24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Dual Attention-based Adversarial Autoencoder for Attributed Network Embedding

Ming Liu and **Jianxin Liao** and **Jingyu Wang** and **Qi Qi** and **Haifeng Sun** [1]

Abstract. Existing embedding methods for Attributed Network aim to learn low-dimensional embeddings for nodes, which can preserve both consistency and complementarity for network structures and node attributes. The main assumption is that nodes with similar structures and/or similar attributes should be close in the embedding space. In reality, nodes with similar attributes might be far away from each other in topology and vice versa. The conflict is often caused by noisy links or incomplete network structures. Previous methods either independently project embeddings based on the assumption without considering the conflicts, or encode embeddings into a shared space ignoring the complementarity. In this paper, we propose a Dual Attention-based Adversarial Attributed Network Embedding framework (DAANE) to preserve the consistency and complementarity between structures and attributes, and reduce the conflict caused by their discrepancy. DAANE includes an attribute attention mechanism designed to detect and weakening the impact of noisy links and a structure attention mechanism applied to assign weights to network structures of different scales and capture a more complete global context. Furthermore, we develop efficient adversarial learning when combining the two heterogeneous embeddings. The adversarial auto-encoder projects embeddings of attributes and structures into the same space. Meanwhile, it completely circumvents the interference of various types of noise by removing the constraints of embedding space. Extensive experiments on three real-world network datasets indicate that the proposed model achieves state-of-the-art results.

1 Introduction

The performance of machine learning methods relies heavily on the choice of data representation or embeddings. Based on the fact, many actual efforts in deploying machine learning algorithms go into the design of learning transformations of the data, which makes it easier to extract useful information for the down-stream classifiers or other tasks [2]. The extracted data is usually a single image or a description of a single item; that is, the algorithm explores complex patterns of the data from an individual perspective. In reality, there are interactive information exists between the data. For example, social networks have not only personal attribute information but also a large number of comments and concerns. The interactions provide many additional relationships underlying the original data. However, it is difficult to extract this part of the information directly because they exist in the form of graph. Therefore, how to combine the graph data with the original data has attracted considerable research attention in the past few years [24, 34, 35].

In order to organize such information with the complex graph, one of effective means learn the attributed graph embeddings, which assigns each node a low-dimensional dense vector via leveraging the structure and/or the attribute information of the graph. A number of attributed network embedding algorithms have been proposed to preserve consistency and complementarity between the two heterogeneous information[17, 20, 23, 37]. In this way, the learned embeddings of nodes in attributed networks in turn help to enhance the performance of many down-stream applications [3, 32, 30].

Despite the strong task performance, existing methods have the following limitations: (1) the conflict between two heterogeneous information. It indicates that nodes with similar structures might be dissimilar in attributes and vice versa, which often caused by the noise/abnormality and incompleteness of the graph data. For example, in telecommunications networks, there are many opposing structural connections, such as fraudsters and victims. Although their attributes are quite different, they still have a relatively close distance in the embedded space because of the preservation of the first-order structural proximity. On the other hand, people's limited energies prevent the appearance of many similar pairs. That is to say, the structure of the graph we observed is not complete, and many close pairs on the property are not directly connected in the graph. For instance, a new Twitter user might follow some well-known movie stars rather than those users with similar tastes. However, most previous methods make efforts to preserve the proximity of both structures and attributes in the representation vector, ignoring conflicts caused by the noise and incomplete graph data. (2) Their combination of the two representations cannot preserve consistency and complementarity simultaneously. Some methods [16, 21, 12] use a shared coding layer to project embeddings into the shared space, which make the learned representations focus more on the consistency. Others [7, 11, 1] extract representations independently from the respective modality to keep the complementarity, and then use some constraints to capture the consistency between the modalities. In the attributed graph, the constraint generally represents whether the corresponding node is the same or has a direct connection. In this case, the conflict of the previous analysis can still affect the preservation of consistency.

To address the challenges mentioned above, we introduce a Dual Attention-based Adversarial Attributed Network Embedding framework, abbreviated as DAANE, which aims at learning low-dimensional vector embeddings of both attributes and structures in the shared space such that the conflicts between them can be effectively captured and alleviated. It includes both an attribute attention auto-encoder and a structural attention auto-encoder. Specifically, the attribute attention auto-encoder is used to process pairs of nodes that

[1] State key laboratory of Networking and Switching Technology, email: [liuming][liaojianxin][wangjingyu][qiqi][sunhaifeng]@ebupt.com Beijing University of Posts and Telecommunication of China

have direct structural connections but large attributes differences. Instead of setting a threshold to determine the magnitude of the difference, we adopt the idea of anomaly detection to find noise or opposite edges adaptively. For each node to be analyzed, the nodes directly connected to it constitute its neighbor set. When the attribute of a neighbor is far from other nodes, the algorithm will appropriately reduce its weight to reduce the impact of such conflicts. Meanwhile, in order to overcome the incompleteness of the graph data, a structural attention auto-encoder is proposed to exploit the global structure proximity. It calculates the neighbor vectors at different scales and assigns different weights to the first-order to high-order neighbors. Furthermore, we take the advantages of the combination and introduce adversarial learning to project heterogeneous information into the common space, which provides a simple but effective solution to learn the consistent and complementary representations simultaneously.

