, which attends different weights to different neighborhoods. To capture the rich contextual neighbor messages, various GNN-based models have been developed to aggregate feature information from neighboring nodes, such as inductive graph learning (GraphSAGE) [9], simplifying graph convolutional network (SGC) [20] and adaptive multi-channel graph convolutional network (AM-GCN) [21].

Heterogeneous Network Embedding. Recently there are varieties of embedding methods for heterogeneous graphs. Most existing models focus on preserving the heterogeneous meta-path based structural information. Metapath2vec [22] generates heterogeneous neighborhood of target nodes from random node sequences obtained by meta-paths [23,24], and then uses skip-gram model to learn node embeddings. HHNE [25] further improves metapath2vec by embedding nodes into the hyperbolic space, where the nodes and their neighbors preserve triangle inequality in heterogeneous information network. HERec [26] also obtains heterogeneous node sequences from meta-path based random walks and then employs matrix factorization (MF) to integrate final node embeddings. HAN [2] uses the hierarchical attention mechanism on heterogeneous networks to learn the contributions of different meta-paths. Recently, MA-GNN [10] applies node content transformation, intra-metapath aggregation and inter-meta-path aggregation to generate node embeddings for heterogeneous graphs. However, these methods do not provide any solutions for the problem of missing attributes in heterogeneous graphs.

Traditional Attribute Completion. Existing methods usually leverage features from texts, relation and behaviors to predict missing features. [27] employs LDA to extract users' semantics based on their interests and searches the users who share the similar interests, which can complete the missing attributes based on the similar users in feature space. VIAL [28] completes the missing attributes based on social-behavior-attribute network model. It first spreads the weight from the source user to other users, which can illustrate the different contributions of the neighbors. Then, the missing values are completed by highest weight of the neighbor's attributes. LSVM [29] completes the missing attributes from user features and attribute relations. Both lexical and visual features make contributions to feature completion. UNHB [30,31] aims at inferring node missing attributes in an supervised learning manner with features extracted from user-generated texts and user names. GBCM [32] proves that friendship links and group affiliations carry the latent information of missing attributes. It maps attribute completion to link-based and group-based classification problems. SAN [33] completes the missing attributes of nodes by link prediction with social attribute network.

GNN-based Attribute Completion. Recently, several deep graph neural network models aiming to complete missing feature values by graph embedding are proposed. HGAT [12] develops an attention-based solution based on heterogeneous graph learning to complete users' missing features in a semi-supervised manner. AGCN [13] joints item recommendation and attribute inference tasks and adaptively updates network embedding by considering both ground-truth attributes and the estimated attribute, which is guided by supervised information to facilitate the two tasks. HGNN-AC [14] first employs pre-trained existing heterogeneous network embedding models to get nodes' structural embedding. Then it uses the topological structure of the graph as guidance to complete the missing attributes from the directly connected attributed nodes by weighted aggregation of attention mechanism. However, all of these methods aggregate high-order neighborhood features based on multiply GCN layers, which can lead over-smoothness. In addition, these methods focus on structure aggregation and ignore feature-space aggregation.

3. Problem definition

A network, denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, is composed of a node set $\mathcal{V}$ and an edge set $\mathcal{E}$. Each edge $e_{ij} \in \mathcal{E}$ is an ordered pair $e_{ij} = (v_i, v_j)$ and contains a weight $w_{ij} > 0$, which represents the strength of the relation. $\mathcal{G}$ can be divided into two categories: undirected network where we have $e_{ij} \equiv e_{ji}$ and $w_{ij} \equiv w_{ji}$ and directed network where we have $e_{ij} \neq e_{ji}$ and $w_{ij} \neq w_{ji}$.

Definition 1 (*Heterogeneous Network*)**.** A heterogeneous network is defined as a network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{O}$ and an edge type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{R}$, in which $\mathcal{O}$ denotes the set of all node types and $\mathcal{R}$ represents the set of all edge types. If $|\mathcal{O}| + |\mathcal{R}| > 2$, the network is ***heterogeneous***.

Definition 2 (*Attributed Network*)**.** An attributed network is defined as a network $\mathcal{G}$ affiliated with an attribute feature matrix, *i.e.*, $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$, where $\mathbf{X} \in \mathbb{R}^{n\times m}$ is the attribute feature matrix, each row corresponding to the node feature vector of node v_i. Here, n and m denote the number of nodes and the dimension of attributes, respectively.

Given an attributed network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$, if network $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}\}$ is *heterogeneous*, then $\mathcal{G}$ is an *attributed heterogeneous network.*

Definition 3 (*Attributed Heterogeneous Network with Missing Features*)**.** An attributed heterogeneous network with missing features is defined as a network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathbf{M}\}$, where $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$ is an attributed heterogeneous network, and $\mathbf{M} \in \{0, 1\}^{n\times m}$ is a mask matrix where the attribute feature of $\mathbf{X}_{ij}$ can be observed only if $\mathbf{M}_{ij} = 1$. On the contrary, $\mathbf{M}_{ij} = 0$ indicates the attribute feature $\mathbf{X}_{ij}$ is missing.

Based on the above definitions, we next formally define our studied problem of attribute completion for heterogeneous networks.

Table 1
Main notations and their definitions.

Notation	Definition
$\mathcal{G}$	The given network
$\mathcal{V}, \mathcal{E}$	The node set and edge set of $\mathcal{G}$
$\mathcal{O}, \mathcal{R}$	The node type set and edge type set of $\mathcal{G}$
$\mathbf{X}$	The node attribute feature matrix of $\mathcal{G}$
$\mathbf{M}$	The mask matrix
$\mathbf{A}$	The adjacency matrix
$\mathbf{Z}$	The node embedding
z_u^f	The feature embedding of target node v_u
z_u^n	The neighborhood embedding of target node v_u
z_u^h	The high-order embedding of target node v_u

Problem (*Heterogeneous Network Attribute Completion*)**.** Given an attributed heterogeneous network with missing features $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathbf{M}\}$, the goal of attribute completion is to predict missing attribute features of target nodes at $\mathbf{M}_{ij} = 0$.

Key notations are summarized in Table 1.

4. Methodology

Fig. 1 presents the overview of our proposed AC-HEN framework, which consists of three key components: (i) *feature aggregation*, (ii) *structure aggregation* and (iii) *multi-view embedding fusion module*. *Feature aggregation* aims to obtain the feature embeddings of target nodes by aggregating their k-nearest neighbor nodes in feature space. *Structure aggregation* is to learn the embeddings of target nodes by considering the first-order and high-order structural proximities in the heterogeneous network. *Multi-view embedding fusion module* is used to fuse the learned multi-view embeddings in both feature space and structure space to predict the missing attribute features.

4.1. Feature aggregation module

The nodes in networks can be divided in two categories: target nodes with incomplete attributes and source nodes with complete attributes. The attributes of the target nodes can be also divided in two categories: missing attributes and existing attributes. Intuitively, we can use the source nodes to complete the missing attributes of the target nodes. However, networks generally contain a large number of nodes, and thus it is impossible to use all the source nodes to predict the missing attributes of the target nodes. Therefore, we use a k-nearest neighbor (k-NN) method [34] to select the source nodes to generate the feature embeddings for target nodes. Specifically, we compute the similarity scores between source nodes and target nodes by the existing attributes of the target nodes. For instance, we can use the cosine similarity between two vectors to measure the feature similarity:

$$S_{u,v} = \frac{x_u^e \cdot x_v^e}{\mid x_u^e \mid\mid x_v^e \mid}, \tag{1}$$

where x_u^e are the existing attributes of target node v_u and x_v^e are the corresponding attributes of source node v_v.

