Deep Attributed Network Embedding Based on the PPMI

Kunjie Dong, Tong Huang, Lihua Zhou(✉), Lizhen Wang, and Hongmei Chen

School of Information, Yunnan University, Kunming 650091, China
kunjiedong@qq.com, huangtong@mail.ynu.edu.cn,
{lhzhou,hmchen}@ynu.edu.cn, lzhwang2005@126.com

Abstract. The attributed network embedding aims to learn the latent low-dimensional representations of nodes, while preserving the neighborhood relationship of nodes in the network topology as well as the similarities of attribute features. In this paper, we propose a deep model based on the positive point-wise mutual information (PPMI) for attributed network embedding. In our model, attribute features are transformed into an attribute graph, such that attribute features and network topology can be handled in the same way. Then, we perform the random surfing and calculate the PPMI on the attribute/topology graph to effectively maintain the structural characteristics and the high-order proximity information. The node representations are learned by a shared Auto-Encoder. Besides, the local pairwise constraint is used in the shared Auto-Encoder to improve the quality of node representations. Extensive experimental results on four real-world networks show the superior performance of the proposed model over the 10 baselines.

Keywords: Attributed network embedding · Random surfing · Positive point-wise mutual information · Auto-encoder

1 Introduction

Network embedding (NE) aims to learn the latent low-dimensional representations of nodes in a network while preserving the intrinsic essence of the network [8,15,19], which can provide precise service and higher efficiency in practical applications, such as targeted detection and personalized recommendation [22,28]. Therefore, NE has aroused many researchers' interests under the drive of great requirements in recent years.

Supported by the National Natural Science Foundation of China (61762090, ,62062066, 61966036, and 61662086), the Natural Science Foundation of Yunnan Province (2016FA026), the Program for Innovation Research Team (in Science and Technology) in University of Yunnan Province (IRTSTYN), and the National Social Science Foundation of China (18XZZ005).

K. Dong and T. Huang—Both authors have contributed equally to this work.

C. S. Jensen et al. (Eds.): DASFAA 2021 Workshops, LNCS 12680, pp. 251–266, 2021.
https://doi.org/10.1007/978-3-030-73216-5_18

In a network consisting of nodes and edges, nodes represent objects and edges describe the interactive relationships amongst nodes. For example, in a cite network, nodes represent papers and edges describe the cite relationship amongst papers. In general, the interactive relationships amongst nodes are referred to as network topology, which plays a vital role in network analysis tasks. Network topology, typically in the form of node adjacency matrix, is the most common form of network representation. An important goal of NE is to preserve the neighborhood relationship of nodes in the network topology. To this end, various NE methods, such as DeepWalk [15] used the random walks based on the sampling strategies to convert a general graph structure into a large collection of linear sequences, and then utilized the skip-gram model [13] to learn low-dimensional representations for nodes from such linear sequences. This is one effective way to express graph structural information, because the sampled node sequences characterize the connections amongst nodes in a graph. However, the procedure involves a slow sampling process, and the hyper-parameters (such as walk length and total walks) are not easy to determine, especially for the large graphs. Because the sampled sequences have finite lengths, furthermore, it is difficult to capture the correct contextual information for nodes that appear at the boundaries of the sampled sequences, such that some relationships amongst nodes cannot be captured accurately and completely. To make up for the shortcomings of random walk, DNGR [4] adopts a random surfing model to capture graph structural information directly, instead of using the sampling-based method for generating linear sequences. The random surfing model first randomly orders the nodes in a graph, and then directly yield a probabilistic co-occurrence (PCO) matrix that capturing the transition probabilities amongst different nodes. Based on the PCO matrix, the positive point-wise mutual information (PPMI) can be computed, which avoids the expensive sampling process. As an explicit representation of a graph, the PPMI can effectively maintain the structural characteristics of a graph and contain the high-order similarity information of the nodes [4], so the PPMI representations of nodes can more accurately capture potentially complex, non-linear relations amongst different nodes. But DNGR used network topology alone while did not take the attribute features affiliated to nodes into consideration.

The attribute features affiliated to nodes, such as authors, research themes and keywords associated with papers in a citation network, describe the individual profile of nodes in a micro-perspective. This information often carries orthogonal and complementary knowledge beyond node connectivity and network topology, so incorporating semantic information is expected to significantly enhance NE based on network topology alone. A network whose nodes are associated with attribute features referred to as an attributed network [2]. The embedding of an attributed network (ANE) aims to learn the latent low-dimensional representations of nodes while preserving the neighborhood relationship of nodes in the network topology as well as the semantics of attribute features. This is not a trivial task, because network topology and attribute features are two heterogeneous information, although they describe the same network from two different

perspectives [5,9]. How to integrate two heterogeneous information and preserve the intrinsic essence contained in network topology and attribute features simultaneously is a key issue in ANE. Some existing approaches, such as TADW [24], ASNE [12], CANE [20], first converted network topology into the feature representations, then which were used to embed into a low-dimensional space. Meanwhile, attribute features were also used to derive low-dimensional embedding on node semantics. The two low-dimensional representations of all these NEs are concatenated to joint learn the final embedding. Due to converting a network topology into a feature representation may lose or may not faithfully represent non-linear relationship amongst the nodes [11], and the individual feature vector only contains individual information without inter-individual association relationships, combining topological feature vector and attribute feature vector together may unsatisfactory to explore and exploit the complementary relationship between these two types of information. Since, UWMNE [11] maintained network topology in the graph form and built attribute graph to represent semantic information, and then used deep neural networks to integrate the topological and semantic information in these graphs to learn a unified embedding representation.