Our contributions in this work can be summarized as follows:

- We propose DAANE, a framework to embed attributed network considering the inconsistent similarity of attributes and structures. Two attention auto-encoders are proposed to adaptively alleviate heterogeneous information conflicts due to noise and incompleteness of the graph data.
- We propose an adversarial learning regularization to project heterogeneous embeddings into the common space, which provides an effective solution to learn the consistent and complementary representation simultaneously.
- We carry out extensive experiments on various real-world datasets. The results show that DAANE significantly outperforms state-of-the-art methods in terms of both node classification, link prediction, and network visualization.

2 Related Work

There are two lines of works related to our model, network embedding and adversarial learning.

2.1 Network Embedding

Network embedding aims to learn the distributed representation vectors for each node in a network. DeepWalk [27] employs a truncated random walk to generate node sequences, which is treated as sentences in language models and fed to the Skip-gram model to learn embeddings. Based on it, Node2Vec [15] adds flexibility in exploiting neighborhoods and designs a biased random walk procedure. LINE [30] is proposed for a large scale network, which preserves both the first-order and second-order proximities to learn network embeddings. DNGR [5] incorporates a random surfing model into deep the auto-encoder to preserve the high non-linearity.

Because numerous networks are often associated with abundant node attributes, attributed network embedding is proposed to learn from node links and attributes jointly. TADW [37] extends DeepWalk by using textual attributes to supervise random walks in a matrix factorization framework. AANE [20] employs the graph Laplacian technique to learn joint embeddings from the topological structure and attributes. GAE [22] first proposes a graph convolutional neural network model(GCN) [23] for attributed networks representation, and further employ it on a variational auto-encoder architecture VGAE [22]. GraphAE [6] proposes a graph auto-encoder that uses GCN in both encoder and decoder. These methods use gcn as the encoder, which faces the problem of over-smoothing and large computational complexity when capturing the structural proximity of long distances. Besides, there are many algorithms that do not use GCN to integrate the structure and attribute information in the graph. ANRL [38] and DANE [11] both design a customized deep model and capture underlying high non-linearity in both topological structures and node attributes. Their results showed that combining different types of auxiliary information, rather than using only the structures to aggregate attributes, can provide different insights of embeddings, and help to capture rich patterns in many real-world networks. However, these methods essentially assume that all proximity should be preserved, which ignores conflicts between the two heterogeneous information. This leads to the learned embeddings that do not reflect the distance between nodes.

2.2 Adversarial Learning

Our work is also related to adversarial learning. Generative adversarial networks(GANs) [14] has achieved great success in many areas [9, 29, 28], which inspired us to investigate network embedding using GANs. ANE [36] and AIDW [8] introduce an adversarial learning mechanism into the auto-encoder and random walk, respectively. The basic idea is that unconstrained embedding spaces are more susceptible to noise interference. AGRE and its variants AGVRE[25] propose an adversarial framework for the attributed network. As before, they use adversarial training to make the embedding space obey the prior distribution to learn more robust node representation. None of the methods above that apply adversarial learning to handle the problem of embedding space shift.

3 Method

3.1 Definition

We define an attributed network as $G = \{V, E, X\}$, where $V = \{v_i\}_{i=1}^{n}$ denotes the node set with size n, and $E \in V \times V$ denotes the edge set. The network is represented by an adjacency matrix A, where $A_{ij} = 1$ if $(v_i, v_j) \in E$ and $A_{ij} = 0$ otherwise. Attributes of nodes in the network are represented by an attribute matrix $X \in \mathbf{R}^{n \times F}$, where F is the dimension of node attributes. X_i is the i-th row of X and represents the attribute vector of node v_i. Attributed network embedding aims to learn low-dimensional representations $Z \in \mathbf{R}^{n \times k}$ from adjacency matrix A and attribute matrix X, such that the learned representations can preserve both network structure and node attributes.

In this paper, we propose a dual attention-based adversarial auto-encoder framework for attributed network embedding. The framework consists of three main parts, the attribute auto-encoder, the structure auto-encoder, and the adversarial component, as shown in Fig. 1.

3.2 Attribute Attention Autoencoder

We use attribute encoders to discover and alternate the effects of noise links. In the attributed network, there are many pairs of nodes that are connected to each other and have different attributes. Some of these are true noise links, while the other is caused by the complementarity between heterogeneous information. Judging the noise based on the attribute distance of a pair of node pairs will lose complementarity. Therefore, we detect noisy links from multiple links of a node. When the relationship between two nodes on one edge is different from the other edges in the subgraph, we can detect that this is a noise edge and weaken its weight.