Then, we choose the top-k similar source nodes for each target node. However, there may exist multiple factors that affect the impact of the source nodes on the target nodes, such as whether two nodes are directly connected, and the weights on edges between nodes may be different. Therefore, different source nodes have different impacts on the target nodes. To model the impacts of source nodes, we use k learnable parameters to dynamic adjust the weights of the top-k source nodes for the target node:

$$z_u^f = \frac{1}{k} \sum_{v_v \in \mathcal{N}_k(v_u)} w_v^u x_v, \tag{2}$$

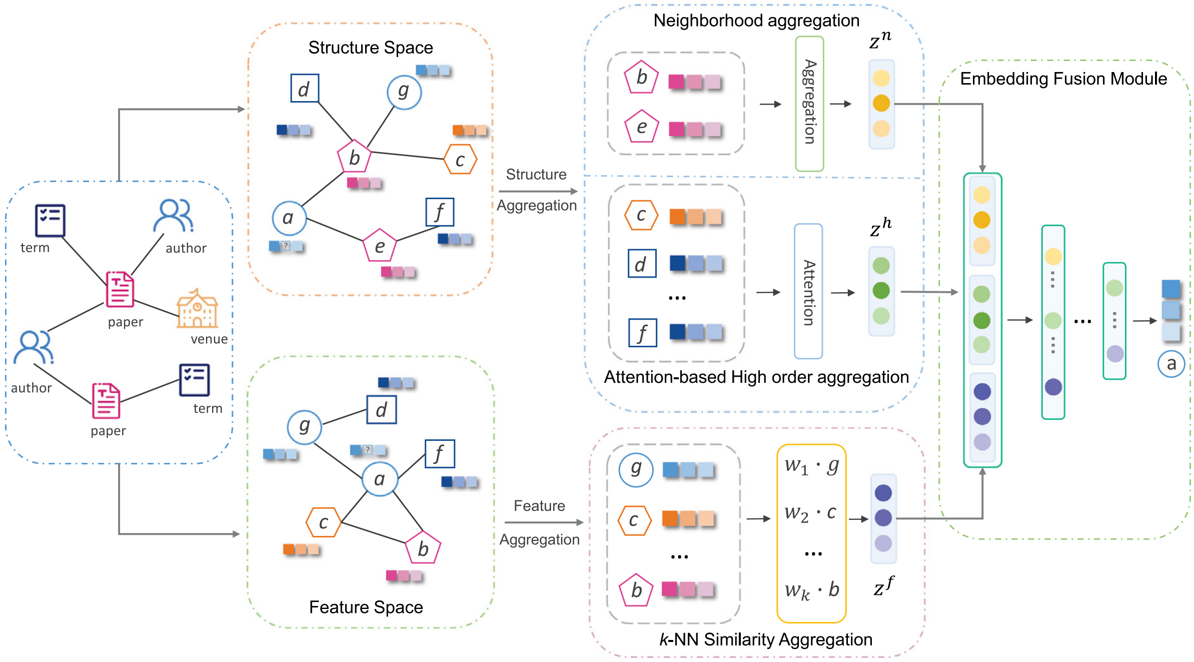


Fig. 1. The overview of the proposed AC-HEN.

where $\mathcal{N}_k(u)$ denotes the set of top-k similar neighbors of target node v_u, x_v are the attribute features of source node v_v, and w_v^u is the learnable weight for node v_v *w.r.t.* target node v_u. z_u^f is the aggregated feature embedding for target node v_u in feature space.

4.2. Structure aggregation module

Although feature aggregation contributes to predict the missing attributes of target nodes in the feature space, it cannot cover all potential useful source nodes. For example, in academic networks, authors and papers are directly connected, but the similarity between their attributes may be low. However, the connected papers are essential for the completion of the missing attributes of authors. Therefore, we propose to leverage the topological structure between nodes to guide to complete the missing attributes for target nodes. To model both the first-order and high-order structure, our proposed *structure aggregation module* consists of two sub-modules: *neighborhood aggregation* and *attention-based high-order aggregation*.

4.2.1. Neighborhood aggregation

For some nodes without attributes, previous works [2,10,35] solve this problem by averagely aggregating the attributes of directly connected neighbors. However, we observe that for attribute completion, directly connected neighbors also have different effects, since these nodes may be of different types, or their local topological structures are different. Therefore, the neighborhood aggregation process for attribute completion needs to consider weighted aggregation. Inspired by SGC [20], the specific aggregation process is defined as follows:

$$\mathbf{H}^{(l)} = \mathbf{A} \cdot \mathbf{H}^{(l-1)} \cdot \mathbf{W}^{(l)} \tag{3}$$

where $\mathbf{W}^{(l)}$ is the learnable weight matrix for the lth layer, $\mathbf{H}^{(l)}$ is the output of the lth layer (*i.e.*, hidden representation of the network), and $\mathbf{H}^{(0)} = \mathbf{X}$. Eventually, $z_u^n = \mathbf{H}_u^{(l)}$ is the neighborhood aggregation embedding of the target node v_u.

Considering that the multi-layer GCN will flatten the feature similarity of all nodes, and that the first-order neighbor contributes more to the attribute completion, we only set $l = 1$, that is, we only use one-layer GCN to learn the embeddings of aggregating first-order neighbors for target nodes.

4.2.2. Attention-based high-order aggregation

To further use higher-order neighbors structure to complete the missing attributes, we also propose an attention-based high-order aggregation to explicitly explore the effects of high-order neighbors on attribute completion of target nodes.

Specifically, we regard each target node as root node. Based on the root node, we sample random walks of fixed length *len*. Let c_i denote the ith node in the walk, starting with $c_0 = v_u$. Node c_i is sampled by the following distribution:

$$P(c_i = v_t \mid c_{i-1} = v_v) = \begin{cases} \frac{w_{vt}}{\sum_{e_{vi} \in \mathcal{E}} w_{vi}} & if\ e_{vt} \in \mathcal{E} \\ 0 & otherwise, \end{cases} \tag{4}$$

where w_{vt} is the weight on the edge e_{vt} from node v_v to node v_t.

For each sampled random walk, we drop the directly connected nodes of the root node, and define the remaining nodes as high-order neighbor nodes. Obviously, higher-order neighbor nodes have different effects on target nodes. Therefore, we propose an attention-based high-order aggregation approach.

