DANE: Domain Adaptive Network Embedding

Yizhou Zhang$^1$, Guojie Song$^{1,2*}$, Lun Du$^{1,2}$, Shuwen Yang$^1$ and Yilun Jin$^1$

$^1$School of Electronic Engineering and Computer Science, Peking University
$^2$Key Laboratory of Machine Perception (Ministry of Education), Peking University

\{zhangyizhou2015, dulun, gisong, swyang, yljin\}@pku.edu.cn

Abstract

Recent works reveal that network embedding techniques enable many machine learning models to handle diverse downstream tasks on graph structured data. However, as previous methods usually focus on learning embeddings for a single network, they can not learn representations transferable on multiple networks. Hence, it is important to design a network embedding algorithm that supports downstream model transferring on different networks, known as domain adaptation. In this paper, we propose a novel Domain Adaptive Network Embedding framework, which applies graph convolutional network to learn transferable embeddings. In DANE, nodes from multiple networks are encoded to vectors via a shared set of learnable parameters so that the vectors share an aligned embedding space. The distribution of embeddings on different networks are further aligned by adversarial learning regularization. In addition, DANE’s advantage in learning transferable network embedding can be guaranteed theoretically. Extensive experiments reflect that the proposed framework outperforms other well-recognized network embedding baselines in cross-network domain adaptation tasks.

1 Introduction

Network embedding, which learns low-dimensional embedding vectors for nodes from networks, is an important technique enabling the applications of machine learning models on network-structured data [Perozzi et al., 2014; Tang et al., 2015; Grover and Leskovec, 2016; Hamilton et al., 2017]. It learns to preserve structural and property similarity in embedding space and can support different downstream machine learning tasks, such as node classification [Yang et al., 2016] and network visualization [Tang et al., 2016].

However, most of existing methods mainly focus on learning representations for nodes from a single network. As a result, when handling multiple networks, they suffer from embedding space drift [Du et al., 2018] and embedding distribution discrepancy [Tzeng et al., 2017], resulting in decreased accuracy when transferring the downstream machine learning models across networks to handle the same task. As the branch of transfer learning that handles the same task on different datasets is usually known as domain adaptation [Mansour et al., 2009], we define the network embedding algorithms that can support such transfer learning on different networks as domain adaptive network embedding, which brings following advantages. The first advantage is that it alleviates the cost of training downstream machine learning models by enabling models to be reused on other networks. The second advantage is that it handles the scarcity of labeled data by transferring downstream models trained well on a labeled network to unlabeled networks. Apart from the above advantages, compared with traditional discriminative domain adaptation methods requiring label information from source domain, domain adaptive network embedding learns representations in an unsupervised manner, requiring no label information from neither the source network nor the target network. Therefore, it makes no difference which network is the source network in downstream tasks, enabling bidirectional domain adaptation.

Currently, most researches of domain adaptation concentrate on CV and NLP fields [Long et al., 2015; Fu et al., 2017], existing domain adaptation methods can not be directly applied on network embedding problems. First, these methods are usually designed for CV and NLP tasks, where samples, e.g. images and sequences are independent and identically distributed, resulting in little requirement for model rotational invariance [Khasanova and Frossard, 2017]. However, network structured data, where nodes are connected with edges representing their relations, require models with rotational invariance because of the phenomenon known as graph isomorphism [Defferrard et al., 2016]. Therefore, existing methods can not model network structural information, which is the core of network embeddings. Second, most existing domain adaptation models learn discriminative representations in a supervised manner [Tzeng et al., 2017], where the value of loss function is only associated with each single sample’s absolute position in their feature space. Network embedding, alternatively, usually aims to learn multi-purpose representations in an unsupervised manner by preserving the relative position of all node pairs, resulting in increased difficulty in optimization. As a result, more stable

*Corresponding author
model architecture and loss functions are required to alleviate such difficulty.

Essentially, domain adaptive network embedding suffers from two major challenges, the more straightforward of which is the embedding space alignment, which means that structurally similar nodes should have similar representations in the embedding space, even if they are from different networks. However, many typical network embedding methods only preserve the structural similarity within a single network [Heimann and Koutra, 2017] which are not applicable for cross-network node pairs. The other less explicit challenge is that distribution shift of embedding vectors also influences the performance of model on target networks, because most machine learning models perform as guaranteed only when they work on data with similar distribution as training data.

In this paper, we propose DANE, an unsupervised network embedding framework handling embedding space drift and distribution shift in domain adaptation via graph convolutional network [Kipf and Welling, 2016a] and adversarial learning [Goodfellow et al., 2014]. To enable GCN to preserve the structural similarity of cross-network node pairs, we apply shared weight architecture, which means that the GCN embeds nodes from both source network and target network via shared learnable parameters.

