HNECV: Heterogeneous Network Embedding via Cloud model and Variational inference

Ming Yuan[1], ✉Qun Liu[1], ✉Guoyin Wang[1], and Yike Guo[2,3]

[1] Chongqing Key Laboratory of Computational Intelligence, Chongqing University of Posts and Telecommunications, ChongQing, China
675260709@qq.com,{liuqun,wanggy}@cqupt.edu.cn
[2] Hong Kong Baptist University, HongKong, China
[3] Imperial College London, London, UK
yikeguo@hkbu.edu.hk

Abstract. Deep learning has been successfully used in heterogeneous network embedding. Although it shows excellent performance on preserving the structure and semantic characteristics of network while a large scale of training data is provided, it is still challenging to model complex structured representations that effectively perform on diverse network tasks. In this work, a new heterogeneous network embedding learning method is presented based on cloud model and variational inference, called HNECV. The model uses meta-path random walks to obtain structural information of original network which can capture abundant semantics of networks from different views. In addition, a novel framework is put forward to build an excellent embedding. We employ the forward cloud transformation algorithm to improve the sampling method of the variational autoencoder in its hidden space, and then a self-supervised learning module is constructed to guide the cluster of node vectors in the hidden space of variational autoencoder. Experimental results indicate that the proposed model can achieve better performance than those of state-of-the-art algorithms. Furthermore, HNECV shows better robustness and steadiness on different network tasks when different ratio of edges are disconnected at training.

Keywords: heterogeneous network · representation learning · variational autoencoder · cloud model · meta-path.

1 Introduction

Network embedding can map the nodes in the network to low-dimensional space and capture the structural information of the network. Recent work has confirmed that this kind of node representation can bring significant performance improvements to tasks such as link prediction [1,2,3], node classification [4,5,6], and node clustering [7,8,9].

However, the traditional network embedding method only focuses on homogeneous network embedding, and the network only owns the same type of nodes and links. In real scenarios, the types of nodes and links in the network are

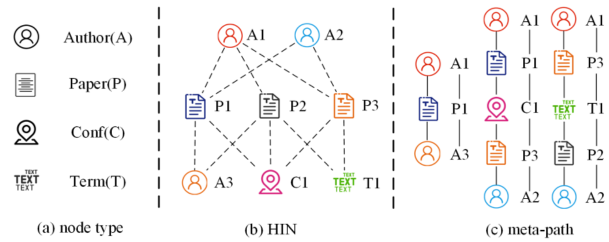

Fig. 1. Example of DBLP heterogeneous information network.

often different, and they construct a heterogeneous information network(HIN). Compared with homogeneous networks, HIN has more complex network relationships and contain rich semantic information. As shown in Fig. 1, the DBLP academic citation network contains four different types of nodes, such as Author(A), Paper(P), Conferences(C) and Term(T), and three different types of link relations, that is Paper-Author(PA), Paper-Conference(PC), and Paper-Term(PT). Obviously, the use of homogeneous network embedding in such a network will inevitably lead to a decline in embedding performance.

Therefore, in order to overcome the challenges brought by HIN, Some methods have been studied from the perspective of heterogeneous neighbors and different links, such as Metapath2vec [10] , HIN2vec [11], HERec [12]. In addition, some researchers use the decomposition method to simplify the HIN [13,14] , which eases the difficulty in the modeling process to a certain extent. Most of the above methods make preprocessing designs for HIN, and then learn the representation of nodes in combination with shallow neural networks.

SHINE [15] uses multiple deep autoencoders to map user information. MCRec [16] designed a co-attention mechanism to learn the importance of nodes and meta-paths. HAN [17] is the first expansion of graph neural network on heterogeneous graphs. HetGAN [18] tries the structure of GAN to learn embedding. RHINE [19] maps different relationships between nodes into different spaces. Besides, there are some works that explore how to model attributes of HIN [20,21,22] or use the GCNs method to aggregate the features of nodes [23,24,25] to learn the latent semantics and high-order neighbor features of the network.