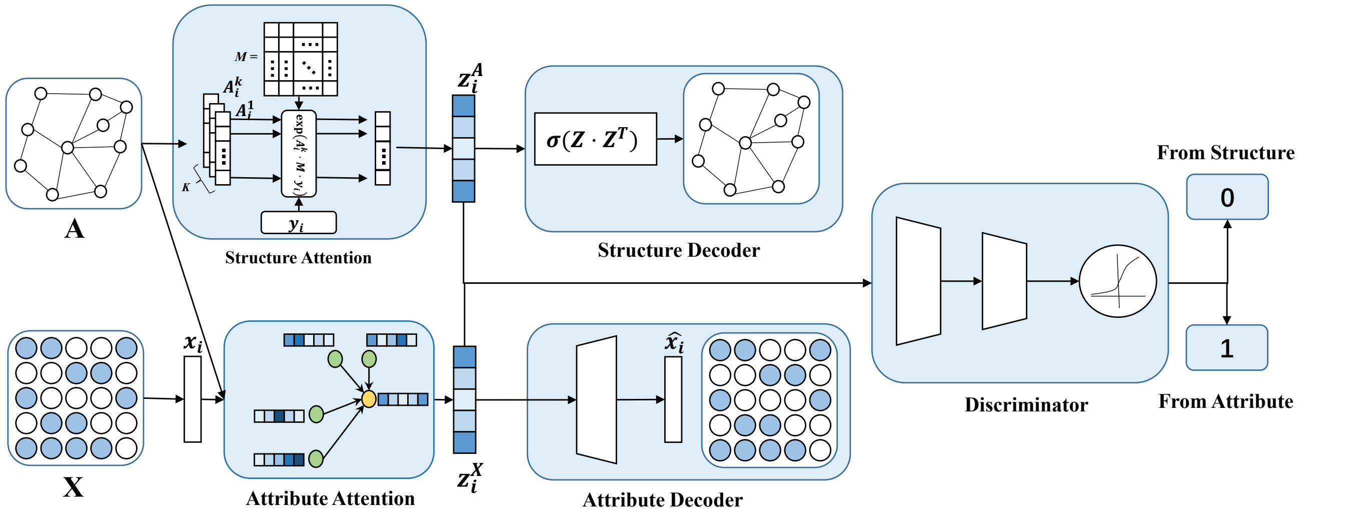


Figure 1. Architecture of DAANE. The encoder uses two attention mechanisms to jointly encode the important information in the graph structure A and the attribute matrix X into embeddings Z^A and Z^X. The decoder reconstructs the network structure and node attributes from embeddings Z^A and Z^X. The right part is the adversarial training. A discriminator network is trained to predict whether a sample arises from the structure embedding or an attribute embedding.

We develop a variant of the graph convolutional network (GCN) as an attribute encoder. GCN encoder consists of a stack of single encoder layers, each of which aggregates the attribute information from the neighboring nodes of a target node. By stacking multiple encoder layers, GCN encoder is able to aggregate the attribute information from the multi-hop ego-network of the target node, which is taken as the target node's attributed local subgraph. Given the input x_i, a single GCN layer can be formalized as follows:

$$z_i^X = \sigma(\sum_{j \in N_i} \alpha_{ij} W x_i), \tag{1}$$

where σ represents the nonlinear activation function. Weight matrix $W \in R^{F*F'}$ uses a linear transformation to map inputs to higher-level features. N_i is the set of node i's neighbors. α_{ij} is the aggregation weight and measures how important node j is to node i.

In GCN encoder model, the neighborhoods of nodes are aggregated with equal. We use the same attention mechanism as in GAT. The comparison process is normalized via softmax as follows:

$$a_{ij} = \frac{\exp(score(z_i^X, z_j^X))}{\sum_{t \in N_i} \exp(score(z_i^X, z_t^X))}, \tag{2}$$

where $score(\cdot)$ is a alignment function which measures the relationship between source node v_j and the target node v_i. MLP has sufficient capacity to approximate any arbitrary function and can be trained to learn deeper connections within the data. We compute $score(z_i^X, z_j^X)$ via the following feed-forward network with a single hidden layer (MLP).

$$score(z_i^X, z_j^X) = \sigma(\mathbf{W}_a^T[z_i^X | z_j^X] + \mathbf{b_a}), \tag{3}$$

where W_a is the weight matrix for the concatenation of z_i^X and z_j^X, b_a is the bias vector.

The attribute decoder maps the embedding z_i^X in embedding space to reconstruction space. The goal of the attribute decoder is to minimize the reconstruction error of as follows:

$$L_a = \sum_{i=1}^{n} ||\widehat{x}_i - x_i||_2^2, \tag{4}$$

where $\widehat{x}_i$ is the reconstructed data point from the attribute decoder, which is transformed from z_i^X by the projection matrix W_f:

$$\widehat{x}_i = W_f z_i^X. \tag{5}$$

3.3 Structure Autoencoder

After obtaining attribute embedding $h_i^{(T)}$, another challenge is how to save the structural information to the embedding vector as well. The most frequently used method is to add a structural proximity loss function. The function imposes a penalty on the pair of vertices that have high structure similarity but mapped far away in the latent representation space. However, there are various scales of proximity in the network structure. Among them, the first-order proximity of two nodes i and j is determined by A_{ij}. Specifically, a larger A_{ij} denotes larger proximity between the nodes i and j. The second-order proximity indicates the similarity of neighborhood structures between two nodes, which means the more common neighborhood shared by a pair of vertices, the more similar they are. GraRep [4] extents the second-order proximity to high-order proximity. The high-order proximity is determined by the high-order proximity matrix $P = A + A^2 + \cdots + A^k$, where $A^k = \prod_1^k A$, and the entry refers to the k-order proximity between vertices v_i and v_j.