Given a node list $\mathcal{V}_u = (v_u, v_{v_1}, v_{v_2}, \ldots, v_{v_h})$, where v_u is the root node (target node) and v_{v_i} is a high-order neighbor of v_u in the sampled random walks. The importance of each high-order neighbor node v_{v_i} to node v_u can be learned through an attention layer. The formulation of attention layer is as follows:

$$\begin{aligned} \mathbf{Z}_{\mathcal{V}_u} &= \mathrm{Softmax}\left(\frac{\mathbf{Q}_{\mathcal{V}_u}(\mathbf{K}_{\mathcal{V}_u})^{\mathsf{T}}}{\sqrt{d}}\right)\mathbf{V}_{\mathcal{V}_u}, \\ \mathbf{Q}_{\mathcal{V}_u} &= \mathbf{H}_{\mathcal{V}_u}\mathbb{W}^Q, \\ \mathbf{K}_{\mathcal{V}_u} &= \mathbf{H}_{\mathcal{V}_u}\mathbb{W}^K, \\ \mathbf{V}_{\mathcal{V}_u} &= \mathbf{X}_{\mathcal{V}_u}\mathbb{W}^V, \end{aligned} \tag{5}$$

where $\mathbf{H}_{\mathcal{V}_u}$ represents the hidden embeddings for $\mathcal{V}_u$ after neighborhood aggregation, $\mathbf{X}_{\mathcal{V}_u}$ denotes the attribute features of $\mathcal{V}_u$, $\mathbb{W}^Q$, $\mathbb{W}^K$, $\mathbb{W}^V$ are the learnable projection matrices, and d is the dimension of hidden embeddings. Softmax($\cdot$) is a softmax function, which is used to normalize the weights.

As shown in Eq. (5), we learn the weights for each high-order neighbor according to the similarities between the hidden embeddings of target node and its high-order neighbor nodes, because the hidden embeddings imply the structural proximities between nodes. Then, we aggregate the attribute features of the high-order neighbor nodes with the learned weights.

Therefore, the embedding of target node v_u *w.r.t.* its high-order neighbor nodes in structure space is defined as:

$$z_u^h = \mathbf{Z}_{\mathcal{V}_u}(u), \tag{6}$$

where $\mathbf{Z}_{\mathcal{V}_u}(u)$ denotes the aggregation embedding of v_u in $\mathbf{Z}_{\mathcal{V}_u}$.

In addition, the above attention can be extended to a multi-head attention [19,36] to capture multiple dependencies between target nodes and their high-order neighbor nodes (brought by the heterogeneity of networks). Then, the obtained embeddings of target node v_u from multi-head attention mechanism are sent to a pooling layer to get the final aggregation embedding:

$$z_u^h = \text{Mean-Pooling}\left(z_u^{(1)}, z_u^{(2)}, \ldots, z_u^{(\#head)}\right), \tag{7}$$

where $z_u^{(i)}$ is the embedding of node v_u obtained from ith head, and #*head* is the number of heads.

4.3. Multi-view embedding fusion module

Now we have three specific embeddings for each target node v_u, *i.e.*, z_u^f, z_u^n, and z_u^h. Considering the missing attributes of the target nodes can be correlated with one of them or even their combinations, we use the Multilayer Perception (MLP) to predict the missing attributes by fusing the learned multi-view embeddings. The formulation of multi-view embedding fusion module is as follows:

$$\begin{aligned} z_u &= \text{Concat}(z_u^f, z_u^n, z_u^h), \\ \hat{\mathbf{X}}_u &= MLP(z_u), \end{aligned} \tag{8}$$

where Concat(, ,) denotes the concatenate function, and $\hat{\mathbf{X}}_u$ is the predicted attributes of target node v_u.

4.4. Model optimization

Our goal in this work is to complete the missing attributes for heterogeneous networks, and we expect the completed attributes can be close to the ground truth attributes and show a performance as good as the ground truth attributes in the downstream tasks. To evaluate these new attributes predicted by the proposed model, following [14], we adopt a strategy of dropping some attributes. That is, we first randomly drop some attributes of nodes and then reconstruct these attributes by the proposed model. In this way, a completion loss can be calculated between reconstructed attributes and the ground truth attributes, which guides the parameter learning of our attribute completion model. More specifically, we randomly divide the nodes into two categories: target nodes and the source nodes. For target nodes, we randomly drop their attributes and set the corresponding mask matrix elements to 0. Then we use the source nodes to complete the missing attributes of the target nodes. Take ACM dataset as example, we randomly choose 20% author nodes as target nodes and drop 50% attributes of target nodes.

Specifically, we only update the missing attribute values with the predicted results, and the existing attributes keep the same. The completed attributes can be calculated as follows:

$$\hat{\mathbf{X}}^c = \mathbf{X} \odot \mathbf{M} + \hat{\mathbf{X}} \odot (\mathbf{E} - \mathbf{M}), \tag{9}$$

where $\hat{\mathbf{X}}$ denotes the predicted attribute matrix by the attribute completion model, $\odot$ is the element-wise product operation, *i.e.*, Hadamard product, and $\mathbf{E}$ presents the all-ones matrix.

Based on the strategy of dropping attributes, we propose a supervised loss to optimize the parameters of our attribute completion model. Specifically, we adopt Euclidean distance between completed attributes and ground truth attributes of target nodes to design the loss function as:

$$\mathcal{L} = \frac{1}{\mid \mathcal{V}_{target} \mid} \sum_{v_u \in \mathcal{V}_{target}} \sqrt{(\hat{\mathbf{X}}_u^c - \mathbf{X}_u^g)^2} \tag{10}$$

where $\mathcal{V}_{target}$ denotes the set of target nodes, $\mathbf{X}_u^g$ is the ground truth attributes of node v_u, and $\hat{\mathbf{X}}_u^c$ is the completed attributes for node v_u.

4.5. Time complexity analysis

We now analyze the time complexity of our proposed AC-HEN for completing the missing attributes in heterogeneous networks. AC-HEN is mainly composed of three sub-modules including feature aggregation, neighborhood aggregation and attention-based high-order aggregation. For feature aggregation, the time complexity of computing feature similarity between nodes is $O(n^2\, m)$, where n is the number of nodes in the graph, and m is the dimension of node attribute features. For neighborhood aggregation, we only perform one-layer simplified GCN to aggregate first-order neighbors' features. Therefore, the computational complexity is $O(|\mathcal{E}|\, d)$ for the GCN operation, where $|\mathcal{E}|$ is the number of edges in the graph and d is embedding dimension. For attention-based high-order aggregation, the computational complexity of sampling high-order nodes of target node is $O(nh)$, where h is the number of the high-order neighbor nodes. The computational complexity of multi-head self-attention is $O(\eta nhd)$, where η is the number of heads. Hence the main time complexity of attention-based high-order aggregation is $O(\eta nhd)$. Therefore, the total time complexity of AC-HEN is $O(n^2 m + |\mathcal{E}|\, d + \eta nhd)$. Nevertheless, the three sub-modules of our model are independent of each other, that is, they can be performed in parallel, so the total time complexity of our AC-HEN can be reduced to $\max(O(n^2\, m), O(|\mathcal{E}|\, d), O(\eta nhd))$.

5. Experiment

In this section, we first introduce the details of three evaluation datasets and the competitor algorithms. We study the effectiveness of our proposed model in completing missing attributes on these datasets compared to state-of-the-art baselines. Then we focus on downstream node classification task to evaluate the performance of the proposed model against other state-of-the-art methods. Finally, ablation study and parameter sensitivity are discussed.