Most of the above methods consider the characteristics of the network structure and semantics, and rarely consider the original real distribution information of the HIN. For example, although some nodes in the real network do not have direct links, they may still have a high similarity. Based on VAE, we can improve the quality of node representation by inducing the distribution of latent space with some restrictions rather than by relying on the structure of networks. In fact, if the distribution information of network data cannot be used, the low-dimensional embedding of nodes learned by the model may not conform to the real network situation, and it will also affect the accuracy of subsequent tasks. To address the above problems, the main contributions of this work are summarized as following:

- We provide a key insight to learn heterogeneous information network embeddings based on variational autoencoders. In order to make better use of the real distribution information of network data, it provides a feasible idea and promotes the application of HIN embedding.
- In this framework, cloud model, which is also known as a "recognition" model, is introduced to approximate the true posterior. And then a self supervised learning module is constructed to guide the cluster of similar nodes in the hidden space. Through the above joint optimization, HNECV closely integrates inference and clustering to learn high quality heterogeneous network embedding.
- We have conducted comprehensive experiments on three real datasets, and the experimental results show that the method in this paper is more superiord and robustness than state-of-the-art algorithms.

2 Preliminaries

Definition 1. *Heterogeneous information network(HIN) [26]. A heterogeneous information network is defined as* $G = (V, E)$*, including a set of nodes* V *and a set of links* E*. The mapping functions of node type and link type in the network are* $\varphi : V \to \mathcal{A}$ *and* $\psi : E \to \mathcal{R}$ *respectively.* $\mathcal{A}$ *and* $\mathcal{R}$ *represent the pre-defined node type and link type, where* $|\mathcal{A}| + |\mathcal{R}| > 2$*. The network schema is defined as* $S = (\mathcal{A}, \mathcal{R})$*, which is the basic prototype of HIN, and can represent a graph with node type* $\mathcal{A}$ *and link type* $\mathcal{R}$*.*

Definition 2. *Meta-path [27]. The meta-path* P *is a path instance defined on the network schema* $S = (\mathcal{A}, \mathcal{R})$*, which is denoted as* $\mathcal{A}_1 \xrightarrow{\mathcal{R}_1} \mathcal{A}_2 \xrightarrow{\mathcal{R}_2} \ldots \xrightarrow{\mathcal{R}_l} \mathcal{A}_{l+1}$*.* $\mathcal{R} = \mathcal{R}_1 \circ \mathcal{R}_2 \circ \ldots \circ \mathcal{R}_l$ *describes the composite relationship between* $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{l+1}$*, and* $\circ$ *represents the composite operator on the relationship.*

Fig. 1(c) shows the meta-path relationship in the DBLP network, each meta-path represents a different semantic. For example, the meta-path APA indicates the co-author relationship between two authors, APCPA indicates the co-conferences relationship, and APTPA indicates the co-term relationship.

3 The Proposed Model

In this section, our model(HNECV) is introduced in detail. The overall framework of the model is shown in Fig. 2. It is mainly composed of three parts: fusion heterogeneous graph structure, forward cloud inference, and self-supervised learning module. Our dataset and codes can be available at website[4].

[4] https://github.com/benym/HNECV

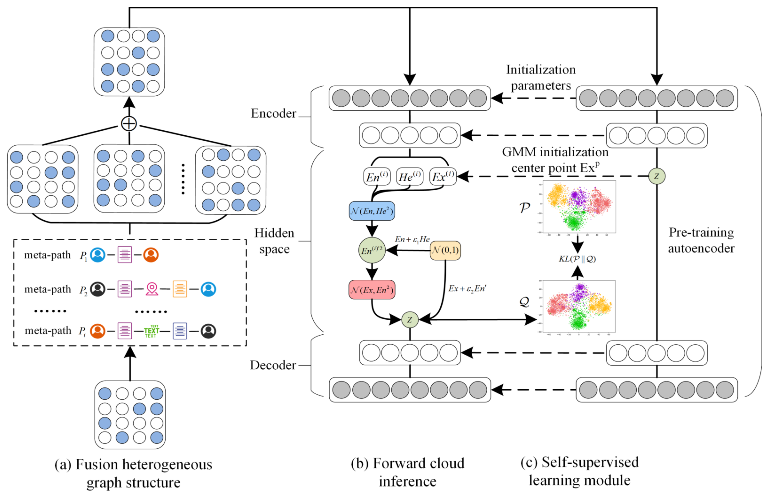

Fig. 2. The framework of HNECV.