Inspired by the DNGR [4] and the UWMNE [11], we propose a deep model based on the PPMI for ANE in this paper. The model is referred to as DANEP. Specifically, we first transform attribute features into an attribute graph, which is homogeneous with topology graph, so we can deal with them in the same way. Next, we carry out random surfing on the attribute/topology graph respectively to generate a attribute/topology probabilistic co-occurrence (PCO) matrix, and then calculate the attribute/topology PPMI based on the attribute/topology PCO matrix. After that, using a shared Auto-Encoder to learn low-dimensional node representations. The advantages of DANEP lie in: the attribute graph describes the geometry of potential non-linear manifolds under attribute features information more clearly, the uniformed graph representation of attribute features and network topology contributes to integrating the complementary relationship between two types of information; the random surfing captures graph structural information concerning attribute/topology, the PPMIs calculated from the attribute/topology PCO matrixes effectively maintain both the structural characteristics and the high-order proximity information of attribute/topology graph; and the shared Auto-Encoder learns high-level abstractions from low-level features as well as captures highly non-linear information conveyed by the graph via non-linear projections. Besides, the local pairwise constraint is further designed in shared Auto-Encoder to improve the quality of node representations. We also conduct extensive experiments on four real-world networks and compare our approach with 10 baselines. The experimental results demonstrate the superiority of our approach.

It is needed to note that our DANEP model is different from the DNGR [4] and the UWMNE [11]. In our DANEP model, we apply the deep learning method on PPMIs of attribute features and network topology, but the DNGR just apply

matrix factorization on PPMI of network topology, and the UWMNE directly use network topology and attribute graph as the input of an Auto-Encoder.

The rest of the paper is arranged as follows. Section 2 offers a brief overview of related work. The details of DANEP are presented in Sect. 3. Section 4 provides extensive experiments and results, and in Sect. 5, conclusions are given.

2 Related Work

2.1 Network Embedding

Many network embedding approaches only utilized network topology to learn the latent low-dimensional representations. DeepWalk [15] first employed the truncated random walks to capture the local information and then learn the latent embedding result by making use of the local information. Node2vec [8] proposed a biased random walk method to explore various neighborhoods. Line [19] considered to preserve the first-order and second-order proximity of network topology into the learned embedding representation. SDNE [21] proposed a semi-supervised model to jointly preserve the first-order and second-order similarity of network topology. Struc2vec [17] utilized a weighted random walk to obtain a similar node sequence and conceived a hierarchical structure strategy to capture node proximity at different scales. GraRep [3] integrated global structural information learned from different models into the embedding representation. DNGR [4] first adopted a random surfing model to capture graph structural information, and then used a stacked denoising Auto-Encoder to learn low-dimensional vertex representations.

2.2 Attributed Network Embedding

In recent years, many researchers learned representations of nodes by integrating network topology and attribute features of nodes. This brings new opportunities and development for embedding learning. In detail, AANE [10] considered the proximity of the attribute features into embedding learning and adopted a distributed manner to accelerate the learning process. TADW [24] proposed the text-associated DeepWalk model to integrate node's text features into embedding learning by matrix factorization. ASNE [12] adopted a deep neural network to model the complex interrelations between attribute features and network topology. DANE [6] employed two symmetrical Auto-Encoders to capture the consistency and complementary information between attribute features and network topology, where the two symmetrical Auto-Encoders are allowed to interact with each other. ANRL [27] utilized a neighbor enhancement Auto-Encoder with attribute-aware skip-gram to extract the correlations of attribute features and the network topology. NANE [14] considered the local and global information in the embedding process by a pairwise constraint. Based on the observations that nodes with similar topology may be dissimilar in their attribute features and vice versa,which are referred to as the partial correlation, PRRE [29] taken the partial correlation of nodes into account in the learning process.

3 The Proposed Model

In this section, we first present the definition of ANE and then develop a deep attributed network embedding model based on the positive pointwise mutual information.

3.1 Problem Definition

Given an attributed network with n nodes and m edges $G = (V, E, \mathbf{A})$, wherein $V = \{v_1, \cdots, v_n\}$ and $E = \{e_{ij}\}_{i,j=1}^{n}$ represent the sets in items of nodes and edges, respectively, and $\mathbf{A} \in R^{n \times m}$ represent the attribute matrix affiliated to the nodes, whose row vector $\mathbf{a}_i \in R^m$ corresponds to the attribute features of the node v_i. Let $\mathbf{S} \in R^{n \times n}$ be the adjacent matrix affiliated to the edges, whose the element s_{ij} corresponds to the relationship of the edge between nodes v_i and v_j, i.e., $s_{ij} = 1$ indicates there exists an edge linked v_i to v_j, and $s_{ij} = 0$ indicates the edge is nonexistent. The goal of the ANE is to find a map function $f(\mathbf{A}, \mathbf{S}) \rightarrow \mathbf{H}$ that map attribute features $\mathbf{A}$ and network topology $\mathbf{S}$ into a unified low-dimensional representation $\mathbf{H} \in R^{n \times d} (d \ll n, d \ll m)$ while preserving the proximities existing in both the attribute of the nodes and the topology of the network. More precisely, nodes with similar attribute and topology in the original network should be closer in the embedding space.

3.2 The Architecture of Proposed Model

The architecture of DANEP is shown in Fig. 1. DANEP first constructs an attribute graph based on the attribute features A, such that the attribute graph and the topology graph are homogeneous. Based on the homogeneous representations of the attribute graph and topology graph, the random surfing is first conducted to obtain the attribute/topology probabilistic co-occurrence (PCO) matrix, and then the PPMIs concerning the attribute graph and topology graph are calculated, represented as **PPMI_AF** and **PPMI_NT**, respectively. The row vectors of **PPMI_AF** and **PPMI_NT** depict the profile and the neighborhood relationships of node v_i with respect to attribute features and network topology. After that, a shared Auto-Encoder equipped with the local enhancement of graph regulation is applied to learn the unified low-dimensional representation for each node from the PPMIs concerning the attribute graph and topology graph.

