BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 02/06 pp233–241
Volume 4, Number 4, December 2021
DOI: 10.26599/BDMA.2021.9020008

Attention-Aware Heterogeneous Graph Neural Network

Jintao Zhang and Quan Xu*

Abstract: As a powerful tool for elucidating the embedding representation of graph-structured data, Graph Neural Networks (GNNs), which are a series of powerful tools built on homogeneous networks, have been widely used in various data mining tasks. It is a huge challenge to apply a GNN to an embedding Heterogeneous Information Network (HIN). The main reason for this challenge is that HINs contain many different types of nodes and different types of relationships between nodes. HIN contains rich semantic and structural information, which requires a specially designed graph neural network. However, the existing HIN-based graph neural network models rarely consider the interactive information hidden between the meta-paths of HIN in the poor embedding of nodes in the HIN. In this paper, we propose an Attention-aware Heterogeneous graph Neural Network (AHNN) model to effectively extract useful information from HIN and use it to learn the embedding representation of nodes. Specifically, we first use node-level attention to aggregate and update the embedding representation of nodes, and then concatenate the embedding representation of the nodes on different meta-paths. Finally, the semantic-level neural network is proposed to extract the feature interaction relationships on different meta-paths and learn the final embedding of nodes. Experimental results on three widely used datasets showed that the AHNN model could significantly outperform the state-of-the-art models.

Key words: Graph Neural Network (GNN); Heterogeneous Information Network (HIN); embedding

1 Introduction

Real data often contains structural information, for example, graph-structured data widely exist in the fields of chemistry[1, 2], physics[3, 4], and social science[5, 6]. As a powerful tool for embedding graph-structured data, Graph Neural Networks (GNNs) learn the embedding representation of nodes by aggregating and updating the neighboring information of nodes, and are widely used in molecules, social networks, and recommendation systems[7, 8]. Recently, based on the pioneering work in the development of the Graph Convolutional Network (GCN)[5], researchers have improved its performance by enhancing the aggregation update function and have proposed many variations[9–11]. Specifically, the introduction of the attention mechanism has greatly improved the performance of the GCN model; this network is named as Graph Attention neTwork (GAT)[9]. These studies demonstrate that GNNs have unique advantages in data mining tasks based on graph-structured data. Moreover, a GNN is built on a Homogeneous Network (HN).

Different from an HN, the rich data in the network often contain different types of nodes and relationships between these nodes, that naturally constitute a Heterogeneous Information Network (HIN)[12, 13]. An HIN contains not only structural information between nodes, but also semantic relationships between nodes[12, 14]. As shown in Fig. 1a, the IMDB dataset can be regarded as an HIN, which contains three types of nodes (i.e., users (U), movies (M), and directors (D)) and their relationships. Moreover, each

• Jintao Zhang is with the College of Sciences, Northeastern University, Shenyang 110004, China. E-mail: 20201825@stu.neu.edu.cn.
• Quan Xu is with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China. E-mail: quanxu@mail.neu.edu.cn.

* To whom correspondence should be addressed.

Manuscript received: 2021-02-21; revised: 2021-04-25; accepted: 2021-04-28

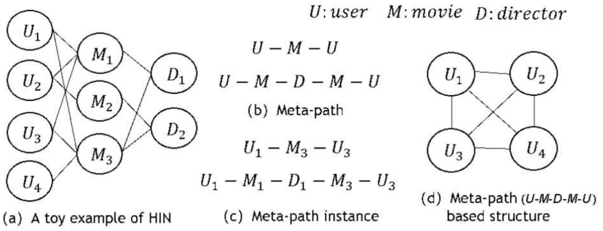

Fig. 1 A toy example of an HIN.

meta-path in an HIN often has different semantic information. For the two meta-paths given in Fig. 1b, User-Movie-User (U-M-U) indicates that two users have watched the same movie, and User-Movie-Director-Movie-User (U-M-D-M-U) represents that two users have watched a movie by the same director. Specifically, when we only consider the structural information of a meta-path, HIN becomes a homogeneous network, as demonstrated in Fig. 1d. Therefore, learning the embedding representation of nodes from HIN requires a specially designed graph neural network[14].

These challenges urge researchers to design a reasonable and effective GNN for embedding an HIN. Recently, some HIN-based GNNs have been proposed, most of which have been inspired by Heterogeneous graph Attention Networks (HANs)[14–16]. Specifically, HAN[14] first utilizes node-level attention (i.e., GAT) to fuse the neighboring information of nodes on each meta-path, and then uses semantic-level attention to integrate node embeddings on different meta-paths to learn the final embedding of nodes. As shown in Fig. 2, the semantic-level attention in HAN is essentially a weighted summation of the embedding representations of nodes on different meta-paths. However, there are two problems with semantic-level attention: One is that the corresponding dimensions of node embedding on different meta-paths may represent different aspects of information, the other is that semantic-level attention cannot easily capture the interactive information between node embeddings on different meta-paths. This makes it difficult for HAN to effectively learn high-quality embedded representations in HAN.