3.1 Fusion Heterogeneous Graph Structure

In heterogeneous information networks, meta-paths can describe different semantic information, and the node sequence generated by random walks of meta-paths can well retain the structure and semantics of HIN. As shown in Fig. 2(a), we use multiple meta-paths $P_1, P_2, \ldots, P_i$ to guide the random walk to obtain the multi-view information of the original HIN. Random walks under different meta-paths can obtain different node sequences $H_1, H_2, \ldots, H_i$, which reflect the structural characteristics of HIN in a specific semantic environment. Hence, the multiple adjacency matrices $G_1, G_2, \ldots, G_i$ are reconstructed according to the node sequences of different meta-paths. Furthermore, we merge these matrices as follows:

$$G_{in} = G_1 \oplus G_2, \ldots \oplus G_i \tag{1}$$

where $\oplus$ is the element-wise addition, and G_{in} is the matrix after fusion.

3.2 Forward Cloud Inference

Generally in variational autoencoder(VAE) [28], the input of the decoder is sampled from the latent space depending on Gaussian distribution. But the reconstructed input data usually lose a lot of information. In order to solve this problem, we extend the sampling process in VAE by introducing the cloud model to obtain the embeddings of nodes. Cloud models algorithm has been widely used in various research fields [29,30]. It can realize the bidirectional transformations between concepts and data based on probability statistics and fuzzy set theory. The probability distribution in the cloud model is also called the Gaussian cloud distribution, which can be described by Expected Value Ex, Entropy En, and Hyper Entropy He. Compared with the Gaussian distribution, the Gaussian cloud distribution describes the distribution of data more accurately.

As same as the structure of the VAE, our model uses MLP as the encoder and decoder. We use the fused adjacency matrix G_{in} as the input of the encoder, suppose G_{in} contains N nodes, the representation of each hidden layer is as following:

$$h_N^c = f\left(W^c h_N^{(c-1)} + b^c\right), c = 1, 2, \ldots, C \tag{2}$$

where $f(\cdot)$ is the Relu function, W^c and b^c are the weight and bias parameters of the model in the c-th layer, C is the number of layers of the encoder or decoder, and input data x_n is denoted as h_N^0.

Here, the latent variable Z in the VAE is no longer described by the mean and variance, but is represented by the Gaussian cloud parameters Ex, En, He. The Ex and En can be compared to the mean and variance of the original Gaussian distribution, He can reflect the thickness of the Gaussian cloud, it can be used to adjust the thickness of Gaussian cloud by sampling more data from the distribution of space. Their formal descriptions in our model are given as follows:

$$Ex = W^c h_N^c + b^c \tag{3}$$

$$En = W^c h_N^c + b^c \tag{4}$$

$$He = W^c h_N^c + b^c \tag{5}$$

Inspired by the forward cloud transformation(FCT) algorithm [30], HNECV leverages two sampling steps to obtain the latent variable Z, which expands the sampling space of the original VAE. Meanwhile, due to the sampling operation cannot be derivable, we employ reparameterization trick, and take the sampling results into the training process to ensure that the model can be optimized. The forward cloud inference of HNECV is shown in Fig. 2(b), and this process can be defined as follows:

$$\varepsilon_1 \sim \mathcal{N}(0,1) = En' \sim \mathcal{N}\left(En, He^2\right) \tag{6}$$

$$En' = \varepsilon_1 \odot He + En \tag{7}$$

$$\varepsilon_2 \sim \mathcal{N}(0,1) = Z \sim \mathcal{N}\left(Ex, En'^2\right) \tag{8}$$

$$Z = \varepsilon_2 \odot En' + Ex \tag{9}$$

The latent variable Z can be obtained by combining Equations (7) and (9):

$$Z = Ex + \varepsilon_1 \odot En + \varepsilon_1 \odot \varepsilon_2 \odot He \tag{10}$$

where ε_1 and ε_2 represent noise variables sampled from the standard Gaussian distribution, and $\odot$ denotes Hadamard product.