High-order proximity can capture the long-distance relationship between two different vertices, but it ignores that the importance of different scales can be quite different. DNGR [5] links the weight distribution to the distance, and the farther the distance is, the smaller the weight is. Despite their good performance, the weight of each scale depends on artificial adjustment, which is not adaptive enough in the face of different network structures. We design a structure attention model to infer the weights of different scales. For vertex v_i, scale vector A_i^k is the i-th column in each A^k, for $k = 1, 2 \cdots K$, which denotes the k-th scale structure information. We aim to learn a representation e_i for node v_i, which can combine the most informative structural information from various scales. e_i is calculated by a weighted summation of every scale vector as follows:

$$z_i^A = \sum_{k=1}^{K} \alpha_k A_i^k. \tag{6}$$

For each scale A_i^k, we compute a positive weight α_k, which reflects the contribution of the different scales in deciding the structure proximity. The weight is calculated by an attention model as follows:

$$\begin{aligned} \alpha_k &= \frac{\exp(d_k)}{\sum_{j=1}^{K} \exp(d_j)}, \\ d_k &= A_i^k \cdot M \cdot y_i, \\ y_i &= \frac{1}{K} \sum_{k=1}^{K} A_i^k, \end{aligned} \tag{7}$$

where y_i is the average of different scale vector, which can capture the global context of the structure information. $M \in \mathrm{R}^{|V| \times |V|}$ is a matrix mapping between the global context embedding y_i and each scale vector A_i^k, which can be learned during the training process [10, 18]. The weight α_k will be affected by many aspects: the structure vector at the current scale, the more structural vectors appearing in the global, and the relationship between the vector dimensions. In our scenario, the relationship between vector dimensions is the correlation between nodes, which is learned by M through training.

Our structure decoder predicts whether there is a link between two nodes. Specifically, we train a link prediction layer based on the graph embedding:

$$L_s = \sum_{i}^{n} \sum_{j}^{n} ||A_{ij} - \mathrm{sigmoid}((\mathrm{z_i^A})^\mathrm{T}, \mathrm{z_j^A})||_2^2. \tag{8}$$

The objective function of the auto-encoder is formulated as the weighted combination of L_a and L_s as follow:

$$L_{ae} = L_a + L_s. \tag{9}$$

3.4 Adeversarial Learning

The combination of representations between different modalities needs to preserve complementarity and consistency. A widely used method is the joint representation that different modalities are projected into the same space, which is $Z^A = Z^X$. This method can guarantee the consistency between two modalities but will lose too much complementary information from two modalities due to the exact same encoding layer. DANE oppositely joint representations with coordinated representations where some constraints between the modalities force the representations to be more complementary. Their constraints can aim at maximizing the correlation between the representations Z^A and Z^X. However, the reliability of the constraint itself has a huge impact on the embedded performance. The constraint in DANE is designed to push Z^A and Z^X together when they are from the same or connected nodes while pushing them away when two nodes are not connected. This constraint reliability, like our previous analysis, is severely affected by the noise and incompleteness of the graph data.

We borrow from both visions, namely the joint representations and the coordinated representations. Our combine method builds on two auto-encoder designed to process each modality independently, which can preserve the complementarity. Instead of adding constraints, adversarial learning is introduced to force the two representations into the same common space, while consistency can also be preserved.

The adversarial model acts as a discriminator(D) to distinguish whether a latent representation is from the attributes(h_a) or the structure(h_s). Discriminator D is a multi-layer perceptron having no activation function in the final layer. We expect D to output 0 when the input vector is sampled from attributes and output 1 otherwise. So the discriminator's loss function of is:

$$L_D = \mathbf{E}_{x \in H_A}[(D(x) - 0)^2] + \mathbf{E}_{x \in H_S}[(D(x) - 1)^2], \tag{10}$$

where $D(x)$ is the output of the discriminator. In GAN, on the contrary, the generator confuses the discriminator by forcing the distribution of fake samples to approximate that of real samples. However, our goal is to learn a common space rather than forcing a representation to obey another distribution. In order to ensure the symmetry of our architecture and loss function, we design following bidirectional adversarial training loss function:

$$L_{adv} = \mathbf{E}_{x \in H_A}[(D(x) - 1)^2] + \mathbf{E}_{x \in H_S}[(D(x) - 0)^2]. \tag{11}$$

We combine the training of two autoencoder and adversarial learning regularization together by defining the overall loss function as follows:

$$L = L_{ae} + \lambda L_{adv}, \tag{12}$$

where λ is a hyperparameter to control the weight of regularization. In this paper, we set λ as 1. Model training is mainly divided into two phases. In the reconstruction phase, the autoencoder updates the encoder and decoder to minimize the L. In the regularization phase, we train the discriminator to optimize L_D. The final embedding is formulated as:

$$z_i = z_i^X + z_i^A. \tag{13}$$

4 Experiments

4.1 Datasets

In our experiments, we employ three benchmark datasets: Cora, Citeseer, and Wiki. Cora and Citeseer are citation networks where nodes are articles, and edges indicate citations between articles. In these datasets, citation relationships are viewed as undirected edges for simplicity. Attributes associated with nodes are extracted from the title and the abstract of each article and are represented as sparse bag-of-words vectors. Wiki is a network with nodes as web pages. The link among different nodes is the hyperlink on the web page. The text information on the web pages is processed similarly as in the other datasets to extract the attributes. We summarize the statistics of these benchmark datasets in Tab. 1.