5.1. Datasets

- DBLP.[1] This is a subset extracted from DBLP dataset, which contains 14 328 papers, 4057 papers, 20 venues and 8789 terms [14]. In this dataset, the attributes of papers are bag-of-words representation of their keywords, and the attributes of authors are bag-of-words representations of affiliates, titles and keywords extracted from their published papers. The attributes of terms uses no computer-science-specialized pre-trained word vectors and the attributes of venues uses one-hot vectors. Authors are grouped into four research areas based on the conferences they published.
- ACM.[2] This is a subset extracted from ACM [14], which comprises 4019 papers, 7167 authors and 60 subjects. In

[1] https://dblp.uni-trier.de/

[2] http://dl.acm.org/

Table 2
Statistics of experimental datasets.

Datasets	Nodes	Edges	Attributes
DBLP	#author(A): 4057 #paper(P): 14328 #term(T): 7723 #venue(V): 20	# A-P: 19645 # P-T: 85810 # P-V: 14328	A: missing
ACM	#paper(P): 4019 #author(A): 7167 #subject(S): 60	# P-P: 9615 # P-A: 13407 # P-S: 4019	A: missing
IMDB	#movie(M): 4278 #director(D): 2081 #actor(A): 5257	# M-D: 4278 # M-A: 12828	M: missing

this dataset, the attributes of papers are bag-of-words representations of their keywords, and the attributes of authors are bag-of-words representations of affiliates, titles and keywords extracted from their published papers. For subjects, their attributes are bag-of-words representations of keywords from their connected papers. The authors are categorized into three classes according to the conference where they published the most papers.

- IMDB.[3] This is a subset of IMDB that includes 4780 movies, 5841 actors and 2269 directors [14]. In this dataset, the attributes of movies are bag-of-words representations of their plots. The attributes of actors and directors are bag-of-words representations of movies they have appeared in and directed, respectively. We divide all movies into three classes according to their genres.

Notice that we assume that some attributes of some authors on ACM and DBLP datasets are missing, and some attributes of some movies on IMDB are missing. The statistics of three evaluation datasets are summarized in Table 2.

5.2. Baselines

To evaluate the performance of our proposed model, we compare it with two categories of baselines: traditional completion methods and GNN-based methods.

Four traditional completion methods are:

- **Matrix Completion (MC)** [37] - MC is able to recover most low-rank matrices from incomplete sets of entries, which can fill in the missing entries of a partially observed matrix.
- **Expectation Maximization (EM)** [38] - EM algorithm is an iterative optimization method to produces maximum likelihood estimates of unknown values when there is a many-to-one mapping from an underlying distribution to the distribution governing the observation.
- **Multilayer Perceptron (MLP)** [39] - MLP is a four-layer fully connected neural network, where each dimension of the output in each layer is weighted by each dimension of the input, and an activation function is inserted in the middle to enable the model to learn nonlinear relationships.
- **Support Vector Regression (SVR)** [40] - SVR is characterized by the use of kernels, sparse solution and Vapnik–Chervonenkis control of the margin and the number of support vectors. Although less popular than SVM, SVR has been proven to be an effective tool in real-value function estimation.

Three state-of-the-art GNN-based methods are:

- **HGAT** [12] - HGAT learns user representation in heterogeneous graphs by embedding both the graph structure and node features with attention mechanism, and then trains an end-to-end semi-supervised predictor using partial labels of users to infer user profiles.
- **AGCN** [13] - AGCN recursively adjusts the graph embedding learning parameters with the previously learned attribute values to optimize both attribute inference and item recommendation.
- **HGNN-AC** [14] - HGNN-AC is a state-of-the-art attribute completion method based on heterogeneous network embedding. It first uses existing heterogeneous network embedding methods to learn node embedding, and then leverages the topological structure of the graph as guidance to predict attributes for target nodes by aggregating the attributes of neighbor nodes with attention operation.

5.3. Experimental setting

For all semi-supervised learning methods, on each dataset, we divide the attributed nodes into training, validation and testing sets by 20%, 40%, 40%, respectively. We set the dropout ratio to 50% for the target nodes. For example, when training the model on DBLP, we choose 20% nodes (*e.g.*, authors) and randomly dropout the 50% attributes of them. The same processing is performed on validation and test sets. For our model, the attention in high-order aggregation module is set to a multi-head attention with the number of attention heads $\#head = 8$. We set k to 4 for the number of nearest neighbors in feature aggregation. We set *len* to 15 for the length of random walks, and select first sampled $h = 5$ high-order neighbor nodes. We use Adam to optimize our model, and the initial learning rate is set to 0.001 during model training. We use early stopping mechanism to stop the model learning once the performance decreases in the validation data.

For a fair comparison, we set the embedding dimension to 256 for all embedding-based attribute completion model. We use the source codes released by authors for baseline evaluation. Specifically, we use the parameter settings provided in their paper, and parameters of all baselines are tuned to be optimal. In experiments, we repeat each experiment 10 times for all methods to report average results. The source code of our model is available at https://github.com/Code-husky/AC-HEN.

5.4. Performance on attribute completion

For attribute completion, we choose two evaluation metrics – *Heat Kernel* [21] and *Pearson correlation coefficient* [41], which are defined as:

$$\begin{aligned} Heat\ Kernel &= e^{-\frac{\|\mathbf{X}_u^c - \mathbf{X}_u^g\|^2}{t}}, \\ Pearson\ Correlation &= \frac{cov(\mathbf{X}_u^c, \mathbf{X}_u^g)}{\sigma_{\mathbf{X}_u^c}, \sigma_{\mathbf{X}_u^g}} \end{aligned} \tag{11}$$

where t is the time parameter in heat conduction equation, $cov(\mathbf{X}_u^c, \mathbf{X}_u^g)$ is the covariance of $\mathbf{X}_u^c$ and $\mathbf{X}_u^g$, and $\sigma_{\mathbf{X}_u^c}$ and $\sigma_{\mathbf{X}_u^g}$ are the variances of $\mathbf{X}_u^c$ and $\mathbf{X}_u^g$, respectively. *Heat Kernel* directly calculates the similarity between the completed attributes and ground-truth attributes. Following [21], we set $t = 2$ in *Heat Kernel*. *Pearson correlation* measures the correlation between the completed attributes and ground-truth attributes. For both metrics, the larger value means the better performance. The experimental results of all methods on three datasets are reported in Table 3. The best results are shown in bold.

As we see, our AC-HEN significantly outperforms all baseline methods for attribute completion on three datasets. Specifically, AC-HEN achieves average 7.81% and 15.36% improvement in

[3] https://www.imdb.com/

Table 3
Results of attribute completion in terms of heart kernel and Pearson correlation coefficient on three datasets.