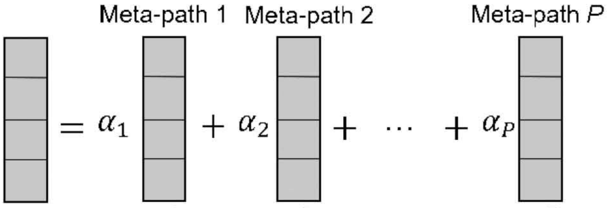

Fig. 2 Semantic-level attention of HAN.

In this paper, we propose an Attention-aware Heterogeneous graph Neural Network (AHNN) for embedding an HIN. Specifically, AHNN first utilizes node-level attention to learn the embedding of nodes on different meta-paths, and then concatenates the embeddings on different meta-paths and employs a semantic-level neural network to learn the final embedding representation of the node. Intuitively, a semantic-level neural network is better than the use of semantic attention for extracting feature interaction information between node embeddings on different meta-paths to learn high-quality embedding representations. The main contributions of this paper are summarized as follows.

- We propose a semantic-level neural network to extract feature interaction information hidden between node embeddings on different meta-paths. In this way, the comprehensive and subtle information between meta-paths can also be fully utilized.
- We propose an AHNN model to study node embedding in HIN, which includes node-level attention and semantic-level neural networks. As a novel method to embedding an HIN, AHNN can effectively extract the rich structural and semantic information in an HIN to learn high-quality embedding representations.
- We experimentally evaluate the performance of the AHNN model on three widely used datasets. The experimental results show that AHNN is superior to the state-of-the-art models.

2 Related Work

The main task of this paper is to utilize GNN and other related technologies to embedding an HIN. Therefore, this section mainly introduces two aspects, that is, GNN and HIN embeddings.

2.1 GNN

As a powerful tool for embedding graph structure data, GNN has been widely used in various data mining tasks, such as classification, clustering, and recommendation[7, 8]. GNNs mainly employ a graph convolution operator to aggregate the neighbouring information of the target node and update its embedding. Kipf and Welling[5] proposed a GCN to aggregate and update the embedding of nodes through the degree matrix between nodes. Recently, researchers have improved the performance of GCN by enhancing the graph convolution operator and have proposed many variants[9–11]. For example, Xu et al.[10] demonstrated the theoretical properties of GCN and its variants, and proposed a more powerful Graph Isomorphic Network (GIN). Moreover, Velickovie et al.[9] enhanced the performance of GCN by introducing flexible attention and multi-head mechanisms, namely GAT. Zhang and Xie[17] further analyzed GAT in theory and proposed Cardinality Preserved Attention (CPA) that can distinctguish structures.

Moreover, the aforementioned GNN often only uses simple weighted summation when aggregating neighboring information, this approach ignores the interactive information between neighbors. Therefore, researchers also have attempted to improve the performance of GNN by aggregating more neighboring information. For example, Zhu et al.[11] utilized the traditional linear aggregator and proposed a bilinear aggregator to capture the interactive information between neighbors. However, the above GNN models based on a homogeneous network cannot be easily applied to an HIN that contains rich structural and semantic information.

2.2 HIN embedding

The ideas of early embedding technology mainly originated from natural language processing and produced many classic methods, such as Deepwalk[18], and LINE[19]. As these methods mainly focus on homogeneous networks, researchers are attempting to improve them to adapt to HIN. For example, Metapath2vec[20] proposed an improved HIN by taking into account a meta-path based random walk and skip-gram. Shi et al.[12] proposed the HERec method to first generate random sequences through random walks based on the guidance of meta-paths, and then transform them into a homogeneous network by deleting different types of nodes before finally employing DeepWalk or other technologies to learn the embedding of nodes in an HIN.

Recently, with the rapid development of deep learning, some HIN embedding technologies based on GNN have emerged. For example, Hu et al.[21] introduced an attention mechanism that depends on the node type and edge type to avoid manual selection of meta-paths, thereby efficiently learning node embedding. MAGNN[22] was used to generate node embeddings by applying node content conversion, aggregation within meta-regions, and aggregation between meta-regions, followed by the application of a specific type of linear transformation to project heterogeneous node attributes into the same latent vector space, and then the use of an attention mechanism to apply intra-metadata aggregation for each set of metadata. In addition, most HIN embedding technologies based on GNN are inspired by HAN, which is the first attempt to introduce GAT into HIN to learn node embedding[14]. Specifically, HAN uses hierarchical attention to learn node embedding, that is, node-level attention and semantic-level attention. However, the attention at the semantic-level attention performs a weighted summation on the embeddings of nodes on different meta-paths, making it difficult to extract the feature interaction information hidden between different meta-paths.

3 Definition

The main purpose of this paper is to achieve embedding of an HIN, thus, in this section, we mainly introduce the related concepts of an HIN.

Definition 1 (HIN) Consider a graph $G = (V, E)$, where V and E represent the object set and link set, respectively. If two mapping functions, i.e., the mapping function $\delta(\cdot)$ denotes the node type, and the mapping function $\epsilon(\cdot)$ is the edge type, could map the nodes $v (\in V)$ and edges $e (\in E)$ to a specific type ($\delta(v) \in A$, $\epsilon(e) \in R$), where A represents the type of nodes, and R denotes the relation of the nodes, then this type of network can be regarded as an information network. If $|A| + |R| > 2$, then G is a heterogeneous information network.