The Construction of the Attribute Graph. In this subsection, we construct an attribute graph based on the attribute features $\mathbf{A}$. Let $\mathbf{B} \in R^{n \times n}$ be the attribute similarity matrix, whose elements $b_{ij} \in \mathbf{B}$ can be measured by the similarity of attribute vectors $\mathbf{a}_i \in \mathbf{A}$ and $\mathbf{a}_j \in \mathbf{A}$, such as the cosine similarity can be calculated by Eq. (1), where "$\cdot$" signifies the dot product of the two vectors, "$||\cdot||$" denotes $L2$ norm, and "$\times$" indicates the product of two scalars.

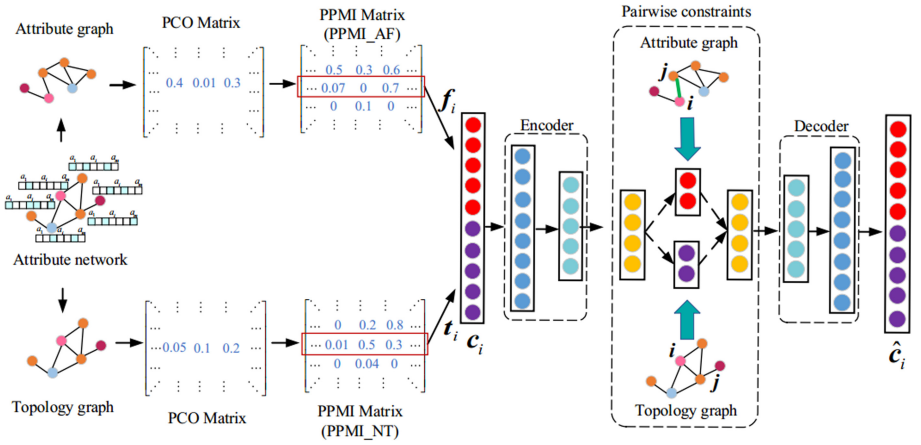

Fig. 1. The architecture of DANEP.

$$b_{ij} = \frac{\mathbf{a}_i \cdot \mathbf{a}_j}{\|\mathbf{a}_i\| \times \|\mathbf{a}_j\|} \tag{1}$$

Intuitively, the distance between two nodes is closer, the more intimate relationship they should have. Therefore, we apply the k-nearest neighbor method [11,18] on the $\mathbf{B}$ to construct the attribute graph with n nodes, where each node v_i is connected to k nodes with top-k similarities in b_i. Let $\mathbf{B}^{new} \in R^{n \times n}$ be the adjacent matrix of the constructed attribute graph, then the element $b_{ij}^{new} = 1$ indicates there exists an edge linked v_i to v_j, and $b_{ij}^{new} = 0$ indicates the edge is nonexistent.

The Calculation of PPMIs. Motivated by DNGR [4], we adopt the random surfing model on the topology graph $\mathbf{S}$/attribute graph $\mathbf{B}^{new}$ to obtain the attribute/topology probabilistic co-occurrence (PCO) matrix through k-step iterative. The iterative process can be represented by Eq. (2), where $\mathbf{p}_0$ is the initial one-hot vector with i-th value is 1 and the other values are 0, coefficient α and $1 - \alpha$ represent the probabilities with respect to the node jumps to the next node and returns to original vertex (restart), respectively.

$$\mathbf{p}_k = \alpha \cdot \mathbf{p}_{k-1} + (1 - \alpha)\mathbf{p}_0 \tag{2}$$

Based on the attribute/topology PCO matrix, the pointwise mutual information (PMI) can be calculated by Eq. (3), where $p(v_i, v_j)$ represents the number of co-occurrences that nodes v_i and v_j are in the same context, $|D| = \sum_{v_i} \sum_{v_j} p(v_i, v_j)$, $p(v_i)$ and $p(v_j)$ represents the number of occurrences of nodes v_i and v_j, respectively.

$$\mathbf{PMI}_{v_i, v_j} = \log\left(\frac{p(v_i, v_j) \cdot |D|}{p(v_i) \cdot p(v_j)}\right) \tag{3}$$

Then, PPMI can be calculated by Eq. (4) [23], which means that negative values in attribute/topology PMI are assigned to zeros.

$$\mathbf{PPMI}_{v_i,v_j} = \max(\mathbf{PMI}_{v_i,v_j}, 0) \tag{4}$$

The Design of the Shared Auto-Encoder. In general, an Auto-Encoder consists of an encoder and a decoder which can extract inherent essence and non-linear information of a network. In DANEP, we designed a shared Auto-Encoder with $2K-1$ layers to incorporate attribute features and network topology. The input of the Auto-Encoder is the concatenation of the row vectors of **PPMI_AF** and **PPMI_NT**, i.e. $\mathbf{c}_i = (\mathbf{f}_i, \mathbf{t}_i) = (f_{i1}, \cdots, f_{in}, t_{i1}, \cdots, t_{in})$, where $\mathbf{C} = [\mathbf{F}, \mathbf{T}] \in R^{n \times 2n}$, $\mathbf{c}_i$, $\mathbf{f}_i$ and $\mathbf{t}_i$ are the i-th row vector of $\mathbf{C}$, $\mathbf{F}$, and $\mathbf{T}$, respectively. Let $\mathbf{y}_{i,k}(k = 1, \cdots, K)$ and $\hat{\mathbf{y}}_{i,k}(k = 1, \cdots, K)$ be the desired embedding representation and the reconstructed representation of the Auto-Encoder, then $\mathbf{y}_{i,k}(k = 1, \cdots, K)$ and $\hat{\mathbf{y}}_{i,k}(k = 1, \cdots, K)$ can be computed by Eq. (5)–(9).