Method	ACM		DBLP		IMDB	
	Heat Kernel	Correlation	Heat Kernel	Correlation	Heat Kernel	Correlation
MC	0.6747 ± 0.0032	0.2369 ± 0.0051	0.3641 ± 0.0045	0.0924 ± 0.0002	0.5158 ± 0.0195	0.1168 ± 0.0079
EM	0.7534 ± 0.0112	0.2446 ± 0.0121	0.2611 ± 0.0102	0.1824 ± 0.0104	0.4037 ± 0.0112	0.1489 ± 0.0104
MLP	0.7739 ± 0.0091	0.3119 ± 0.0102	0.2987 ± 0.0131	0.1486 ± 0.0124	0.3724 ± 0.0095	0.1429 ± 0.0072
SVR	0.7376 ± 0.0074	0.3029 ± 0.0079	0.2819 ± 0.0201	0.1504 ± 0.0156	0.4372 ± 0.0212	−0.0121 ± 0.0011
HGAT	0.7931 ± 0.0101	0.3723 ± 0.0108	0.3821 ± 0.0141	0.3541 ± 0.0150	0.5815 ± 0.0991	0.2965 ± 0.0105
AGCN	0.7832 ± 0.0021	0.4433 ± 0.0024	0.3942 ± 0.0045	0.3872 ± 0.0051	0.5721 ± 0.0045	0.2954 ± 0.0051
HGNN-AC	0.8057 ± 0.0052	0.4632 ± 0.0061	0.4241 ± 0.0054	0.4098 ± 0.0048	0.5874 ± 0.0021	0.3219 ± 0.0027
AC-HEN	**0.8903 ± 0.0028**	**0.5327 ± 0.0045**	**0.4556 ± 0.0031**	**0.4691 ± 0.0029**	**0.6171 ± 0.0018**	**0.3754 ± 0.0012**

Table 4
P-values of t-test for heat kernel and Pearson correlation on attribute completion.

Method	ACM		DBLP		IMDB	
	Heat Kernel *p*-value	Correlation *p*-value	Heat kernel *p*-value	Correlation *p*-value	Heat kernel *p*-value	Correlation *p*-value
MC	$6.79 * e^{-15}$	$3.39 * e^{-16}$	$1.27 * e^{-9}$	$8.20 * e^{-17}$	$1.46 * e^{-11}$	$7.88 * e^{-16}$
EM	$7.98 * e^{-12}$	$1.08 * e^{-16}$	$4.48 * e^{-12}$	$4.58 * e^{-15}$	$5.69 * e^{-16}$	$1.27 * e^{-14}$
MLP	$3.76 * e^{-12}$	$3.40 * e^{-16}$	$5.27 * e^{-12}$	$1.16 * e^{-15}$	$3.09 * e^{-16}$	$2.44 * e^{-14}$
SVR	$2.97 * e^{-13}$	$1.84 * e^{-15}$	$2.32 * e^{-12}$	$2.86 * e^{-15}$	$2.02 * e^{-13}$	$3.17 * e^{-16}$
HGAT	$1.26 * e^{-11}$	$4.06 * e^{-14}$	$1.70 * e^{-8}$	$1.10 * e^{-12}$	$1.45 * e^{-8}$	$7.52 * e^{-10}$
AGCN	$2.67 * e^{-12}$	$1.21 * e^{-11}$	$1.76 * e^{-8}$	$1.50 * e^{-12}$	$1.20 * e^{-7}$	$1.76 * e^{-10}$
HGNN-AC	$1.50 * e^{-10}$	$6.39 * e^{-11}$	$1.75 * e^{-7}$	$3.32 * e^{-10}$	$5.63 * e^{-7}$	$1.44 * e^{-8}$

terms of *Heat Kernel* and *Pearson correlation* in comparison to the best performed baseline (*i.e.*, HGNN-AC) across all datasets. The main reason is that our AC-HEN considers neighbor aggregation in both feature space and network structure space. Feature aggregation guides AC-HEN to use cosine similarity to aggregate the attributes of top-k similar neighbors in feature space, which may result in the large improvement in *Pearson correlation*. Additionally, attention-based high-order aggregation further captures the effects of high-order neighbors on attribute completion for target nodes, improving model performance in both two evaluation metrics.

We also observe that GNN-based methods (*i.e.*, HGAT, A-GCN and HGNN-AC) are obviously better than traditional completion methods (*i.e.*, MC, EM, MLP and SVR). This is because the GNN-based methods can effectively capture similar feature information through network structures, as well as the propagation of information. The traditional methods only fill in the missing values based on the attribute matrix, which leads to lower performance for attribute completion. Among the baselines, HGNN-AC performs the best. This may be because it does not use multi-layer GNN, but only considers first-order neighbors and distinguishes the importance of different neighbors. Inspired by this idea, we use only one layer to aggregate first-order neighbors in GCN, and sample high-order neighbors through random walks, avoiding multi-layer GCN and in same time taking into account the impact of high-order neighbors on attribute completion in heterogeneous networks. Therefore, we can empirically conclude the superiority of our AC-HEN for attribute completion, as AC-HEN can aggregate the first-order neighbors', high-order neighbors' and feature neighbors' information simultaneously.

We next employ t-test to further verify if there is a significant difference between the result of AC-HEN and those of baselines over three datasets on attribute completion. Specifically, the null hypothesis H_0 and the alternative hypothesis H_1 for each pair of methods are defined:

$$\begin{aligned} &H_0 : A \approx B, \\ &H_1 : A < B, \end{aligned} \tag{12}$$

where B denotes the result of AC-HEN, and A represents the results of a specific baseline. Namely, H_0 means that the result of AC-HEN is equal to that of the baseline, and H_1 means that AC-HEN is significantly better than the baseline. We calculate *p-value* for each test, and the hypothesis is checked at $p = 0.05$ significance level. The *p-values* of t-test for Heat Kernel and Pearson Correlation on attribution completion task are shown in Table 4.

As we see, all p-values of t-test for Heat Kernel and Pearson Correlation on three datasets are significantly less than 0.05. Therefore, we can reject the null hypothesis H_0 and accept alternative Hypothesis H_1. That is, our proposed AC-HEN is significantly better than all baselines in terms of Heat Kernel and Pearson Correlation. In summary, this experiment confirms that the improvement of our proposed model over state-of-the-art baselines in attribute completion is statistically significant.

5.5. Performance on node classification

Node classification is an important downstream network analytical task to evaluate the quality of the learned node embeddings with node attributes. We perform node classification task on the evaluation datasets to compare the performance of our AC-HEN with the baselines. We first complete the missing attributes for target nodes by performing our model or baselines. Then, we send the whole graph into MAGNN [10] to obtain embeddings of nodes. Finally we feed the embeddings to a linear support vector machine (SVM) [42] classifier to get the classification results. The average Macro-F1 and Micro-F1 of all methods with variance on three datasets are summarized in Table 5.

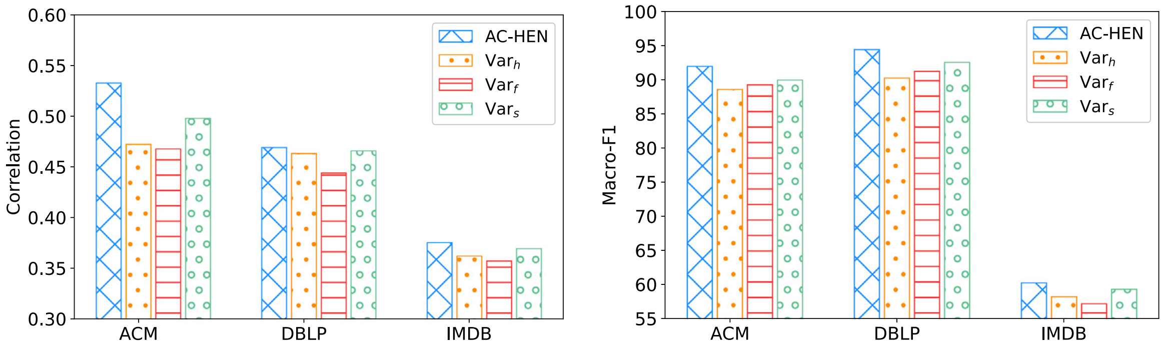
Since all the node representations for all methods are learned through MAGNN, there is no big difference in the results of node classification. However, as shown in Table 5, our AC-HEN achieves the best performance in terms of both Macro-F1 and Micro-F1 on all datasets. More specifically, after completing missing attributes for movie nodes, we perform node classification tasks on movies on IMDB data. We observe that AC-HEN is significantly better than state-of-the-art HGNN-AC by 1.67% and 1.91% improvement in terms of Macro-F1 and Micro-F1 on IMDB dataset. This is due to the fact that the proposed AC-HEN provides more comprehensive embeddings of target nodes than HGNN-AC in attribute completion. We also perform node classification on authors on ACM and DBLP after completing the missing attributes for author nodes. Our AC-HEN achieves average gains of 0.85% Macro-F1