Definition 2 (Meta-path) A meta-path is a path in the form $A_1(R_1)A_2 \cdots A_{l-1}(R_{l-1})A_l$ (abbreviated as $A_1 A_2 \cdots A_l$).

4 Proposed Model

In this section, we introduce in detail the proposed model, that is AHNN. The AHNN is mainly divided into two parts, as shown in Fig. 3. First, we use node-level attention to fuse the neighbor nodes information of target node under each meta-path. Next, the embedding of target node under each meta-path is concatenated, and a semantic-level neural network is utilized to learn the final embedding of the target node in HIN.

4.1 Node-level attention

Because each node's original feature is a vector and each node on each meta-path should have different features. In order to maintain the heterogeneity of the nodes, we utilize multiple type-specific transformation matrices to map the original feature of the each node to different meta-path feature spaces. Specifically, given the original feature $\boldsymbol{X}_u (\in R^{1\times F})$ of node u, we map it to the specific

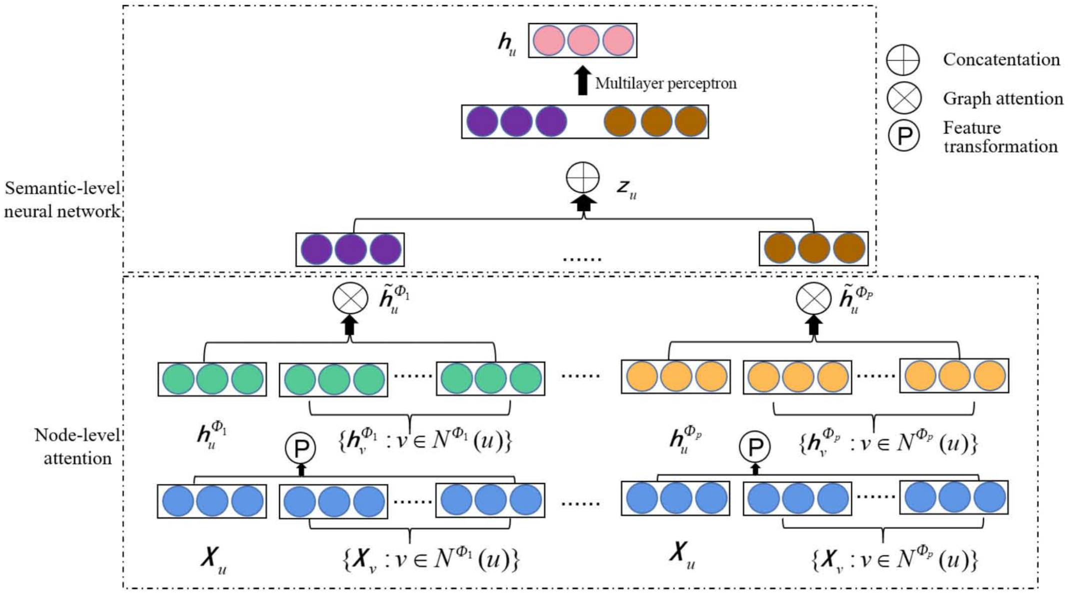

Fig. 3 Framework of the AHNN.

meta-path feature space through a transformation matrix $\boldsymbol{M}_{\Phi_i} (\in R^{F \times D})$,

$$\boldsymbol{h}_u^{\Phi_i} = \boldsymbol{X}_u \cdot \boldsymbol{M}_{\Phi_i} \tag{1}$$

where F is the dimension of the original feature, D is the transform dimension, Φ_i represents the i-th meta-path, and $\boldsymbol{h}_u^{\Phi_i} (\in R^{1 \times D})$ is the projected feature of node u in meta-path Φ_i. Obviously, $\boldsymbol{M}_{\Phi_i}$ can map all nodes to the same meta-path feature space to distinguish it from other meta-path feature spaces. Moreover, $\boldsymbol{M}_{\Phi_i}$ can make node-level attention aggregate different and rich information in each meta-path feature space.

After mapping the original features to different meta-path feature spaces, we can aggregate the neighboring information through node-level attention to update the node information under each meta-path. Specifically, given the meta-path Φ_i, and node u and its project feature $\boldsymbol{h}_u^{\Phi}$, the feature $\tilde{\boldsymbol{h}}_u^{\Phi}$ of node u is updated by aggregating the features $\{\boldsymbol{h}_i^{\Phi} : i = 1, \ldots, |N^{\Phi}(u)|\}$ of its neighbor nodes, where $N^{\Phi}(u)$ denotes the neighbors of node u, and $|N^{\Phi}(u)|$ denotes the number of neighbors. Note that $N^{\Phi}(u)$ also includes itself. The formula is as follows:

$$\tilde{\boldsymbol{h}}_u^{\Phi} = \sigma\Big(\sum_i \alpha_{ui}^{\Phi} \boldsymbol{h}_i\Big) \tag{2}$$