$$\mathbf{y}_{i,1} = f(\mathbf{W}_1 \mathbf{c}_i + \mathbf{b}_1) \tag{5}$$

$$\mathbf{y}_{i,k} = f(\mathbf{W}_k \mathbf{y}_{i,k-1} + \mathbf{b}_k)(k = 2, \cdots, K-1) \tag{6}$$

$$\mathbf{y}_i = \hat{\mathbf{y}}_{i,1} = \mathbf{y}_{i,K} = f(\mathbf{W}_K \mathbf{h}_{i,K-1} + \mathbf{b}_K) \tag{7}$$

$$\hat{\mathbf{y}}_{i,k} = f(\mathbf{W}_{K+k-1} \hat{\mathbf{y}}_{i,k-1} + \mathbf{b}_{K+k-1})(k = 2, \cdots, K-1) \tag{8}$$

$$\hat{\mathbf{y}}_{i,K} = f(\mathbf{W}_{2K-1} \hat{\mathbf{y}}_{i,K-1} + \mathbf{b}_{2K-1}) \tag{9}$$

Where $f(\cdot)$ represents the non-linear activation function, and $\theta = \{\mathbf{W}_k, \mathbf{b}_k\}(k = 1, \cdots, 2K-1)$ are weight and bias parameters of the shared Auto-Encoder.

Let $\hat{\mathbf{C}}$ be the output of the decoder, where $\hat{\mathbf{c}}_i = \hat{\mathbf{y}}_{i,K} = f(\mathbf{W}_{2K-1} \hat{\mathbf{y}}_{i,K-1} + \mathbf{b}_{2K-1})$. The goal of Auto-Encoder is to minimize the reconstruction loss between the $\mathbf{C}$ and $\hat{\mathbf{C}}$, so the loss function is defined as:

$$\mathcal{L}_{rec} = \sum_{i=0}^{n} ||\hat{\mathbf{c}}_i - \mathbf{c}_i||_2^2 \tag{10}$$

To further improve the quality of node representation of the shared Auto-Encoder, we designed the local pairwise constraint, which is used to reinforce the consistency and complementary information contained in attribute features and network topology. Given the adjacent matrix $\mathbf{S}/\mathbf{B}^{new}$ of attribute/topology graph, the local pairwise constraint is defined as:

$$\begin{aligned} \mathcal{L}_{local} = \frac{1}{2} \sum\nolimits_{i=1}^{n} \sum\nolimits_{j=1}^{n} s_{ij} ||\mathbf{y}_i - \mathbf{y}_j||_2^2 + \frac{1}{2} \sum\nolimits_{i=1}^{n} \sum\nolimits_{j=1}^{n} b_{ij}^{new} ||\mathbf{y}_i - \mathbf{y}_j||_2^2 \\ = tr((\mathbf{Y}^C)^T \mathbf{L}_1 \mathbf{Y}^C) + tr((\mathbf{Y}^C)^T \mathbf{L}_2 \mathbf{Y}^C) \end{aligned} \tag{11}$$

where $\mathbf{L}_1 = \mathbf{D}' - \mathbf{S}$, $\mathbf{L}_2 = \mathbf{D}'' - \mathbf{B}^{new}$, both $\mathbf{D}' = [d'_{ij}] \in R^{n \times n}$ and $\mathbf{D}'' = [d''_{ij}] \in R^{n \times n}$ are diagonal matrices, $D'_{ii} = \sum_{j=1}^{n} s_{ij}$, $D''_{ii} = \sum_{j=1}^{n} b_{ij}^{new}$.

Thus, the objective function of the DANEP is defined as:

$$\mathcal{L} = \alpha \mathcal{L}_{local} + \beta \mathcal{L}_{rec} \tag{12}$$

Where α and β are the hyper-parameter to balance the weights among different losses.

4 Experiments and Results

In this section, we conduct extensive experiments on the four real-world networks by adopting three widely used applications, i.e., node classification, node clustering, and network visualization, to evaluate the effectiveness of our proposed method DANEP.

4.1 Datasets

In experiments, four publicly available networks with class labels are used, i.e., Cora, Citeseer, BlogCatelog and Flicker networks, where the first two datasets are academic papers citation network, and the last two datasets are the social networks. In Cora/Citeseer networks, nodes and edges represent academic papers and the citation relationships amongst those papers, respectively, each paper can be represented as a bag-of-words vector with 1433/3703-dimensions, and papers are divided into 7/6 categories, such as Genetic algorithm, Neural Networks and Reinforcement Learning. In BlogCatelog/ Flicker networks, nodes and edges represent the users and relationships amongst those users, respectively, each user can be represented as a bag-of-words vector with 8189/12047-dimensions, and those users are divided into 6/9 categories based on social preferences. The statistics for each network are summarized in Table 1.

Table 1. The statistics of networks.