Table 1. Detailed information of the three datasets

Dataset	Nodes	Edges	Attributes	Classes
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6
Wiki	2405	17891	4973	19

Baseline Methods: We compare our model with the following baselines at both node classification and link prediction tasks. All the baselines fall into three categories, namely "Attributes-only," "Structure-only," and "Attributes+Structure." Models in "Attributes-only" leverage node attribute information only to extract node representation, from which we select SVD and auto-encoder as our baselines. "Structure-only" models consider structure information only, i.e., preserving structural proximity in embedding space, while ignoring attribute information. In this group, we choose Deepwalk, SDNE as our baselines. Methods in the "Attribute+Structure" group capture both nodes attributes and structure proximity simultaneously, and we consider several recent state-of-the-art algorithms as our baselines. A detailed description of our baselines is illustrated as follows:

- **Auto-encoder(AE)** [19] is the conventional auto-encoder model with nodes attributes as input only. The number of hidden units is set the same as our model.
- **Singular Value Decomposition (SVD)** [13] is a linear model that can extract node representations by decomposing the node attribute matrix. The largest 200 eigenvalues are kept when performing SVD.
- **DeepWalk(DW)** [27] learns embeddings using structural information only. DeepWalk learns the node embedding from a collection of random walks using skip-gram with hierarchical softmax. As for the parameters, the number of random walks is 10, the number of vertex per walk $\gamma = 80$, window size $t = 10$, and embedding dimension $k = 128$.
- **SDNE** [33] is a deep model that capture both first-order and second-order structure proximity in embedding. The hidden units in SDNE are set the same as our model, and the hyper-parameters of α, β, and v are tuned by using grid search on the validation set.
- **DW+** concatenates Deepwalk embedding with SVD.
- **TADW** [37] is an approach that utilizes both structure and text information to learn embeddings. We set the coefficient of regularization term to be 0.1.
- **GAE** [22] use graph convolutional neural network as encoder and the representation is applied to reconstruct the network structure.
- **GraphAE** [6] uses the GCN in both encoder and decoder. Their encoder aggregates the information to the target node's representation, and the decoder propagates the information to each source node to reconstruct the attributes.
- **ARGE** [25] Adversarially Regularized Graph Autoencoder applies the adversarial mechanism to attributed network. The learned representation is enforced to match a prior distribution via an adversarial training scheme. We set the prior distribution to a Gaussian distribution according to their paper.
- **DANE** [11] use two independent auto-encoder to model attributes and structure information and several regularizations in hidden representation to preserve various proximities.

4.2 Node Classification

In our model, the dimension of the embedding is set to 128. After having obtained the node embedding, several classifiers in machine learning can deal with this task. We train a one-vs-rest classifier for each class and select the classes with maximum scores in Logistic Regression. We randomly sample 10% and 50% labeled nodes to train, and the rest of the nodes are used to test performances. We repeat this process ten times and report the average performance in terms of Micro-F1 and Macro-F1. Tab 2 shows the classification performance with different training ratios on different datasets, where the best results are **bold-faced**. From these tables, we have the following observations:

- Methods in the Network-Only group can achieve good results in Cora while performing worse on the other datasets. The nature of the datasets can explain this phenomenon. Documents in both Citeseer and Wiki have more words, so "Attribute-only" methods perform better. Cora has more structural connections and sparse attribute information, "Structure-only" methods get a better result in this dataset.
- Most models considered by topology and attributes have achieved better results than single information. It demonstrates that in attributed network embedding, preserving proximity from both information sources can learn better latent embeddings of nodes. Specifically, directly concatenate these two kinds of information may improve the performance but not obvious, as "DW+" gets worse node classification result than SVD in and Wiki dataset. Therefore, simple concatenation is not sufficient to capture the consistency and complementarity between these two types of information.
- Among methods using both node attribute and topology information, our model outperforms all the other baselines on all three datasets. Besides, our model achieves a more stable performance when there is a small amount of label information. Our model alleviates the conflicts between modalities and captures the consistency and complementarity in a common space, thus improve the robustness and discrimination of the learned embedding.

4.3 Link Prediction

In this section, we evaluate the ability of learned embeddings to reconstruct networks and predict future connections via link prediction. We split the edges in the network according to the ratio of 85%, 5% and 10% as positive instances for training, validation, and testing, respectively. After having obtained the embeddings for each node, and the predicted probability of a link can be achieved directly from the inner product of the embeddings between two nodes. We adopt the area under the ROC-curve (AUC) and average precision from prediction scores (AP) as the evaluation metrics. Higher values of AUC and AP indicate better performance. Tab. 3 shows the link prediction performance of our CAN and the baselines on the three attributed networks. We bold the best results and underline the next best results. As shown, our CAN consistently performs better than any of the baseline model. This is mainly because our structure auto-encoder optimizes a loss function consisting of the reconstruction error of all the edges and capture the global context of the structure information.