where σ represents the activation function, and the attention $\alpha_{ui}^{\Phi} (i = 1, \ldots, |N^{\Phi}(u)|)$ denotes the attention between nodes u and its neighbors under the meta-path Φ. The attention α_{ui}^{Φ} can be regarded as a variation of self-attention[5], and the relevant formula is as follows:

$$\alpha_{ui}^{\Phi} = \frac{\exp(\sigma(\boldsymbol{a}_{\Phi} \cdot [\boldsymbol{h}_u \| \boldsymbol{h}_i]^{\mathrm{T}}))}{\sum\limits_{j \in N^{\Phi}(u)} \exp(\sigma(\boldsymbol{a}_{\Phi} \cdot [\boldsymbol{h}_u \| \boldsymbol{h}_i]^{\mathrm{T}}))} \tag{3}$$

where $\boldsymbol{a}_{\Phi} (R^{1 \times 2D})$ represents the attention weight that is shared for all nodes under the meta-path Φ, and $\|$ represents the concatenate operation. Note that α_{ui}^{Φ} represents the importance of node i to node u, so α_{ui}^{Φ} is asymmetric, that is $\alpha_{ui}^{\Phi} \neq \alpha_{iu}^{\Phi}$, as shown in Eq. (3). In order to vividly show the process of aggregating neighbouring information through node-level attention, we provide a simple explanation in Fig. 4.

Because the HIN has the property of being scale-free, the variance of the graph data is very large. To overcome these limitations and stabilize the process of attention learning, we extend node-level attention to employ multi-head attention. Specifically, on a given meta-path, we repeatedly execute node-level attention K times and then concatenate all the results; the relevant Equation is as follows:

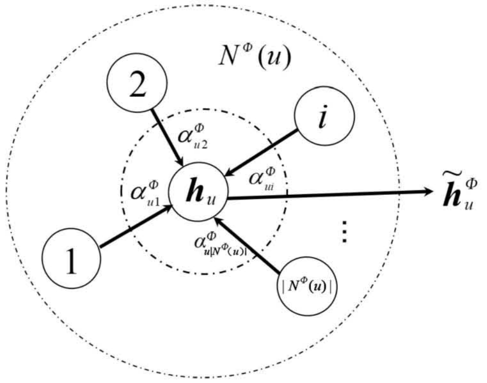

Fig. 4 Node-level attention. For a given meta-path Φ and target node u, calculate the attention between the target node and its neighbors, and then aggregate the feature information of the neighboring nodes through attention.

$$\tilde{\boldsymbol{h}}_u^{\Phi} = \|_{k=1}^{K} \, \sigma\Big(\sum_{i \in N^{\Phi}(u)} \alpha_{ui}^{\Phi,k} \cdot \boldsymbol{h}_i^{\Phi} \Big) \tag{4}$$

where $\alpha_{ui}^{\Phi,k}$ represents the attention between node i and node u under the meta-path Φ and the k-th multi-head, K represents the number of multi-head, and $\tilde{\boldsymbol{h}}_u^{\Phi}(\in R^{1\times KD})$ represents the embedding of node u after performing node-level attention. Similarly, we can perform the node-level attention operation for node u in each meta-path for the given meta-path set $\{\Phi_1, \Phi_2, \ldots, \Phi_P\}$. Afterward, we obtain the embedding representation $\tilde{\boldsymbol{h}}_u^{\Phi_1}, \tilde{\boldsymbol{h}}_u^{\Phi_2}, \ldots, \tilde{\boldsymbol{h}}_u^{\Phi_P}$ of node u under each meta-path. Moreover, all nodes adapt this strategy to aggregate and update embedding, thus, one can denote $\tilde{\boldsymbol{h}}^{\Phi_i} (i = 1, 2, \ldots, P)(\in R^{N\times KD})$ as a matrix composed of embeddings of all nodes on a given meta-path, where N represents the number of nodes.

4.2 Semantic-level neural network

Each meta-path in the heterogeneous information network contains rich structural and semantic information, and the useful information between them is different. In order to mine the interactive information hidden between them, under the assumption that the embedding representations on any two meta-paths are independent of each other, we perform a semantic-level neural network to learn the final embedding of nodes. Specifically, for a given node u, we first concatenate its embeddings on each meta-path after performing node-level attention, the relevant formula is as follows:

$$\boldsymbol{Z}_u = \|_{i=1}^{P} \, \tilde{\boldsymbol{h}}_u^{\Phi_i} \tag{5}$$

where P denotes the number of meta-path, and $\boldsymbol{Z}_u(\in R^{1\times PKD})$ represents the concatenation embedding. In fact, determining how to choose an effective meta-path in HIN and integrating it into the model is a huge challenge, however, Eq. (5) does not need to be used because it can flexibly concatenate the embeddings on multiple meta-paths. Thanks to the approximate theory of MultiLayer Perceptron (MLP), we can learn the final embedding representation of the node u by employing

$$\boldsymbol{h}_u = \sigma(\boldsymbol{Z}_u \cdot \boldsymbol{W} + \boldsymbol{b}) \tag{6}$$

where $\boldsymbol{h}_u(\in R^{1\times D_f})$ is the final embedding of node u, and $\boldsymbol{W}(\in R^{PKD\times D_f})$ and $\boldsymbol{b}$ $(\in R^{1\times D_f})$ represent the weight and bias of MLP, respectively, D_f represents the final embeding dimension. Note that $\boldsymbol{W}$ and $\boldsymbol{b}$ are shared by all nodes.