Dataset	Nodes	Edges	Features	Classes
Cora	2708	5278	1433	7
Citeseer	3312	4660	3703	6
BlogCatalog	5196	171743	8189	6
Flicker	7575	239738	12047	9

4.2 Baselines

To verify the effectiveness of DANEP model, we select 10 approaches as the baselines, including: 4 "Topology-only" algorithms, i.e., DeepWalk [15], Node2Vec [8], GraRep [3], DNGR [4], and 6 "Topology +Attribute" algorithms, i.e., AANE [10], TADW [24], ASNE [12], DANE [6], ANRL [27], NANE [14]. The details of these baselines are illustrated as follows:

"Topology-Only" Algorithms: DeepWalk [15]: It employed the truncated random walks to capture the local topology information, and then learned the latent embedding representation by making full use of the captured local information.

Node2Vec [8]: It proposed biased random walks to project node into a low-dimensional space while preserving the network essence by exploring and preserving network neighborhoods of nodes.

GraRep [3]: It developed a model to learn the node representation for the weighted graph by integrating global structural similarity in the learning process.

DNGR [4]: It adopted a random surfing model to capture topology information, and then utilized the stacked denoising Auto-Encoder to extract meaningful information into the low-dimensional vector representation.

"Topology +Attribute" Algorithms: AANE [10]: AANE considered and integrated the proximity of attribute features into the embedding learning and adopted a distributed manner to accelerate the learning process.

TADW [24]: It employed a matrix factorization method based on DeepWalk to learn low-dimensional representations of text and network topology, and then concatenate them to form the final representation.

DANE [6]: DANE allowed neighborhood topology obtained by random walks and attribute features to interact with each other to preserve the consistent and complementary information during the learning process.

ANRL [27]: It designed a neighbor enhancement Auto-Encoder model with an attribute-aware skip-gram to integrate the attribute features and network topology proximities in the learning process simultaneously.

ASNE [12]: ASNE integrated the adjacent matrix of network topology and attribute matrix on the input layer, and allowed them to interact with each other for capturing the complex relationships and the more serviceable information.

NANE [14]: It cascaded the adjacent matrix of network topology and cosine similarity of attribute features into the unified representation to capture the local information and non-linear correlation in the network.

4.3 Parameter Settings

To get a fair comparison, we set the embedding dimension d of all datasets to be 128 for all baselines. For DeepWalk and Node2Vec, we set the window size as 10, the walk length as 80, and the number of walks per node as 10. For GraRep, the maximum transition step is set to 5. For TADW, we set the regularization parameter to 0.2. Besides, the default values of the other parameters for these methods were set the same as the open-source codes released by the original authors.

4.4 Node Classification

In this subsection, we randomly select 10%, 30%, 50% nodes as the training set and the remained nodes as the testing set, apply the linear SVM as the classifier

Table 2. The performance evaluation of node classification.

Metrics	Methods	10%		30%		50%	
		Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1
Cora	DeepWalk	0.7341	0.7180	0.7905	0.7796	0.8233	0.8136
	Node2Vec	0.7059	0.6880	0.7747	0.7627	0.8013	0.7913
	GraRep	0.7375	0.7207	0.7750	0.7561	0.7835	0.7631
	DNGR	0.6493	0.6380	0.7071	0.6968	0.7335	0.7164
	AANE	0.6582	0.6177	0.7195	0.6891	0.7305	0.7015
	TADW	0.7945	0.7777	0.8360	0.8220	0.8436	0.8298
	DANE	0.7751	0.7545	0.8188	0.8033	0.8302	0.8143
	ANRL	0.7487	0.7224	0.7659	0.7430	0.7731	0.7538
	NANE	0.5114	0.4660	0.5816	0.5470	0.6459	0.6113
	ASNE	0.457	0.4353	0.5347	0.5099	0.5723	0.5466
	DANEP	**0.8215**	**0.8078**	**0.8398**	**0.8259**	**0.8575**	**0.8435**
Citeseer	DeepWalk	0.5037	0.4723	0.5888	0.5502	0.6199	0.5790
	Node2Vec	0.4875	0.4508	0.5621	0.5223	0.5851	0.5401
	GraRep	0.5063	0.4644	0.5421	0.4881	0.5542	0.4957
	DNGR	0.4581	0.4228	0.4970	0.4547	0.5331	0.4895
	AANE	0.6549	0.6008	0.6834	0.6293	0.6930	0.6436
	TADW	0.6621	0.6181	0.7182	0.6520	**0.7421**	**0.6972**
	DANE	0.6415	0.5960	0.6950	0.6520	0.7163	0.6722
	ANRL	**0.6795**	**0.6371**	0.7270	0.6747	0.7397	0.6880
	NANE	0.4488	0.4171	0.5783	0.5334	0.6298	0.5749
	ASNE	0.3261	0.3028	0.4110	0.3695	0.4385	0.3874
	DANEP	0.6494	0.5970	**0.7357**	**0.6834**	0.7349	0.6787
Flickr	DeepWalk	0.4389	0.4352	0.5223	0.5144	0.5483	0.5385
	Node2Vec	0.3899	0.3863	0.4896	0.4804	0.5171	0.5049
	GraRep	0.4908	0.4829	0.5422	0.5327	0.5558	0.5466
	DNGR	0.4656	0.4027	0.4653	0.4552	0.4768	0.4656
	AANE	0.5865	0.6068	0.6151	0.6323	0.6244	0.6369
	TADW	0.6117	0.6026	0.7020	0.6940	0.7218	0.7143
	DANE	0.6453	0.6439	0.7160	0.7144	0.7395	0.7380
	ANRL	0.2740	0.1984	0.2947	0.2268	0.2978	0.2250
	NANE	0.3733	0.3690	0.4993	0.4931	0.5290	0.5224
	ASNE	0.4366	0.4316	0.5218	0.5116	0.5500	0.5413
	DANEP	**0.8457**	**0.8431**	**0.8585**	**0.8565**	**0.8690**	**0.8670**
BlogCatalog	DeepWalk	0.5781	0.5733	0.6624	0.6549	0.6899	0.6811
	Node2Vec	0.5296	0.5258	0.6283	0.6215	0.6592	0.6509
	GraRep	0.6890	0.6851	0.7272	0.7230	0.7450	0.7413
	DNGR	0.5896	0.5850	0.6515	0.6423	0.6682	0.6585
	AANE	0.8565	0.8539	0.8846	0.8828	0.8920	0.8897
	TADW	0.8199	0.8175	0.8610	0.8799	0.8789	0.8772
	DANE	0.8436	0.8400	0.8180	0.8752	0.8876	0.8856
	ANRL	0.8073	0.8004	0.8285	0.8225	0.8333	0.8274
	NANE	0.6806	0.6778	0.7764	0.7739	0.8043	0.8016
	ASNE	0.5838	0.5823	0.6651	0.6615	0.6759	0.6695
	DANEP	**0.8749**	**0.8739**	**0.9126**	**0.9117**	**0.9226**	**0.9219**