4.4 Ablation analysis:

To comprehensively analyze the performance of different components of our model, we compare DAANE with its variants to indicate the importance of DAANE's unique design.

4.4.1 Influence of attention mechanism

To prove that our method can alleviate the conflicts between the modalities, we compare our model and its three variants. First, we removed the attention mechanism in the structure auto-encoder and only encoded the first-order structure proximity. We mark this baseline as "-structure." Then, we replace the attribute attention mechanism with the average weight while keeping the multi-scale structure attention. This baseline is named "-attribute." Moreover, we replace all attention models, where only the first-order neighbor's attribute is equally aggregated to the latent vector. We mark this baseline as

Table 2. Results of Node Classification.

Dataset	Cora				Citeseer				Wiki			
Training ratio	10%		50%		10%		50%		10%		50%	
Method	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1
AE	0.678	0.642	0.735	0.702	0.688	0.592	0.705	0.622	0.533	0.416	0.689	0.597
SVD	0.483	0.351	0.685	0.667	0.613	0.531	0.695	0.627	0.654	0.502	0.771	0.667
DeepWalk	0.763	0.745	0.813	0.792	0.503	0.465	0.575	0.525	0.573	0.425	0.674	0.552
SDNE	0.767	0.751	0.821	0.790	0.537	0.491	0.581	0.530	0.537	0.431	0.611	0.523
DW+	0.760	0.746	0.839	0.821	0.530	0.486	0.679	0.6211	0.640	0.486	0.764	0.641
TADW	0.803	0.787	0.856	0.845	0.673	0.607	0.739	0.705	0.675	0.507	0.797	0.672
GAE	0.804	0.793	0.823	0.814	0.605	0.523	0.620	0.534	0.688	0.500	0.725	0.559
GraphAE	0.811	0.789	0.861	0.848	0.681	0.639	0.743	0.712	0.710	0.569	0.803	0.698
ARGE	0.783	0.769	0.843	0.808	0.662	0.609	0.721	0.683	0.650	0.469	0.683	0.578
DANE	0.798	0.785	0.865	0.847	0.638	0.595	0.725	0.687	0.736	0.598	0.785	0.677
DAANE	**0.834**	**0.819**	**0.878**	**0.861**	**0.704**	**0.659**	**0.768**	**0.731**	**0.741**	**0.636**	**0.808**	**0.703**

Table 3. Results of link prediction.

Dataset	Cora		Citeseer		Wiki	
	AUC	AP	AUC	AP	AUC	AP
AE	79.14	79.40	81.39	82.08	77.26	82.08
SVD	79.1	82.46	85.82	88.76	83.73	87.98
DeepWalk	80.53	82.79	73.22	76.21	81.27	82.39
SDNE	77.87	81.95	74.46	78.91	81.68	82.66
DW+	81.06	83.13	73.92	76.72	89.57	91.16
TADW	93.01	93.95	94.51	95.67	92.19	93.11
GAE	91.47	92.37	90.52	91.59	91.81	92.91
GraphAE	89.9	88.77	93.51	94.61	88.87	88.75
ARGE	91.7	92.64	90.96	92.97	91.17	92.49
DANE	88.19	89.56	84.93	84.68	91.01	92.45
DAANE	**95.67**	**96.86**	**96.11**	**96.23**	**93.41**	**94.83**

"-attention." For a fair comparison, all variant models use adversarial training when combining representation. The experiment setup is the same as section 4.2. Limited by space, we only display the macro-f1 for node classification.

As shown in the Tab. 4, DAANE performs better than all variants. At the same time, variants that use attention mechanisms have more or fewer improvements on all datasets than "-attention." This shows that our model can effectively reduce conflicts. Another phenomenon is that "-structure" performs better than "-attribute" in Cora and poorer in Citeseer and Wiki. This may be because Cora has more network links, and have fewer papers with similar themes but no citations. Therefore, weakening the noise link will increase the performance of the model. The Citeseer and Wiki have relatively few network connections. At this point, fixing incomplete graph data can result in higher performance.

Table 4. Ablation analysis of attention mechanism

	Cora		Citeseer		Wiki	
Method	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1
DAANE	0.878	0.861	0.768	0.731	0.808	0.703
-structure	0.862	0.849	0.742	0.712	0.778	0.671
-attribute	0.848	0.835	0.753	0.721	0.748	0.630
-attention	0.831	0.818	0.711	0.632	0.733	0.619

Table 5. Ablation analysis of adversarial learning

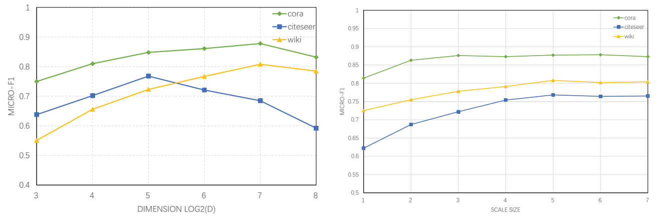
	Cora		Citeseer		Wiki	
Method	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1
concatenate	0.825	0.819	0.713	0.634	0.708	0.621
joint	0.838	0.832	0.733	0.701	0.783	0.681
coordinated	0.863	0.841	0.737	0.698	0.772	0.669
DAANE	0.878	0.861	0.768	0.731	0.808	0.703