Finally, we consider training the model through the cross-entropy loss function for semi-supervised node classification, the relevant formula is as follows:

$$L = -\sum_{i \in \Omega_{\text{train}}} \ln(\boldsymbol{h}_i \cdot \boldsymbol{C}) \cdot \boldsymbol{Y}_i^{\mathrm{T}} \tag{7}$$

where Ω_{train} represents all nodes in the training set, $\boldsymbol{Y}_i(\in R^{1\times N_C})$ and $\boldsymbol{h}_i(\in R^{1\times D_f})$ are the ground-truth and final embedding of node i in the training set, N_C denotes the number of class, and $\boldsymbol{C}(\in R^{D_f\times N_C})$ is the parameter of the classifier.

5 Experiment

In this section, we present the results of multiple experiments conducted based on three widely-used datasets to verify the performance of our AHNN model.

5.1 Datasets

We chose three datasets for experimentation, namely, DBLP, IMDB, and ACM[14]. Among them, DBLP contains 4057 authors, 14 238 papers, 20 conferences, 8789 projects, and the relations between them. We divided the authors into four categories (i.e., database, data mining, machine learning, and information retrieval) according to the fields of the authors' submission conferences, and the feature vectors of the authors are represented by the bag-of-words of keywords in their papers. IMDB contains 5841 actors, 4780 movies, and 2269 directors. We divided the movies into three categories (i.e., action, comedy, and drama) according to their genres, and the features of movies are represented by the bag-of-words of the plot. ACM contains 3025 papers, 5835 authors, and 56 subjects. We divided the papers into three categories (i.e., database, wireless communication, and data mining) according to the fields of the papers to which they belong. The detailed information of the three datasets is shown in Table 1.

5.2 Baselines

Because the model proposed in this paper is an embedding model that utilizes GNN to learn node representation in HIN, we consider the classic representation model DeepWalk[18] (which is based on random walk), the meta-path-based random walk in HIN, named HERec[12], the GNN-based methods GCN[5] and GAT[9], and the graph neural learning model HAN[14] in HIN.

- DeepWalk[18]: A network embedding method based on performing a random walk in homogeneous graphs. We ignore the heterogeneity of nodes by performing random walks in the entire heterogeneous graph.
- HERec[12]: A heterogeneous information network

Table 1 Statistical information of three datasets.

Dataset	Relation (A-B)	# A	# B	# A-B	#Feature	#Training	#Validation	#Test	Meta-path
DBLP	Paper-Author	14 328	4057	19 645	334	800	400	2857	APA
	Paper-Conference	14 328	20	14 328					APCPA
	Paper-Term	14 327	8789	88 420					APTPA
IMDB	Movie-Actor	4780	5841	14 340	1232	300	300	2687	MAM
	Movie-Director	4780	2269	4780					MDM
ACM	Paper-Author	3025	5835	9744	1830	600	300	2125	MAM
	Paper-Subject	3025	56	3025					MDM

embedding model that performs random walk conducted by meta-path, which only preserves the nodes with the same type. Note that we perform HERec in each meta-path and report the best performance.

- GCN[5]: A homogeneous network embedding model that aggregates neighboring information by graph convolution. Here, we perform GCN in each meta-path, this approach only preserves nodes of the same type.
- GAT[9]: A homogeneous network embedding model that aggregates neighboring information by graph attention. Here, we perform GAT in each meta-path, this approach only preserves nodes of the same type.
- HAN[14]: A heterogeneous network embedding model by performing hierarchical attention (i.e., node-level attention and semantic-level attention).

5.3 Implementation details

For the proposed AHNN model, we set the head to 2, the hidden dimensions (i.e., dimension in node-level attention) to 64 (that is, $K = 2$ and $D = 32$, see Section 5.6 in detail). Following Ref. [14], we set the final embedding dimensions to 64 (i.e., $D_f = 64$) for all models, and set the multi-head and hidden dimensions to 8 and the number of layers to 1 for GCN, GAT, and HAN (i.e., $K = 8$ and $D = 8$). Moreover, we set the learning rate of the GCN, GAT, HAN, and AHNN models to 0.005, the dropout to 0.6, and the regularization parameters to 0.0001. For GCN, GAT, HAN, and AHNN, we set the patience to 100 so that the training will not be stopped until the validation loss does not decrease for 100 consecutive epochs . Furthermore, following Refs. [12] and [14], for DeepWalk and HERec, we set the window size to 5, the walk length to 100, the walks per node to 40, and the number of negative samples to 5.