Table 3. The performance evaluation of node clustering.

Metrics	Methods	Cora	Citeseer	BlogCatalog	Flickr	Average
ACC	DeepWalk	0.5609	0.4029	0.3646	0.3163	0.4112
	Node2Vec	0.6128	0.4208	0.3553	0.3209	0.4275
	GraRep	0.5027	0.3238	0.3710	0.2934	0.3727
	DNGR	0.5948	0.3929	0.3599	0.2750	0.4057
	AANE	0.3904	0.5379	0.4320	0.1338	0.3735
	TADW	0.6686	0.5689	0.6848	0.3640	0.5716
	DANE	**0.7187**	0.4884	0.4879	0.2445	0.4849
	ANRL	0.5129	0.5730	0.4720	0.2137	0.4429
	NANE	0.1503	0.2430	0.5837	0.2861	0.3158
	ASNE	0.3889	0.4088	0.3896	0.2283	0.3539
	DANEP	0.7179	**0.6097**	**0.7048**	**0.6886**	**0.6803**
NMI	DeepWalk	0.4021	0.1371	0.1966	0.1694	0.2263
	Node2Vec	0.4396	0.2227	0.2060	0.1801	0.2621
	GraRep	0.3749	0.1673	0.2040	0.1482	0.2236
	DNGR	0.4424	0.2017	0.1836	0.1462	0.2435
	AANE	0.2206	0.2774	0.2759	0.0901	0.216
	TADW	**0.5515**	0.3550	0.4352	0.1833	0.3813
	DANE	0.5494	0.2975	0.3277	0.1165	0.3228
	ANRL	0.3812	0.3619	0.3417	0.1004	0.2963
	NANE	0.2096	0.2637	0.3583	0.1329	0.2411
	ASNE	0.2096	0.1221	0.2165	0.1290	0.1693
	DANEP	0.5510	**0.3792**	**0.5207**	**0.6002**	**0.5128**

and use 5-fold cross-validation to train the classifier in the learning process. This process is repeated 10 times and the average performance in terms of both Macro-F1 and Micro-F1 [25] is reported as the classification results. The detailed results are shown in Table 2, where the bold numbers indicate the best results. From Table 2, we have the following observations and analyses:

(1) DANEP obtains the best performance with respect to the Micro-F1 and Macro-F1 on the Cora, Flickr and BlogCatalog datasets when the training rates are 10%, 30% and 50%, respectively. The improved performance concerning the Micro-F1 and Macro-F1 is significantly on different datasets, such as DANEP achieves the 20.04%, 19.92%, 14.25%, 14.21%, 12.95 and 12.9% than the best baseline DANE on Flicker dataset when the training rates are 10%, 30% and 50%, respectively. Those results demonstrated the superiority of DANEP with random surfing and PPMI schemes.
(2) ANRL and TADW achieve the highest values on the Citeseer dataset when the training rates are 10% and 50%, respectively, which indicate that the neighbor enhancement mechanism and text-associated matrix factorization have some the ability to capture the essence of the network, but they are still obviously inferior to DANEP.