Assuming that the latent variable Z obeys the standard Gaussian distribution $\mathcal{N}(0,1)$, the distance between the two distributions can be calculated by KL divergence as follows:

$$\begin{aligned}&D_{KL}\left(\mathcal{N}\left(Ex, En^2\right) \| \mathcal{N}(0,1)\right) = \\ &\mathcal{L}_{KL} = \frac{1}{2}\sum_{d=1}^{D}\left[1 + \ln\left(\left(En_d\right)^2\right) - \left(Ex_d\right)^2 - \left(En_d\right)^2\right]\end{aligned} \tag{11}$$

where D is the number of dimensions of the hidden space, Ex_d and En_d are the Ex and En of the d-dimension respectively.

After obtaining the latent variable Z, HNECV can generate an adjacency matrix containing N nodes through the following equations:

$$h_N^{(c-1)} = f\left(W^c Z + b^c\right) \tag{12}$$

$$h_N^c = f\left(W^c h_N^{(c-1)} + b^c\right) \tag{13}$$

$$\hat{x} = W^c h_N^c + b^c \tag{14}$$

Then, we can calculate the distance between ground truth x and the generation $\hat{x}$ to get the reconstruction error of the model. Due to the sparsity of network data, we impose greater penalties on more meaningful non-zero elements. The reconstruction objective function of HNECV is shown as follows:

$$\mathcal{L}_{rec} = \sum_{n=1}^{|N|} \left\|(\hat{x}_n - x_n) \odot B_n\right\|_2^2 \tag{15}$$

where $B_i = \{B_{i,j}\}_{j=1}^{|N|}$ is a penalty item. If the i-th row and j-th column element of the adjacency matrix G_{in} is 0, B=1, otherwise B¿1.

3.3 Self-supervised Learning Module

Aforementioned HNECV can represent nodes as dense vectors. In fact, different nodes have different categories in the networks. In order to achieve the quality node vector in the latent space, it is important to enhance the cohesiveness in the hidden space. Therefore, as shown in Fig. 2(c), we designed a self-supervised learning module to integrate the clustering learning process naturally into the model learning process without any supervision information.

In particular, for the low-dimensional representation Z_i of the node object x_i, the certainty degree that it belongs to the k-th category in the hidden space can be calculated by FCT [30]:

$$\eta_{ik} = \mathrm{e}^{-\frac{\left(Z_i - Ex_k^p\right)^2}{2En_k'^2}} \tag{16}$$

where Ex_k^p is the expectation of the k-th Gaussian distribution, which is initialized with the mean μ learned by GMM in our model, it will be used in the self-supervised learning. En_k' is the random number sampled from the k-th Gaussian distribution initially. In order to calculate the probability that the vector representation Z_i belongs to each category, the Equation (16) can be rewritten as the Equation (17), which is used to measure the similarity between the node vector representing Z_i and the cluster center vector:

$$\mathcal{Q}_{ik} = \frac{\mathrm{e}^{-\frac{\tau + \left(Z_i - Ex_k^p\right)^2}{\tau + 2En_k'^2}}}{\sum_{k=1}^K \mathrm{e}^{-\frac{\tau + \left(Z_i - Ex_k^p\right)^2}{\tau + 2En_k'^2}}} \tag{17}$$

where $\tau = 1$ is used to prevent the model collapse caused by the minimum value. $\mathcal{Q}_{ik}$ is the probability that the vector Z_i is assigned to the cluster Ex_k^p.