Table 5
Results of node classification task in terms of Macro-F1 and Micro-F1 on three datasets.

Method	ACM		DBLP		IMDB	
	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
MC	89.37 ± 0.21	89.62 ± 0.22	91.08 ± 0.05	91.22 ± 0.08	57.49 ± 0.12	58.07 ± 0.21
EM	89.39 ± 0.10	89.68 ± 0.11	91.22 ± 0.21	91.97 ± 0.28	53.75 ± 0.06	54.58 ± 0.05
MLP	90.99 ± 0.54	91.03 ± 0.43	92.27 ± 0.12	92.82 ± 0.12	57.39 ± 0.61	57.94 ± 0.52
SVR	90.13 ± 0.10	90.25 ± 0.09	92.45 ± 0.31	93.03 ± 0.27	56.84 ± 0.59	57.47 ± 0.62
HGAT	91.19 ± 0.21	91.30 ± 0.17	92.52 ± 0.14	93.20 ± 0.11	57.96 ± 0.31	58.55 ± 0.37
AGCN	91.53 ± 0.17	91.62 ± 0.19	92.51 ± 0.09	93.17 ± 0.11	58.74 ± 0.45	58.86 ± 0.39
HGNN-AC	91.69 ± 0.11	91.75 ± 0.13	93.07 ± 0.25	93.68 ± 0.33	59.29 ± 0.18	59.74 ± 0.11
AC-HEN	**91.94** ± **0.21**	**92.31** ± **0.17**	**94.39** ± **0.22**	**94.99** ± **0.21**	**60.28** ± **0.15**	**60.88** ± **0.17**

Table 6
P-values of t-test for Macro-F1 and Micro-F1 on node classification.

Method	ACM		DBLP		IMDB	
	Macro-F1 *p*-value	Micro-F1 *p*-value	Macro-F1 *p*-value	Micro-F1 *p*-value	Macro-F1 *p*-value	Micro-F1 *p*-value
MC	$3.01 * e^{-12}$	$7.14 * e^{-16}$	$2.73 * e^{-16}$	$1.22 * e^{-14}$	$8.77 * e^{-15}$	$6.89 * e^{-14}$
EM	$4.42 * e^{-13}$	$3.76 * e^{-15}$	$3.51 * e^{-13}$	$3.78 * e^{-13}$	$6.38 * e^{-19}$	$1.01 * e^{-18}$
MLP	$1.23 * e^{-8}$	$2.75 * e^{-9}$	$8.37 * e^{-13}$	$9.21 * e^{-13}$	$4.64 * e^{-11}$	$1.26 * e^{-11}$
SVR	$7.23 * e^{-12}$	$3.73 * e^{-13}$	$4.53 * e^{-11}$	$1.14 * e^{-12}$	$7.91 * e^{-12}$	$9.82 * e^{-12}$
HGAT	$3.73 * e^{-8}$	$7.24 * e^{-10}$	$8.79 * e^{-13}$	$1.20 * e^{-12}$	$3.62 * e^{-13}$	$5.11 * e^{-12}$
AGCN	$7.25 * e^{-7}$	$1.17 * e^{-8}$	$2.21 * e^{-12}$	$1.67 * e^{-13}$	$3.90 * e^{-10}$	$1.21 * e^{-10}$
HGNN-AC	$3.32 * e^{-4}$	$4.49 * e^{-8}$	$1.55 * e^{-9}$	$1.97 * e^{-10}$	$4.50 * e^{-11}$	$5.46 * e^{-11}$



(a) Pearson correlation on attribute completion

(b) Macro-F1 on node classification

Fig. 2. Experimental results of ablation study for AC-HEN.

and 1.00% Micro-F1 compared to the best performed baseline (*i.e.*, HGNN-AC) on these two datasets. In these two academic networks, there is no direct connection between authors, and the direct connection of author nodes has only paper nodes. That is, the P-V and P-T edges on DBLP, and the P-P and P-S edges on ACM are not used as HGNN-AC only aggregates the attributes of first-order neighbors. This is why our model is better than all baselines, because our AC-HEN not only considers the similarity between author nodes through feature aggregation, but also fuses the connections of the above edges in ACM and DBLP through attention-based high-order aggregation.

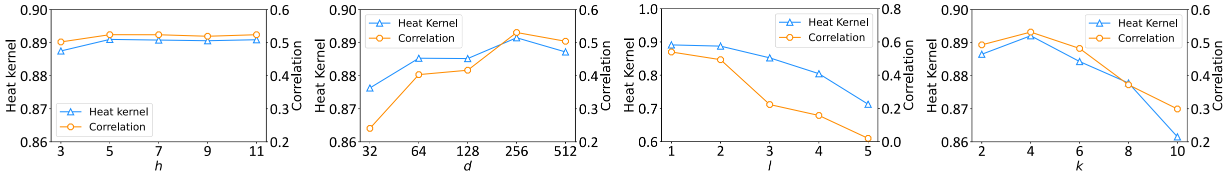
We also employ t-test to verify if there is a significant difference between the result of AC-HEN and those of state-of-the- art baselines on node classification task after attribute completion. The null hypothesis H_0 is that the mean result of AC-HEN is equal to that of the baselines (*i.e.*, MC, EM, MLP, SVR, HGAT, AGCN and HGNN-AC). The alternative hypothesis H_1 is that the mean result of AC-HEN is significantly better than that of the baselines. The *p-values* of t-test for Macro-F1 and Micro-F1 on node classification are summarized in Table 6.

As we can see, all *p-values* for both Macro-F1 and Micro-F1 *w.r.t.* all baselines on three datasets are less than 0.05, which means that we need to reject the null hypothesis and accept the alternative hypothesis. The results show that AC-HEN can significantly improve the performance of node classification in terms of Macro-F1 and Micro-F1 after attribute completion, which indicates the superiority of AC-HEN when competing with state-of-the- art baselines.