5.4 Classification

For all models, when we learn the final embedding representations of the nodes, a criterion for evaluating its quality is its performance in downstream tasks. In this section, following Ref. [14], we utilize the classic KNN classifier with $k = 5$ to classify the embedding of all models and report Macro-F1 and Micro-F1, as shown in Table 2. In order to make a fair comparison, we repeated the classification of all models ten times and then reported the average value. From Table 2, we can see that our AHNN model has the best performance in most cases. Specifically, the performance of AHNN is the best on the ACM dataset compared to other models considered, and it is comparable to HAN and better than other models on IMDB and DBLP. The main reason for this result is that the number of nodes in the ACM is small and the original feature dimension of the nodes is larger. For IMDB and DBLP, they have a large number of nodes but a small original feature dimension of the nodes, making it difficult for a semantic-level neural network of AHNN to extract abundant feature interaction information hidden between meta-paths. The classification results also show that the AHNN model can extract more useful information from a larger number of node features to improve the quality of embedding.

5.5 Clustering

Similar to the previous section, following Ref. [14], in this section, we use KMeans to cluster the nodes by embedding. The clustering results can also be used as a criterion for judging the quality of embeddings. Because KMeans is an unsupervised process and is affected by the initial center, here, we repeated the experiment 10 times and reported the average result. It can be seen from Table 3 that the performance of AHNN on ACM is greatly improved compared to the performance of other models. AHNN has a slight performance improvement over HAN on DBLP, and HAN is better than other models. For IMDB, the performance of AHNN is poorer than HAN and better than other models. The main reason for this result is that the ACM dataset has fewer nodes and more features. Relatively speaking, IMDB has many nodes, few features, and a strong correlation between categories. The clustering results also show that the AHNN model is suitable for datasets with strong

Table 2 Results of the classification.

(%)

Dataset	Metric	Training	Deepwork	HERec	GCN	GAT	HAN	AHNN
ACM	Macro-F1	20	77.25	66.17	85.22	86.70	87.35	**90.47**
		40	80.47	70.89	86.11	87.11	87.73	**90.01**
		60	82.55	72.38	87.44	88.79	88.82	**89.11**
		80	84.17	73.92	88.97	89.82	89.80	**90.37**
	Micro-F1	20	76.92	66.03	85.14	86.70	87.35	**90.42**
		40	79.99	70.73	85.88	87.02	87.59	**89.79**
		60	82.11	72.24	87.30	88.73	88.69	**88.85**
		80	83.88	73.84	88.90	89.75	89.75	**90.18**
DBLP	Macro-F1	20	77.43	91.68	89.87	92.42	92.44	**92.68**
		40	81.02	92.16	90.24	92.60	**92.78**	92.33
		60	83.67	92.80	91.27	93.08	93.66	**93.66**
		80	84.81	92.34	92.99	94.69	95.33	**95.72**
	Micro-F1	20	79.37	92.69	90.82	93.32	93.35	**93.69**
		40	82.73	93.18	91.26	93.60	**93.73**	93.29
		60	85.27	93.70	92.49	93.99	**94.52**	94.43
		80	86.26	93.27	93.94	95.27	95.89	**96.23**
IMDB	Macro-F1	20	40.72	41.65	38.26	43.27	**45.88**	44.93
		40	45.19	43.86	38.36	44.28	46.40	**47.39**
		60	48.13	46.27	39.35	43.59	48.81	**50.50**
		80	50.35	47.64	41.06	44.73	49.06	**52.33**
	Micro-F1	20	46.38	45.81	41.68	46.30	**49.59**	48.81
		40	49.99	47.59	41.27	47.04	49.73	**50.92**
		60	52.21	49.88	42.13	46.31	51.84	**53.37**
		80	54.33	50.99	43.45	46.45	52.35	**54.60**

Table 3 Results of the clustering.

(%)

Dataset	Metric	Deepwork	HERec	GCN	GAT	HAN	AHNN
ACM	NMI	41.61	40.70	55.70	55.10	56.69	**62.73**
	ARI	35.10	37.13	59.30	58.75	60.50	**66.39**
DBLP	NMI	76.53	76.73	76.16	77.82	78.20	**78.58**
	ARI	81.35	80.98	81.50	82.81	84.29	**84.64**
IMDB	NMI	1.45	1.20	3.47	6.56	**9.49**	8.92
	ARI	2.15	1.65	3.81	8.19	**8.24**	7.53

independence between classes.

5.6 Parametric analysis

In this section, we mainly analyze the impact of the number of attention heads K, the size of hidden dimensions D, and the dimension of the final embedding D_f on the AHNN model. First, we fix the hidden dimensions to 16 and the final embedding dimensions to 64, and then we test the ARI and NMI of the AHNN model under different numbers of attention heads. As shown in Fig. 5, the optimal number of attention heads is 2. Next, we set the number of attention heads to 2, and the final embedding dimensions to 64, and then test the performance of the AHNN model under different hidden dimensions. As shown in Fig. 6, the optimal hidden dimensions is 64. Finally, we fix the attention heads to 2 and the hidden dimensions to 64, and then test the performance of the AHNN model in different final embedding dimensions, as shown in Fig. 7. After the experimental analysis of the above three parameters, although ARI and NMI fluctuate slightly, we can conclude that too large or too small parameter values will lead to poor performance of the AHNN model.