Aiming to strengthen the confidence of the clustering task and enhance the cohesion of each category in the hidden space, we introduced an auxiliary distribution $\mathcal{P}$ as follows [31]:

$$\mathcal{P}_{ik} = \frac{\mathcal{Q}_{ik}^2 / f_k}{\sum_{k'} \mathcal{Q}_{ik'}^2 / f_{k'}} \tag{18}$$

where $f_k = \sum_i \mathcal{Q}_{ik}$ are soft cluster frequencies, which standardizes the contribution of cluster centers. Then, we leverage KL divergence to calculate the difference between the two distributions:

$$\mathcal{L}_{self} = D_{KL}(\mathcal{P} \| \mathcal{Q}) = \sum_i \sum_k \mathcal{P}_{ik} \log \frac{\mathcal{P}_{ik}}{\mathcal{Q}_{ik}} \tag{19}$$

During the optimization procedure, a novel strategy of self-supervised learning [32] is used, where the distribution $\mathcal{P}$ depends on the distribution $\mathcal{Q}$, and the $\mathcal{Q}$ distribution is supervised by the $\mathcal{P}$ distribution.

Therefore, the joint optimization loss function is as following:

$$\begin{aligned} &\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{KL} + \mathcal{L}_{self} \\ &= \sum_{n=1}^{|N|} \|(\hat{x}_n - x_n) \odot B_n\|_2^2 \\ &+ \tfrac{1}{2} \sum_{d=1}^{D} \left[1 + \ln\left((En_d)^2\right) - (Ex_d)^2 - (En_d)^2\right] \\ &+ \sum_i \sum_k \mathcal{P}_{ik} \log \tfrac{\mathcal{P}_{ik}}{\mathcal{Q}_{ik}} \end{aligned} \tag{20}$$

4 Experiment

To verify the effectiveness of HNECV, we conducted several experiments on three real data sets and compared some state-of-the-art baselines, such as Deepwalk(DW) [5], self-supervised learning method DEC [29], SDCN [9] and VAE-based network embedding method VGAE [4], Metapath2vec (MP2V) [11], HERec [13], HAN [18]. Here, we mainly show the performance of the model on three different network tasks. The brief datasets are described as following:

- DBLP[5]: Contains 14376 papers(P), 14475 authors(A), 20 conferences(C), and 8920 Terms(T), average degree of the network is 4.73. The nodes are divided into 4 categories according to the author's research field.
- AMiner[6]: We extracted a subset of 13978 papers(P), 16543 authors(A), and 2,152 conferences(C), average degree of the network is 2.05. The nodes are divided into 8 categories according to the author's research field.
- Yelp[7]: Contains 2614 businesses(B), 1286 users(U), 8 star(St), and 2 services(S), average degree of the network is 9.22. We label according to the type of business, divided into 3 categories.

[5] https://dblp.uni-trier.de
[6] https://www.aminer.cn/citation
[7] https://www.yelp.com/dataset/

4.1 Classification

In this section, we adopt the KNN classifier with K=5 as the evaluation algorithm for node classification tasks, and use Micro-F1 and Macro-F1 as evaluation metrics. We perform the classification task 10 times and take the average value. The classifier selects representations of nodes in the network generated by the model randomly at a ratio of 30%, 50%, 70%, and 90% as the training sets.

Table 1. Experimental results of node classification task.