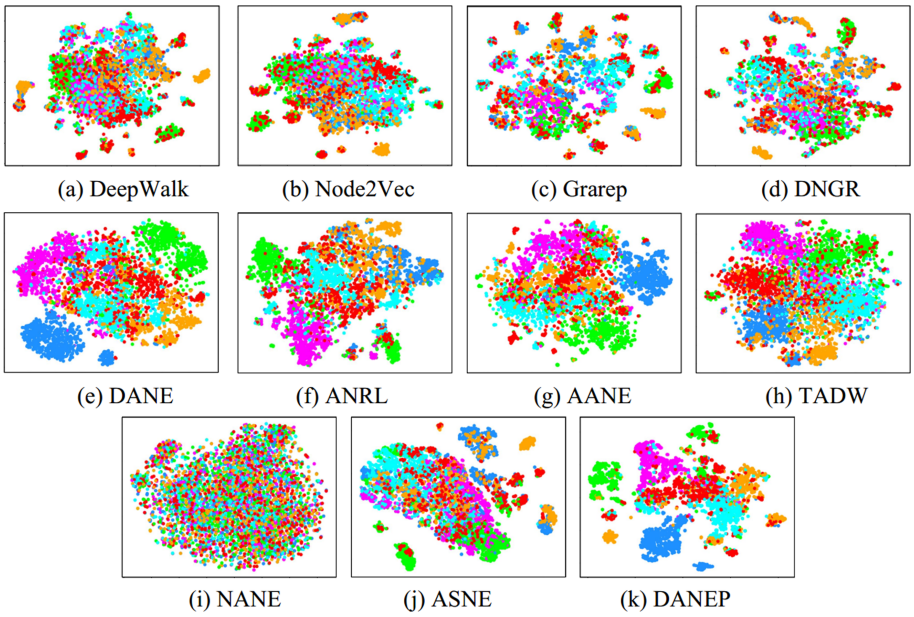

Fig. 2. The visualization result of different methods on the BlogCatalog dataset

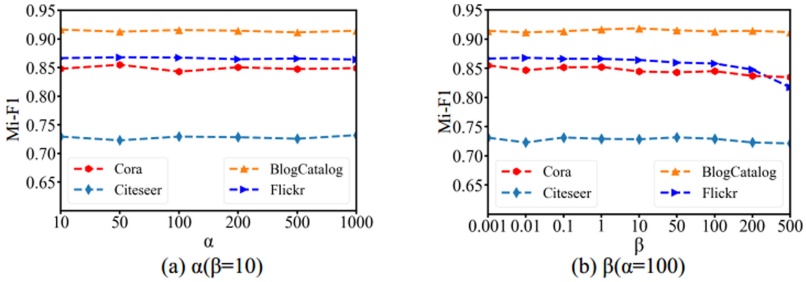

Fig. 3. The sensitivity of DANEP w.r.t. different α and β for node classification

4.5 Node Clustering

Node clustering is an unsupervised downstream task of network analysis based on the learned node representation. In this study, we use k-means [1] as the clustering algorithm, accuracy (ACC) [6] and normalized mutual information (NMI) [14] as metrics to evaluate the clustering performance. Similarly, this process is repeated 10 times and the average performance in terms of both ACC and NMI is reported as the clustering results. The final results for each baseline are shown in Table 3. From Tables 3, we have the following observations and analyses:

(1) DANEP acquires the best clustering performance on the Citeseer, BlogCatalog and Flickr datasets against all the baselines. The promotion of performance is significantly on different datasets, such as, DANEP with 32.46% and 41.69% than the best baseline TADW on the Flicker dataset. Besides, DANEP ranked the second on the Cora dataset, but only with the slightly inferior in ACC than DANE, i.e., −0.008, and in NMI than TADW, i.e., −0.005, respectively. Those results indicated that DANEF based on the graph representation and PPMI has a good clustering performance than all baselines.
(2) From the perspective of average performance, TADW obtains better clustering results than the other baselines, but TADW is seriously inferior to DANEP. In detail, DANEP averagely improves 10.87% in ACC and 13.11% in NMI than TADW, which demonstrated attribute graph and PPMI matrix have powerful assistance in node clustering.

4.6 Network Visualization

To verify whether the learned node representations have the discriminative essence features, we use the t-SNE [16] to project the learned embedding representation for each node into the 2D space. The color of a point indicates the class label. The desired embedding layout should be that nodes with the same color (label) to closer each other and different colors (label) to distant each other with the obvious boundary. Due to the space limitation, we only show the visualization result on the BlogCatalog dataset in Fig. 2, and the visualization results on other datasets are similar.

From Fig. 2, we can see that the DANEP, i.e., sub-figure (k), performs the best result with the nodes of the same color are close to each other and the boundaries amongst the different colors are discernible. Besides, DANE, subfigure (e), performs the suboptimal result that the separation of boundaries is inferior to DANEP. Nevertheless, the visualization results of the DeepWalk, Node2Vec, Grarep, DNGR, ANRL, AANE, TADW, NANE and ASNE, i.e., subfigure (a), (b), (c), (d), (f), (g), (h), (i) and (j), are mixed with different color nodes.

4.7 Sensitivity Analysis of Parameters

The hyper-parameters α and β are used to balance the weights between the pairwise constraint loss and reconstruction loss of the DANEP. In this subsection, we analyze the sensitivity of hyper-parameters of DANEP via node classification and node clustering tasks. Experimental results of Micro-F1 of node classification and ACC of node clustering are presented in Fig. 3 and Fig. 4, respectively. The trends of Macro-F1 and NMI with respect to α and β are similar to that of Micro-F1 and ACC, so we do not present them due to the space limitation.

From Fig. 3, we can observe that the tendencies of the Micro-F1 value of node classification are stable under different hyper-parameters and different datasets,

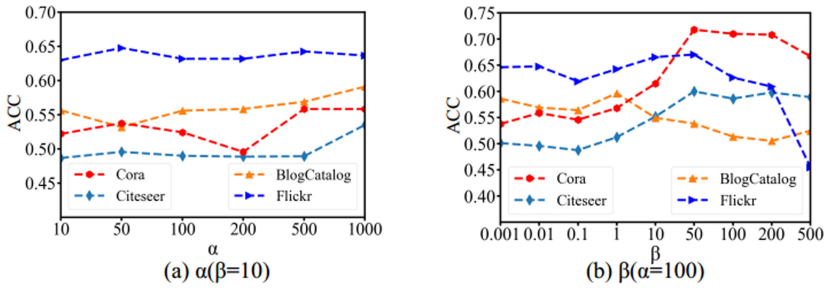

Fig. 4. The sensitivity of DANEP w.r.t. different α and β for node clustering

which indicates DANEP has stable performance for node classification. In Fig. 4, the fluctuation of ACC of node clustering is obvious with the various hyper-parameter β than hyper-parameter α, which indicates that reconstruction loss plays a vital role in the node clustering process.