5.6. Ablation study

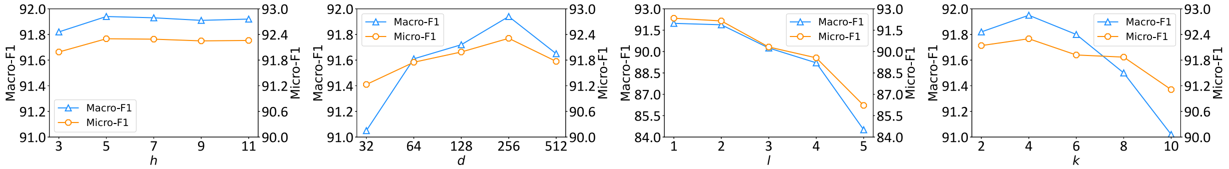
To verify each component of AC-HEN, we further conduct the ablation study. We compare our model with two carefully designed variations. Despite the changed part(s), all variations have the same framework structure and parameter settings. The performance of all variations in terms of Pearson correlation and Macro-F1 on three datasets are shown in Fig. 2.

- **Var**$_f$—This variation removes the feature aggregation module, which means that the multi-view embeddings only contain z_u^n and z_u^h.
- **Var**$_h$—In this variation, we remove the attention-based high-order aggregation module, meaning that the multi-view embeddings only include z_u^f and z_u^n.
- **Var**$_s$—In this variation, we employ self-attention instead of multi-head attention in attention-based high-order aggregation module, meaning that z_u^h is learned only by self-attention.



(a) #high-order neighbors (b) Dimension d (c) #layers l (d) #nearest neighbors k

Fig. 3. Sensitivity of AC-HEN *w.r.t.* parameters for attribute completion on ACM dataset.



(a) #high-order neighbors (b) Dimension d (c) #layers l (d) #nearest neighbors k

Fig. 4. Sensitivity of AC-HEN *w.r.t.* parameters for node classification on ACM dataset.

Effect of feature aggregation. The comparison between $\mathbf{Var}_f$ and AC-HEN highlights the effectiveness of the proposed feature aggregation for attribute completion in heterogeneous networks. From Fig. 2(a), we can see that $\mathbf{Var}_f$ performs the worst on all datasets. Specifically, AC-HEN significantly improves 11.5%, 6.11%, and 5.99% over $\mathbf{Var}_f$ in terms of *Pearson correlation* on ACM, DBLP, and IMDB datasets, respectively. This indicates that our proposed feature aggregation, considering the similarity between nodes in feature space, is essential for attribute completion in heterogeneous networks.

Effect of attention-based high-order aggregation. The comparison between $\mathbf{Var}_h$ and AC-HEN reflects the importance of attention-based high-order aggregation module for attribute completion. As shown in Fig. 2(b), $\mathbf{Var}_h$ produces worse results than AC-HEN on all datasets, reducing 4.41% performance in terms of Macro-F1 on DBLP dataset, which demonstrates the crucial role of our designed attention-based high-order aggregation module in capturing the effects of high-order neighbors on attribute completion for target nodes.

Effect of multi-head attention. The comparison between $\mathbf{Var}_s$ and AC-HEN demonstrates the necessary of employing multi-head attention in attention-based high-order aggregation module for attribute completion. As shown in Fig. 2, multi-head attention produces better results than self-attention on all datasets, improving 7.01% and 2.23% in terms of Pearson correlation and Macro-F1 on ACM dataset. Multi-head attention allows our AC-HEN to capture multiple dependencies from high-order neighbor nodes.

5.7. Parameter sensitivity

We now investigate the sensitivity of our model with respect to the important parameters, including the number of high-order neighbors h, the hidden embedding dimension d, the number of layers l, and the number of nearest neighbors k. All experiments are conducted on ACM dataset, and the results on attribute completion and node classification are depicted in Figs. 3 and 4, respectively.

The number of high-order neighbor nodes: we first test the effect of the number of high-order nodes h in attention-based high-order aggregation module. We can see that with the growth of the number of high-order nodes, the performance shows a trend of first rising and then remains stable. This is because high-order aggregation helps improve the performance of our model. However, in reality, due to the sparsity of the network, most nodes have not many high-order neighbors. When the number of walk samples is fixed, some high-order neighbors are sampled repeatedly. Therefore, when $h \geq 5$, no more new high-order neighbors are actually sampled.

The hidden embedding dimension: we next test the effect of the hidden embedding dimension d. From Fig. 3(b) and Fig. 4(b), we can find that the performance of our model shows a trend of first rising and then decreasing, as the dimension d increases. This is because too small embedding dimension cannot capture the all essential information of nodes to complete missing attributes, while too large embedding dimension may contain superfluous information which hurts the attribute completion performance.

The number of GCN layers: we also study the effect of the number of GCN layers l on model performance. We can observe that with the growth of the number of GCN layers, the performance of AC-HEN continuously decreases. This is consistent with our previous analysis. The reason is that graph convolutions can be regraded as feature smoothing in the graph. With the increment of the number of GCN layers, GCN can get high-order neighbors' information, but the obtained node embeddings tend to be over smooth. Therefore, we only use one-layer GCN to capture first-order neighbors' features in neighborhood aggregation, and design the attention-based high-order aggregation to capture high-order information to get the best overall performance.

The number of nearest neighbors k in feature space: we finally evaluate the effect of k-NNs in feature space on model performance. As shown in Fig. 3(d) and Fig. 4(d), the performance first increases and then gradually decreases, as the number of nearest neighbors k increases. This is intuitive because the proper similar neighbors in feature space are more conducive to attribute completing. Specifically, our model achieves the best performance on both attribute completion and node classification at the same time when k is set to 4.

5.8. Model efficiency analysis

We also compare the efficiency of our AC-HEN with other GNN baselines for attribute completion. We report the experiment results on three datasets in Table 7.

From the results in Table 7, we can see that our AC-HEN achieves the second best efficiency performance after HGAT. However, from above experimental results (Tables 3 and 5), AC-HEN is better than HGAT in both direct measurement for attribute completion and the downstream task on node classification. AC-HEN is faster than the best performed baseline HGNN-AC on all

Table 7
Run time comparison of HGNN-AC with GNN baselines.

Method	ACM	DBLP	IMDB
HGAT	2295.66	3869.32	1032.01
AGCN	8269.55	9133.40	3232.65
HGNN-AC	4528.36	7018.00	2203.32
AC-HEN	3155.47	6176.97	1644.61
Speedup*	**2.66**×	**1.48**×	**1.96**×

*Speedup of AC-HEN over AGCN.

datasets. This is because the pre-learning stage of HGNN-AC is time-cost, as it needs to sample meta-path from the datasets and employs existing heterogeneous graph embedding methods to get node embeddings based on the sampled meta-paths. More specifically, our AC-HEN achieves up to 2.66× speedup over state-of-the-art AGCN on ACM dataset. The main reason is that our AC-HEN adopts the idea of simplifying graph convolutional networks, that is, omitting non-linear activation function. Therefore, the training efficiency of AC-HEN can be improved significantly.

6. Conclusion

We present an effective attribute completion framework AC-HEN for heterogeneous networks. AC-HEN first uses feature aggregation and structure aggregation to obtain the multi-view embeddings that consider neighbor aggregation in both feature space and network structural space. Additionally, the obtained multi-view embeddings distinguish contributions of different neighbor nodes by conducting weighted aggregation. Finally, the multi-view embeddings are used to complete the missing attributes in a weak supervised learning manner. Experiments on three real-world heterogeneous networks show that AC-HEN achieves better performance compared to state-of-the-art baselines in both attribute completion and node classification tasks. For future work, we are interested in inducing an effective and efficient dynamic heterogeneous graph neural network to preserve both structural and temporal information for attribute completion.