Datasets	Metrics	Train	DW	DEC	SDCN	VGAE	MP2V	HERec	HAN	HNECV
DBLP	Micro-F1	30%	0.6077	0.7832	0.6873	0.8444	0.9055	0.9088	0.9126	**0.9311**
		50%	0.6566	0.7992	0.7051	0.8603	0.9122	0.9106	0.9181	**0.9313**
		70%	0.6838	0.8105	0.7094	0.8686	0.9159	0.9173	0.9203	**0.9341**
		90%	0.7131	0.8305	0.7108	0.8815	0.9131	0.9224	0.9161	**0.9337**
	Macro-F1	30%	0.5880	0.7544	0.6798	0.8301	0.8984	0.9021	0.9010	**0.9261**
		50%	0.6399	0.7758	0.6983	0.8474	0.9054	0.9031	0.9075	**0.9269**
		70%	0.6682	0.7896	0.7019	0.8568	0.9105	0.9107	0.9094	**0.9292**
		90%	0.6984	0.8111	0.7060	0.8693	0.9052	0.9145	0.9053	**0.9288**
AMiner	Micro-F1	30%	0.6855	0.6757	0.2504	0.6497	0.6751	0.7036	0.5902	**0.7123**
		50%	0.7006	0.6853	0.2768	0.6665	0.6836	0.7068	0.6329	**0.7196**
		70%	0.7160	0.6906	0.3018	0.6774	0.6940	0.7070	0.6605	**0.7251**
		90%	0.7112	0.6978	0.3301	0.6876	0.6942	0.7147	0.6772	**0.7383**
	Macro-F1	30%	0.6790	0.6630	0.1890	0.6371	0.6553	0.6944	0.5662	**0.7069**
		50%	0.6947	0.6718	0.2209	0.6570	0.6766	0.6978	0.6162	**0.7130**
		70%	0.7101	0.6768	0.2507	0.6674	0.6840	0.6987	0.6474	**0.7186**
		90%	0.7052	0.6829	0.2846	0.6773	0.6848	0.7076	0.6601	**0.7289**
Yelp	Micro-F1	30%	0.7050	0.3556	0.3013	0.6711	0.6565	0.6601	**0.7342**	0.7253
		50%	0.7198	0.3577	0.3533	0.6732	0.6839	0.6720	0.7270	**0.7372**
		70%	0.7234	0.3706	0.3803	0.6757	0.6757	0.6605	0.7253	**0.7361**
		90%	0.7313	0.3691	0.3927	0.6840	0.6782	0.6599	0.7312	**0.7590**
	Macro-F1	30%	0.6447	0.3059	0.2169	0.5848	0.5770	0.5961	0.5999	**0.6742**
		50%	0.6678	0.3149	0.2129	0.5873	0.5907	0.5927	0.5916	**0.6900**
		70%	0.6711	0.3113	0.2304	0.5860	0.5905	0.5859	0.5919	**0.6887**
		90%	0.6763	0.3088	0.2802	0.5900	0.5953	0.5798	0.5809	**0.7160**

As shown in Table 1, HNECV outperforms all baselines in most cases. The Micro-F1 and Macro-F1 have increased by 1.3% and 4.6% respectively. For the network with small average degree, like the AMiner, the classification results of various methods almost are same because of its sparsity. Compared with AMiner, DBLP has additional Term nodes, HIN embedding exhibits better performances than homogeneous network embedding method. In addition, because Yelp is the network with larger average degree and the nodes in it have more neighbors. The models that use multiple meta-paths, such as HNECV and HAN, show good results due to the use of multi-view information. Since the node classification task focuses on the use of network structure information, the models that cannot capture the structure information of networks, such as DEC, SDCN and VGAE, have achieved poor classification results.

4.2 Clustering

Table 2. Experimental results of node clustering task.

Algorithms	DBLP		AMiner		Yelp	
	NMI	ARI	NMI	ARI	NMI	ARI
Deepwalk	0.5841	0.4960	0.3160	0.2227	0.2940	0.3179
DEC	0.7075	0.7686	0.1967	0.0355	0.0012	0.0009
SDCN	0.5977	0.5724	0.0033	0.0051	0.0005	0.0007
VGAE	0.6737	0.7275	0.1318	0.0474	0.1522	0.0855
Metapath2vec	0.6395	0.6369	0.2645	0.2083	0.3540	0.4047
HERec	0.6844	0.7104	0.3230	0.2322	0.3511	0.4018
HAN	0.5987	0.5929	0.0375	0.0165	**0.3635**	**0.4255**
HNECV	**0.7950**	**0.8471**	**0.3438**	**0.2765**	0.3584	0.4119

In this section, K-means algorithm is used as the clustering method to test the node clustering task, then NMI and ARI are used as the evaluation metrics. In the DBLP, AMiner, and Yelp networks, the values of K are set to 4, 8, and 3 respectively. We also perform the clustering task 10 times and record the average of the results. The results are shown in Table 2.