4.4.2 Influence of adversarial learning

To inspect the effectiveness of adversarial regularization, we compare our model to the three ways of combining representation. The first is a simple concatenate of the two representations. The second is the joint representation that makes the two representations share the same layer. The third is a coordinated representation; we combine the representations in the same way as in DANE. The experiment setup is the same as section 4.2, and we display the macro-f1 for node classification. As shown in the Tab. 5, the simple concatenate method gets the poorest performance, and our DAANE achieves the best performance on three datasets. Compared to the other two merge methods, the performance of our model has a wide range of improvements in each data set. It demonstrates that our learned embedding can draw on their respective advantages and preserve more consistency and complementarity.

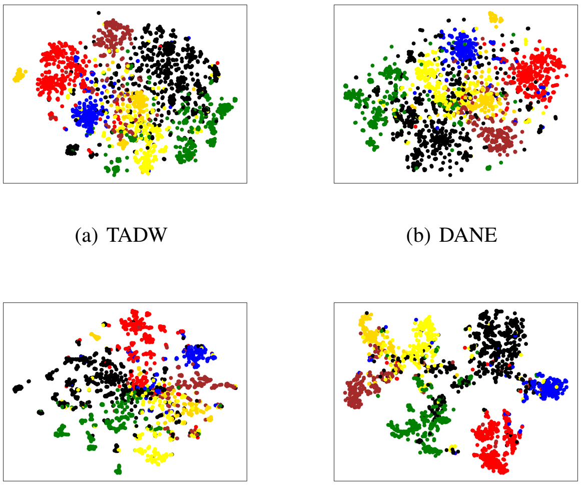
4.5 Parameter Sensitivity

We discuss the parameter sensitivity in this section. Specifically, we explore how the different choices of the maximal scale size K, dimension d can affect node classification with the training ratio as 50%. We vary dimension of embedding from [8, 16, 32, 64, 128, 256]. As shown in Figure2, the trend of the curves in three datasets is very similar-performance increase when dimension gets larger at first and decreases when the size of embedding larger than a specific value. From the experimental results, we can find that our model is somewhat sensitive to dimension. Figure 5(b) shows the Micro-F1 scores over different choices of scale K. We can observe that as K increases, it can provide more useful complementary information. When K is large enough, learned k-order relational information becomes weak and shifts towards a steady distribution.



(a) Dimension (b) Scales

Figure 2. Parameter sensitivity of dimension d and scale size k.



(a) TADW (b) DANE

(c) ARGE (d) DAANE

Figure 3. Visualization of network representations learned by different algorithms on the Cora. Each point indicates one node and each color represents one category.

4.6 Network Visualization

Visualization is another way to demonstrate the effectiveness of learned representation. A good embedding algorithm should be able to distinguish nodes of different labels by separating them in the embedding space. Following [26] we first learn a low dimensional $k = 128$ embedding for each node and then map those embedding in 2D with T-SNE [31]. Fig. 3 shows the visualization of Cora. It can be clearly seen that DAANE obtains a higher quality vector. The embedding we learned is more compact than the other methods. Nodes of the same category in DAANE are gathered together, and the interval between different categories is considerable.

5 Conclusion

In this paper, we present an attributed network embedding method with consideration of consistency, complementarity, and conflicts between different information. The proposed dual attention model can capture the proximity of attributes and structures and alleviate the conflicts caused by noisy and incomplete graph data. Moreover, an adversarial regularization learning is introduced as a simple but effective solution to learn consistent and complementary representations. Experiments on three real-world datasets demonstrate our superior performance in learning network embedding. As a part of future work, we plan to investigate the adversarial model in a dynamic attributed network.

6 Acknowledgement

This work was supported in part by the National Key R&D Program of China 2018YFB1800502, in part by the National Natural Science Foundation of China under Grants 61671079 and 61771068, in part by the Beijing Municipal Natural Science Foundation under Grant 4182041, and in part by the Ministry of Education and China Mobile Joint Fund MCM20180101.

REFERENCES

[1] Sambaran Bandyopadhyay, N. Lokesh, and M. Narasimha Murty, ‘Outlier aware network embedding for attributed networks’, *CoRR*, **abs/1811.07609**, (2018).

[2] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent, ‘Representation learning: A review and new perspectives’, *IEEE Trans. Pattern Anal. Mach. Intell.*, **35**(8), 1798–1828, (2013).

[3] Smriti Bhagat, Graham Cormode, and S. Muthukrishnan, ‘Node classification in social networks’, in *Social Network Data Analytics*, 115–148, (2011).

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu, ‘Grarep: Learning graph representations with global structural information’, in *CIKM*, pp. 891–900, (2015).

[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu, ‘Deep neural networks for learning graph representations’, in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pp. 1145–1152, (2016).

[6] Keting Cen, Huawei Shen, Jinhua Gao, Qi Cao, Bingbing Xu, and Xueqi Cheng, ‘A graph auto-encoder for attributed network embedding’, *CoRR*, **abs/1906.08745**, (2019).