6 Conclusion

In this article, we proposed AHNN. Specifically, the AHNN model first aggregates the neighboring information of each node under different meta-paths through node-level attention, and then concatenates the embeddings on different meta-paths before employing a semantic-level neural network to mine the feature interaction information hidden between different meta-paths and different dimensions to learn the final embeddings of nodes. Moreover, the AHNN model can flexibly model multiple meta-paths. The extensive experimental results on classification and clustering tasks all demonstrated that the AHNN model could solve

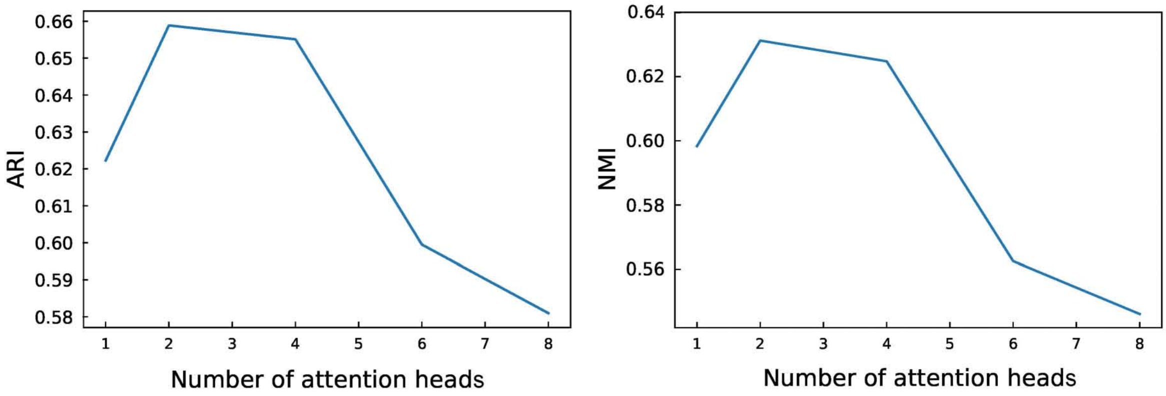

Fig. 5 Impact of the number of attention heads.

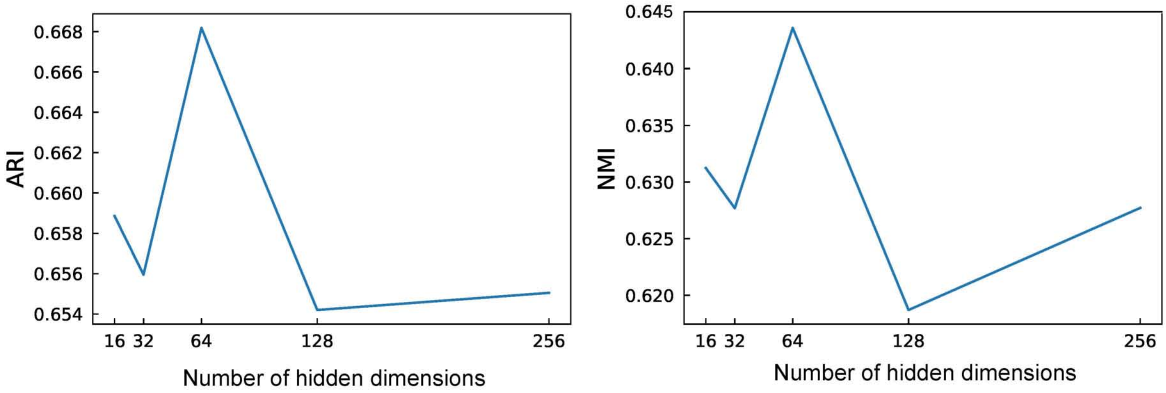

Fig. 6 Impact of the number of hidden dimensions.

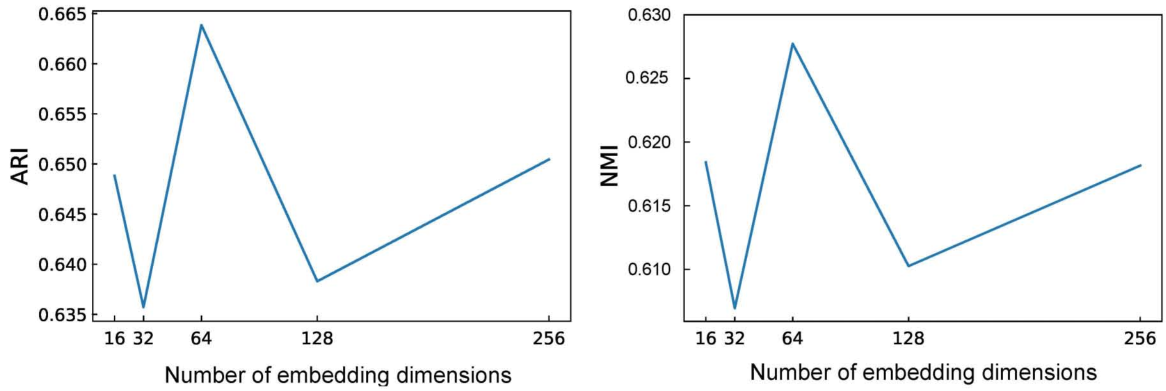

Fig. 7 Impact of the number of embedding dimensions.

the problem effectively. Moreover, the experimental results also showed that the AHNN model is suitable for embedding HIN with a large number of original node features and strong node category independence. Furthermore, AHNN provides an alternative method for embedding an HIN, and its performance depends on the characteristics of the dataset. In future work, we will enhance AHNN to further improve its performance.