From the Table 2, it shows that HNECV has an average increase of 3.4% and 3.74% on NMI and ARI. It is worth noting that the HNECV in the Yelp dataset is slightly lower than the HAN method. This may be due to the fact that in the Yelp network with greater node average and more neighborhood information, it is difficult for the traditional random walk of the meta-path to fully consider the structure information of the network.

4.3 Robustness

In order to verify the stability of HNECV, we randomly deleted 30% of P-A links in the DBLP dataset and compared our method with some typical algorithms. Specifically, when we delete the P-A type links, we need also delete the author-related P-C and P-T links. Note that due to the particularity of HIN, it is difficult for us to randomly delete the links from the overall data. As we know, when an link is lost, its related links are needed to be deleted too. For simplicity, we start to delete the P-A type link. We compare ours with all above baselines, and calculate the average decline ratios of each metrics for every algorithm. The experimental results are shown in Table 3.

From the Table 3, we find that those algorithms which rely on the network structure have a large performance drop, such as DeepWalk and VGAE. The performance degradations of DEC and HAN in the classification task is moderate, but DEC has a sharp decline in the clustering task, which shows that it is not enough to learn quality node vector representations just by considering the clustering characteristics of the network. On the other hand, it also verifies the effectiveness of HAN and HNECV by considering multi-view information. In addition, because HNECV expands the sampling range of the hidden space, a lot of information in the original data can be captured in the hidden space. After combining with self-supervised clustering, the similar low dimension vectors will be cohesive more and more closely. It promises HNECV to achieve the most robust results in the entire comparison algorithm.

Table 3. Stability experiments with 30% link deleted.

Node classfication						
Metrics	Traning	DW	DEC	VGAE	HAN	HNECV
Micro-F1	30%	0.2820	0.6026	0.3574	0.7327	**0.8373**
	50%	0.2759	0.6310	0.4060	0.7376	**0.8450**
	70%	0.2831	0.6410	0.4202	0.7410	**0.8485**
	90%	0.2874	0.6477	0.4360	0.7444	**0.8536**
Macro-F1	30%	0.2443	0.5824	0.3356	0.7186	**0.8307**
	50%	0.2392	0.6122	0.3838	0.7273	**0.8393**
	70%	0.2008	0.6314	0.4001	0.7298	**0.8419**
	90%	0.1974	0.6317	0.4159	0.7241	**0.8481**
Avg Decline Ratio	/	61.80%	21.65%	53.99%	19.68%	**9.42%**
Node clustering						
NMI	100%	0.0212	0.0091	0.0081	0.3913	**0.5769**
ARI	100%	0.0224	0.0054	0.0053	0.3801	**0.5878**
Avg Decline Ratio	/	95.92%	99.00%	99.03%	35.27%	**29.02%**

5 Conclusion

In this paper, we compress the cloud model into the sampling procedure for hidden space of VAE framework to expand the sampling space. In addition, we leverage self-supervised learning module to promote the compactness of the sampling for hidden space. Experiments show that our model excels compared with various state-of-the-art algorithms. In particular, our method exhibits better robustness and steadiness. In the future, we will continue to work on the disentangling control of hidden space of VAE based on HNECV and cloud model.