[7] Sarath Chandar, Mitesh M. Khapra, Hugo Larochelle, and Balaraman Ravindran, ‘Correlational neural networks’, *CoRR*, **abs/1504.07225**, (2015).

[8] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang, ‘Adversarial network embedding’, in *AAAI*, (2018).

[9] Ming Ding, Jie Tang, and Jie Zhang, ‘Semi-supervised learning on graphs with generative adversarial nets’, in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pp. 913–922, (2018).

[10] Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou, ‘Attentive pooling networks’, *CoRR*, **abs/1602.03609**, (2016).

[11] Hongchang Gao and Heng Huang, ‘Deep attributed network embedding’, in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pp. 3364–3370, (2018).

[12] Hongchang Gao, Feiping Nie, Xuelong Li, and Heng Huang, ‘Multi-view subspace clustering’, in *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 4238–4246, (2015).

[13] Gene H Golub and Christian Reinsch, ‘Singular value decomposition and least squares solutions’, in *Linear Algebra*, Springer, (1971).

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, ‘Generative adversarial nets’, in *Advances in neural information processing systems*, pp. 2672–2680, (2014).

[15] Aditya Grover and Jure Leskovec, ‘node2vec: Scalable feature learning for networks’, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 855–864, (2016).

[16] Zepeng Gu, Bo Lang, Tongyu Yue, and Lei Huang, ‘Learning joint multimodal representation based on multi-fusion deep neural networks’, in *Neural Information Processing - 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II*, pp. 276–285, (2017).

[17] William L. Hamilton, Zhitao Ying, and Jure Leskovec, 'Inductive representation learning on large graphs', in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 1024–1034, (2017).

[18] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier, 'An unsupervised neural attention model for aspect extraction', in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 388–397, (2017).

[19] Geoffrey E Hinton and Ruslan R Salakhutdinov, 'Reducing the dimensionality of data with neural networks', *science*, **313**(5786), 504–507, (2006).

[20] Xiao Huang, Jundong Li, and Xia Hu, 'Accelerated attributed network embedding', in *Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, Texas, USA, April 27-29, 2017.*, pp. 633–641, (2017).

[21] Miao Kang, Kefeng Ji, Xiangguang Leng, and Zhao Lin, 'Contextual region-based convolutional neural network with multilayer fusion for SAR ship detection', *Remote Sensing*, **9**(8), 860, (2017).

[22] Thomas N. Kipf and Max Welling, 'Variational graph auto-encoders', *CoRR*, **abs/1611.07308**, (2016).

[23] Thomas N. Kipf and Max Welling, 'Semi-supervised classification with graph convolutional networks', in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, (2017).

[24] Ming Liu, Jianxin Liao, Jingyu Wang, and Qi Qi, 'AGRM: attention-based graph representation model for telecom fraud detection', in *2019 IEEE International Conference on Communications, ICC 2019, Shanghai, China, May 20-24, 2019*, pp. 1–6, (2019).

[25] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang, 'Adversarially regularized graph autoencoder for graph embedding', *arXiv preprint arXiv:1802.04407*, (2018).

[26] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang, 'Tri-party deep network representation', in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 1895–1901, (2016).

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena, 'Deepwalk: online learning of social representations', in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pp. 701–710, (2014).

[28] Alec Radford, Luke Metz, and Soumith Chintala, 'Unsupervised representation learning with deep convolutional generative adversarial networks', *arXiv preprint arXiv:1511.06434*, (2015).

[29] Sai Rajeswar, Sandeep Subramanian, Francis Dutil, Christopher Pal, and Aaron Courville, 'Adversarial generation of natural language', *arXiv preprint arXiv:1705.10929*, (2017).

[30] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei, 'LINE: large-scale information network embedding', in *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pp. 1067–1077, (2015).

[31] Laurens van der Maaten and Geoffrey Hinton, 'Visualizing data using t-SNE', *Journal of Machine Learning Research*, **9**, 2579–2605, (2008).

[32] Chun Wang, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang, 'MGAE: marginalized graph autoencoder for graph clustering', in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pp. 889–898, (2017).

[33] Daixin Wang, Peng Cui, and Wenwu Zhu, 'Structural deep network embedding', in *SIGKDD*, pp. 1225–1234. ACM, (2016).

[34] Zhongdao Wang, Liang Zheng, Yali Li, and Shengjin Wang, 'Linkage based face clustering via graph convolution network', in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 1117–1125, (2019).

[35] Libing Wu, Cong Quan, Chenliang Li, Qian Wang, Bolong Zheng, and Xiangyang Luo, 'A context-aware user-item representation learning for item recommendation', *ACM Trans. Inf. Syst.*, **37**(2), (2019).

[36] Yang Xiao, Ding Xiao, Binbin Hu, and Chuan Shi, 'Ane: Network embedding via adversarial autoencoders', in *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, (2018).

[37] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang, 'Network representation learning with rich text information', in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pp. 2111–2117, (2015).

[38] Zhen Zhang, Hongxia Yang, Jiajun Bu, Sheng Zhou, Pinggang Yu, Jianwei Zhang, Martin Ester, and Can Wang, 'ANRL: attributed network representation learning via deep neural networks', in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, (2018).