5 Conclusion

In this study, we develop the DANEP model to integrate attribute features and network topology into a unified graph format and encode each node into a low-dimensional embedding representation. In our model, the k-nearest neighbor graph can reveal some potential non-linear manifold under the attribute features, the random surfing model and PPMI can capture the structural characteristics and high-order proximity information of the attribute/topology graph, and the pairwise constraint can improve the quality of node representation. Experiment results on four real-life datasets in node classification, node clustering and visualization tasks indicated that the performance of the DANEP outperformed 10 representative baselines, including "Topology-only" algorithms and "Topology+Attribute" algorithms.

The DANEP is designed to handle the homogeneous networks with single-typed nodes and edges. However, real-world networks are usually with multiple-typed nodes and edges, which contain richer semantic information and more complex network topology for network representation learning [7,26]. Therefore, extending the DANEP to heterogeneous networks and improving the stability of clustering are our future works.

References

1. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: SODA, pp. 1027–1035 (2007)
2. Cai, H., Zheng, V.W., Chang, K.C.: A comprehensive survey of graph embedding: problems, techniques, and applications. IEEE Trans. Knowl. Data Eng. **30**(9), 1616–1637 (2018)

3. Cao, S., Lu, W., Xu, Q.: GraRep: learning graph representations with global structural information. In: CIKM, pp. 891–900 (2015)
4. Cao, S., Lu, W., Xu, Q.: Deep neural networks for learning graph representations. In: AAAI, pp. 1145–1152 (2016)
5. Dong, K., Zhou, L., Kong, B., Zhou, J.: A dual fusion model for attributed network embedding. In: Li, G., Shen, H.T., Yuan, Y., Wang, X., Liu, H., Zhao, X. (eds.) KSEM 2020, Part I. LNCS (LNAI), vol. 12274, pp. 86–94. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-55130-8_8
6. Gao, H., Huang, H.: Deep attributed network embedding. In: IJCAI, pp. 3364–3370 (2018)
7. Gao, X., Chen, J., Zhan, Z., Yang, S.: Learning heterogeneous information network embeddings via relational triplet network. Neurocomputing **412**, 31–41 (2020)
8. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Knowledge Discovery and Data Mining, pp. 855–864 (2016)
9. Huang, T., Zhou, L., Wang, L., Du, G., Lü, K.: Attributed network embedding with community preservation. In: DSAA, pp. 334–343 (2020)
10. Huang, X., Li, J., Hu, X.: Accelerated attributed network embedding. In: SIAM, pp. 633–641 (2017)
11. Jin, D., Ge, M., Yang, L., He, D., Wang, L., Zhang, W.: Integrative network embedding via deep joint reconstruction. In: IJCAI, pp. 3407–3413 (2018)
12. Liao, L., He, X., Zhang, H., Chua, T.: Attributed social network embedding. IEEE Trans. Knowl. Data Eng. **30**(12), 2257–2270 (2018)
13. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS, pp. 3111–3119 (2013)
14. Mo, J., Gao, N., Zhou, Y., Pei, Y., Wang, J.: NANE: attributed network embedding with local and global information. In: Hacid, H., Cellary, W., Wang, H., Paik, H.-Y., Zhou, R. (eds.) WISE 2018, Part I. LNCS, vol. 11233, pp. 247–261. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02922-7_17
15. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: KDD, pp. 701–710 (2014)
16. Rauber, P.E., Falcão, A.X., Telea, A.C.: Visualizing time-dependent data using dynamic t-SNE. In: EuroVis - Short Papers, pp. 73–77 (2016)
17. Ribeiro, L.F.R., Saverese, P.H.P., Figueiredo, D.R.: struc2vec: learning node representations from structural identity. In: Knowledge Discovery and Data Mining, pp. 385–394 (2017)
18. Ruan, J., Dean, A.K., Zhang, W.: A general co-expression network-based approach to gene expression analysis: comparison and applications. BMC Syst. Biol. **4**, 8 (2010)
19. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: LINE: large-scale information network embedding. In: WWW, pp. 1067–1077 (2015)
20. Tu, C., Liu, H., Liu, Z., Sun, M.: CANE: context-aware network embedding for relation modeling. In: ACL, Volume 1: Long Papers, pp. 1722–1731 (2017)
21. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Knowledge Discovery and Data Mining, pp. 1225–1234 (2016)
22. Wang, Z., Liu, H., Du, Y., Wu, Z., Zhang, X.: Unified embedding model over heterogeneous information network for personalized recommendation. In: IJCAI, pp. 3813–3819 (2019)
23. Weigend, A.S., Rumelhart, D.E., Huberman, B.A.: Generalization by weight-elimination with application to forecasting. In: NIPS, pp. 875–882 (1990)

24. Yang, C., Liu, Z., Zhao, D., Sun, M., Chang, E.Y.: Network representation learning with rich text information. In: IJCAI, pp. 2111–2117 (2015)
25. Yang, Y., Chen, H., Shao, J.: Triplet enhanced autoencoder: model-free discriminative network embedding. In: IJCAI, pp. 5363–5369 (2019)
26. Yu, G., Wang, Y., Wang, J., Domeniconi, C., Guo, M., Zhang, X.: Attributed heterogeneous network fusion via collaborative matrix tri-factorization. Inf. Fusion **63**, 153–165 (2020)
27. Zhang, Z., et al.: ANRL: attributed network representation learning via deep neural networks. In: IJCAI, pp. 3155–3161 (2018)
28. Zhou, L., Lü, K., Yang, P., Wang, L., Kong, B.: An approach for overlapping and hierarchical community detection in social networks based on coalition formation game theory. Expert Syst. Appl. **42**(24), 9634–9646 (2015)
29. Zhou, S., et al.: PRRE: personalized relation ranking embedding for attributed networks. In: CIKM, pp. 823–832 (2018)