References

1. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864. ACM, New York (2016)
2. Cui, P., Wang, X., Pei, J., Zhu, W.: A survey on network embedding. IEEE Transactions on Knowledge and Data Engineering **31**(5), 833–852 (2018)
3. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)
4. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710. ACM, New York (2014)
5. Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R.: struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 385–394. ACM, New York (2017)
6. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
7. Wang, C., Pan, S., Hu, R., Long, G., Jiang, J., Zhang, C.: Attributed graph clustering: A deep attentional embedding approach. arXiv preprint arXiv:1906.06532 (2019)
8. Fan, S., Wang, X., Shi, C., Lu, E., Lin, K., Wang, B.: One2multi graph autoencoder for multi-view graph clustering. In: Proceedings of The Web Conference 2020. pp. 3070–3076. ACM, New York (2020)
9. Bo, D., Wang, X., Shi, C., Zhu, M., Lu, E., Cui, P.: Structural deep clustering network. In: Proceedings of The Web Conference 2020. pp. 1400–1410. ACM, New York (2020)
10. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 135–144. ACM, New York (2017)
11. Fu, T.y., Lee, W.C., Lei, Z.: Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1797–1806. ACM, New York (2017)
12. Shi, C., Hu, B., Zhao, W.X., Philip, S.Y.: Heterogeneous information network embedding for recommendation. IEEE Transactions on Knowledge and Data Engineering **31**(2), 357–370 (2018)
13. Tang, J., Qu, M., Mei, Q.: Pte: Predictive text embedding through large-scale heterogeneous text networks. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1165–1174. ACM, New York (2015)
14. Xu, L., Wei, X., Cao, J., Yu, P.S.: Embedding of embedding (eoe) joint embedding for coupled heterogeneous networks. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. pp. 741–749. ACM, New York (2017)
15. Wang, H., Zhang, F., Hou, M., Xie, X., Guo, M., Liu, Q.: Shine: Signed heterogeneous information network embedding for sentiment link prediction. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. pp. 592–600. ACM, New York (2018)

16. Hu, B., Shi, C., Zhao, W.X., Yu, P.S.: Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1531–1540. ACM, New York (2018)
17. Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., Yu, P.S.: Heterogeneous graph attention network. In: The World Wide Web Conference. pp. 2022–2032. ACM, New York (2019)
18. Hu, B., Fang, Y., Shi, C.: Adversarial learning on heterogeneous information networks. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 120–129. ACM, New York (2019)
19. Shi, C., Lu, Y., Hu, L., Liu, Z., Ma, H.: Rhine: Relation structure-aware heterogeneous information network embedding. IEEE Transactions on Knowledge and Data Engineering **99**(1), 1–15 (2020)
20. Cen, Y., Zou, X., Zhang, J., Yang, H., Zhou, J., Tang, J.: Representation learning for attributed multiplex heterogeneous network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1358–1368. ACM, New York (2019)
21. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 793–803. ACM, New York (2019)
22. Hu, B., Zhang, Z., Shi, C., Zhou, J., Li, X., Qi, Y.: Cash-out user detection based on attributed heterogeneous information network with a hierarchical attention mechanism. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 946–953. AAAI, Palo Alto (2019)
23. Li, J., Peng, H., Cao, Y., Dou, Y., Zhang, H., Yu, P.S., He, L.: Higher-order attribute-enhancing heterogeneous graph neural networks. arXiv preprint arXiv:2104.07892 (2021)
24. Wu, L., Li, Z., Zhao, H., Liu, Q., Wang, J., Zhang, M., Chen, E.: Learning the implicit semantic representation on graph-structured data. arXiv preprint arXiv:2101.06471 (2021)
25. Li, X., Wen, L., Qian, C., Wang, J.: Gahne: Graph-aggregated heterogeneous network embedding. In: 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence. pp. 1012–1019. IEEE, Baltimore (2020)
26. Shi, C., Li, Y., Zhang, J., Sun, Y., Philip, S.Y.: A survey of heterogeneous information network analysis. IEEE Transactions on Knowledge and Data Engineering **29**(1), 17–37 (2016)
27. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. Proceedings of the VLDB Endowment **4**(11), 992–1003 (2011)
28. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
29. Li, D., Han, J., Shi, X., Chan, M.C.: Knowledge representation and discovery based on linguistic atoms. Knowledge-Based Systems **10**(7), 431–440 (1998)
30. Wang, G., Xu, C., Li, D.: Generic normal cloud model. Information Sciences **280**, 1–15 (2014)
31. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: International conference on machine learning. pp. 478–487. PMLR, ACM, New York (2016)
32. Nigam, K., Ghani, R.: Analyzing the effectiveness and applicability of co-training. In: Proceedings of the ninth international conference on Information and knowledge management. pp. 86–93. ACM, New York (2000)