Acknowledgment

This work was supported by the Key Scientific Guiding Project for the Central Universities Research Funds (No. N2008005), the Major Science and Technology Project of Liaoning Province of China (No. 2020JH1/10100008), and the National Key Research and Development Program of China (No. 2018YFB1701104).

References

[1] D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, Convolutional networks on graphs for learning molecular fingerprints, in *Proc.* 28th *Int. Conf. Neural Information Processing Systems*, Montreal, Canada, 2015,

pp. 2224–2232.

[2] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, Neural message passing for quantum chemistry, in *Proc. 34th Int. Conf. Machine Learning*, Sydney, Australia, 2017, pp. 1263–1277.

[3] M. D. Cranmer, R. Xu, P. Battaglia, and S. Ho, Learning symbolic physics with graph networks, arXiv preprint arXiv: 1909.05862, 2019.

[4] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia, Graph networks as learnable physics engines for inference and control, in *Proc. 35th Int. Conf. Machine Learning*, Stockholm, Sweden, 2018, pp. 4470–4479.

[5] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, in *Proc. 5th Int. Conf. Learning Representations*, Toulon, France, 2017, https://openreview.net/forum?id=SJU4ayYgl.

[6] X. N. He, K. Deng, X. Wang, Y. Li, Y. D. Zhang, and M. Wang, LightGCN: Simplifying and powering graph convolution network for recommendation, in *Proc. 43rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, doi: 10.1145/3397271.3401063.

[7] V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Breson, Benchmarking graph neural networks, arXiv preprint arXiv: 2003.00982, 2020.

[8] Z. W. Zhang, P. Cui, and W. W. Zhu, Deep learning on graphs: A survey, *IEEE Transactions on Knowledge and Data Engineering*, doi: 10.1109/TKDE.2020.2981333.

[9] P. Velickovie, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, Graph attention networks, in *Proc. 6th Int. Conf. Learning Representations*, Vancouver, Canada, 2018, https://openreview.net/forum?id=rJXMpikCZ.

[10] K. Xu, W. H. Hu, J. Leskovec, and S. Jegelka, How powerful are graph neural networks? in *Proc. 7th Int. Conf. Learning Representations*, New Orleans, LA, USA, 2019, https://openreview.net/forum?id=ryGs6iA5Km.

[11] H. M. Zhu, F. L. Feng, X. N. He, X. Wang, Y. Li, K. Zheng, and Y. D. Zhang, Bilinear graph neural network with neighbor interactions, in *Proc. 29th Int. Joint Conf. Artificial Intelligence*, Yokohama, Japan, 2020, pp. 1452–1458.

[12] C. Shi, B. B. Hu, W. X. Zhao, and P. S. Yu, Heterogeneous information network embedding for recommendation, *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 357–370, 2019.

[13] X. Wang, M. Q. Zhu, D. Y. Bo, P. Cui, C. Shi, and J. Pei, AM-GCN: Adaptive multi-channel graph convolutional networks, in *Proc. 26th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, San Diego, CA, USA, 2020, pp. 1243–1253.

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

[15] W. J. Chen, Y. L. Gu, Z. C. Ren, X. N. He, H. T. Xie, T. Guo, D. W. Yin, and Y. D. Zhang, Semi-supervised user profiling with heterogeneous graph attention networks, in *Proc. 28th Int. Joint Conf. Artificial Intelligence*, Macao, China, 2019, pp. 2116–2122.

[16] H. T. Hong, H. T. Guo, Y. C. Lin, X. Q. Yang, Z. Li, and J. P. Ye, An attention-based graph neural network for heterogeneous structural learning, in *Proc. 34th AAAI Conf. Artificial Intelligence*, New York, NY, USA, 2020, pp. 4132–4139.

[17] S. Zhang and L. Xie, Improving attention mechanism in graph neural networks via cardinality preservation, in *Proc. 29th Int. Joint Conf. Artificial Intelligence*, New York, NY, USA, 2020, pp. 1395–1402.

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

[19] J. Tang, M. Qu, M. Z. Wang, M. Zhang, J. Yan, and Q. Z. Mei, LINE: Large-scale information network embedding, in *Proc. 24th Int. Conf. World Wide Web*, Florence, Italy, 2015, pp. 1067–1077.

[20] Y. X. Dong, N. V. Chawla, and A. Swami, Metapath2vec: Scalable representation learning for heterogeneous networks, in *Proc. 23rd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Halifax, Canada, 2017, pp. 135–144.

[21] Z. N. Hu, Y. X. Dong, K. S. Wang, and Y. Z. Sun, Heterogeneous graph transformer, in *Proc. Web Conf.*, Taipei, China, 2020, pp. 2704–2710.

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

Quan Xu received the PhD degree from University of Lille, France in 2011. He is an associate professor at the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, China. His research interests include industrial Internet, cloud services, and big data analytics and visualization.

Jintao Zhang is an undergraduate student at the College of Sciences, Northeastern University, China. His current research interests include big data analytics and recommender systems.