CRediT authorship contribution statement

Kai Wang: Methodology, Software, Data curation, Visualization, Writing – original draft. **Yanwei Yu:** Conceptualization, Supervision, Writing – review & editing, Funding acquisition. **Chao Huang:** Formal analysis, Writing – review & editing. **Zhongying Zhao:** Writing – review & editing. **Junyu Dong:** Resources, Funding acquisition.

Declaration of competing interest

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

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China under grant Nos. 62176243, 61773331 and 41927805, and the National Key Research and Development Program of China under grant Nos. 2019YFC1509100 and 2018AAA0100602.

References

[1] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.

[2] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P.S. Yu, Heterogeneous graph attention network, in: Proceedings of the Web Conference 2019, ACM, 2019, pp. 2022–2032.

[3] C. Shi, X. Han, L. Song, X. Wang, S. Wang, J. Du, S.Y. Philip, Deep collaborative filtering with multi-aspect information in heterogeneous networks, IEEE Trans. Knowl. Data Eng. 33 (4) (2019) 1413–1425.

[4] P. Cui, X. Wang, J. Pei, W. Zhu, A survey on network embedding, IEEE Trans. Knowl. Data Eng. 31 (5) (2018) 833–852.

[5] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S.Y. Philip, A comprehensive survey on graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. 32 (1) (2020) 4–24.

[6] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, AI Open 1 (2020) 57–81.

[7] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: International Conference on Learning Representations, ICLR, 2017.

[8] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: International Conference on Learning Representations, ICLR, 2018.

[9] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems, NeurIPS, 2017, pp. 1024–1034.

[10] X. Fu, J. Zhang, Z. Meng, I. King, Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding, in: Proceedings of the Web Conference 2020, 2020, pp. 2331–2341.

[11] Z. Liu, C. Huang, Y. Yu, B. Fan, J. Dong, Fast attributed multiplex heterogeneous network embedding, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 995–1004.

[12] W. Chen, Y. Gu, Z. Ren, X. He, H. Xie, T. Guo, D. Yin, Y. Zhang, Semi-supervised user profiling with heterogeneous graph attention networks, in: IJCAI, 19, 2019, pp. 2116–2122.

[13] L. Wu, Y. Yang, K. Zhang, R. Hong, Y. Fu, M. Wang, Joint item recommendation and attribute inference: An adaptive graph convolutional network approach, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 679–688.

[14] D. Jin, C. Huo, C. Liang, L. Yang, Heterogeneous graph neural network via attribute completion, in: Proceedings of the Web Conference 2021, 2021, pp. 391–400.

[15] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 1067–1077.

[16] S. Cao, W. Lu, Q. Xu, Grarep: Learning graph representations with global structural information, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015, pp. 891–900.

[17] J. Qiu, Y. Dong, H. Ma, J. Li, C. Wang, K. Wang, J. Tang, Netsmf: Large-scale network embedding as sparse matrix factorization, in: The World Wide Web Conference, 2019, pp. 1509–1520.

[18] Y. Sun, S. Wang, T.-Y. Hsieh, X. Tang, V. Honavar, MEGAN: A generative adversarial network for multi-view network embedding, in: Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019.

[19] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: International Conference on Learning Representations, 2018.

[20] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, et al., Simplifying graph convolutional networks, in: International Conference on Machine Learning, ICML, 2019, pp. 6861–6871.

[21] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, J. Pei, Am-gcn: Adaptive multi-channel graph convolutional networks, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1243–1253.

[22] Y. Dong, N.V. Chawla, A. Swami, metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.

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

[24] Y. Sun, J. Han, X. Yan, P.S. Yu, T. Wu, Pathsim: Meta path-based top-k similarity search in heterogeneous information networks, Proc. VLDB Endow. 4 (11) (2011) 992–1003.

[25] X. Wang, Y. Zhang, C. Shi, Hyperbolic heterogeneous information network embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 5337–5344.

[26] C. Shi, B. Hu, W.X. Zhao, S.Y. Philip, Heterogeneous information network embedding for recommendation, IEEE Trans. Knowl. Data Eng. 31 (2) (2018) 357–370.
[27] A. Chaabane, G. Acs, M.A. Kaafar, et al., You are what you like! information leakage through users' interests, in: Proceedings of the 19th Annual Network & Distributed System Security Symposium, NDSS, Citeseer, 2012.
[28] N.Z. Gong, B. Liu, Attribute inference attacks in online social networks, ACM Trans. Priv. Secur. 21 (1) (2018) 1–30.
[29] Q. Fang, J. Sang, C. Xu, M.S. Hossain, Relational user attribute inference in social media, IEEE Trans. Multimed. 17 (7) (2015) 1031–1044.
[30] D. Rao, M. Paul, C. Fink, D. Yarowsky, T. Oates, G. Coppersmith, Hierarchical bayesian models for latent attribute detection in social media, in: Fifth International AAAI Conference on Weblogs and Social Media, 2011.
[31] D. Rao, D. Yarowsky, A. Shreevats, M. Gupta, Classifying latent user attributes in twitter, in: Proceedings of the 2nd International Workshop on Search and Mining User-Generated Contents, 2010, pp. 37–44.
[32] E. Zheleva, L. Getoor, To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles, in: Proceedings of the 18th International Conference on World Wide Web, 2009, pp. 531–540.
[33] N.Z. Gong, A. Talwalkar, L. Mackey, L. Huang, E.C.R. Shin, E. Stefanov, E. Shi, D. Song, Joint link prediction and attribute inference using a social-attribute network, ACM Trans. Intell. Syst. Technol. (TIST) 5 (2) (2014) 1–20.
[34] B. Sun, J. Du, T. Gao, Study on the improvement of K-nearest-neighbor algorithm, in: 2009 International Conference on Artificial Intelligence and Computational Intelligence, vol. 4, IEEE, 2009, pp. 390–393.
[35] S. Yun, M. Jeong, R. Kim, J. Kang, H.J. Kim, Graph transformer networks, Adv. Neural Inf. Process. Syst. 32 (2019) 11983–11993.
[36] Y. Wang, Z. Duan, B. Liao, F. Wu, Y. Zhuang, Heterogeneous attributed network embedding with graph convolutional networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, 33, (01) 2019, pp. 10061–10062.
[37] E.J. Candès, B. Recht, Exact matrix completion via convex optimization, Found. Comput. Math. 9 (6) (2009) 717–772.
[38] F. Dellaert, The expectation maximization algorithm, Technical Report, Georgia Institute of Technology, 2002.
[39] M. Minsky, S.A. Papert, Perceptrons: An Introduction To Computational Geometry, MIT Press, 2017.
[40] M. Awad, R. Khanna, Support vector regression, in: Efficient Learning Machines, Springer, 2015, pp. 67–80.
[41] J. Benesty, J. Chen, Y. Huang, I. Cohen, Pearson correlation coefficient, in: Noise Reduction in Speech Processing, Springer, 2009, pp. 1–4.
[42] J.A. Suykens, Support vector machines: a nonlinear modelling and control perspective, Eur. J. Control 7 (2–3) (2001) 311–327.