The distribution shift in domain adaptation is handled via an adversarial learning component based on least square generative adversarial network [Mao et al., 2017]. Compared with the original GAN used in existing domain adaptation models, the loss function and architecture of LSGAN can generate gradients with larger scale for samples lying a long way to the decision boundary, which relieves the problem of vanishing gradients when models are close to convergence.

In summary, our contributions are:

- We formulate the task of designing a domain adaptive network embedding framework. It is beneficial for transferring models across multiple networks, which has not been explored by previous network embedding methods.
- We propose DANE, a framework that can achieve embedding space alignment and distribution alignment via shared weight graph convolutional network and adversarial learning regularization based on LSGAN.
- We constructed two datasets to test network embedding methods’ performance on the task of supporting domain adaptation and conducted experiments on DANE and other baselines via these datasets. The result indicates that our model has leading performance in this task.

2 Related Work

Network Embedding. Network embedding maps the vertices or edges of a network into a low-dimensional vector space. Such mapping can benefit the application of many machine learning models in many downstream tasks on network structured data [Yang et al., 2016; Tang et al., 2016; Zhang et al., 2018b; Zhang et al., 2018a; Wang et al., 2019]. Existing methods include transductive methods and inductive methods. Transductive methods directly optimize the representation vectors. They usually apply matrix factorization [Li et al., 2019] or Skip-Gram model inspired by word2vec [Perozzi et al., 2014; Tang et al., 2015; Grover and Leskovec, 2016]. Inductive methods learn functions which take the structural information and node features as input and output their representation vectors. They usually model the mapping function via deep neural networks like graph convolutional networks [Kipf and Welling, 2016b; Hamilton et al., 2017]. All the methods above mainly focus on representing a single network better, without considering the domain adaptation on multiple unconnected networks.

Domain Adaptation. Domain adaptation aims to learn machine learning models transferable on different but relevant domains sharing the same label space [Mansour et al., 2009]. Recent research mainly focuses on learning domain invariant representation via neural networks so that deep learning models trained on labeled source datasets can be transferred to a target dataset with few or no labeled samples [Ganin and Lempitsky, 2015; Long et al., 2015; Peng and Dredze, 2017; Fu et al., 2017; Tzeng et al., 2017]. Above methods are mostly designed for image or text while not considering the application on graph structured data because of their lack of rotational invariance and limits in handling graph isomorphism. Only a few methods [Alam et al., 2018; Das and Lee, 2018] make use of information of a similarity graph whose edges represent artificially designed similarity (like dot product of representation vectors) of independent samples (text or images), instead of real links between relevant nodes.

3 Problem Definition

Definition 1: Domain Adaptation on Networks. Domain adaptation on networks aims to train a machine learning model M for a downstream task by minimizing its loss function on G_{src} and ensure that M can also have good performance when we transfer it on G_{tgt} to handle the same task.

To this end, following two constraints need to be satisfied:

- **Embedding Space Alignment.** Embedding space alignment aims to project the nodes of G_{src} and G_{tgt} into a shared embedding space Z, where structurally similar nodes have similar representation vectors even if they are from different networks, so that M can be transferable on G_{src} and G_{tgt}.
- **Distribution Alignment.** Distribution alignment aims to constrain p_{src}(z) and p_{tgt}(z) close almost everywhere in Z, so that V_{src} and V_{tgt} can have similar distributions.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>Network</td>
</tr>
<tr>
<td>G_{src}</td>
<td>Source network</td>
</tr>
<tr>
<td>G_{tgt}</td>
<td>Target network</td>
</tr>
<tr>
<td>N_{G}</td>
<td>Node set of network G</td>
</tr>
<tr>
<td>E_{G}</td>
<td>Edge set of network G</td>
</tr>
<tr>
<td>X_{G}</td>
<td>Node feature matrix of network G</td>
</tr>
<tr>
<td>Z</td>
<td>Embedding space</td>
</tr>
<tr>
<td>V_{G}</td>
<td>Embedding vector set of network G</td>
</tr>
<tr>
<td>p_{G}(z)</td>
<td>Probability density function of V_{G} in Z</td>
</tr>
</tbody>
</table>

Table 1: Notations.
4 Proposed Method

4.1 Overall Framework
To enable domain adaptation on networks, we propose domain adaptive network embedding (DANE), a model that leverages a Shared Weight Graph Convolutional Network to achieve embedding space alignment and apply Adversarial Learning Regularization to achieve distribution alignment. The overview of our model is shown in Fig 1.

- **Shared Weight Graph Convolutional Network.** In order to learn embeddings that preserve the cross-network structural similarity, we use shared learnable parameters when encoding the nodes from two networks to vectors via GCN.

- **Adversarial Learning Regularization.** To align the distribution of embedding vectors from different networks, we apply adversarial learning to force our model to learn embeddings that can confuse the discriminator trained to distinguish which network a representation vector is from.

Without loss of generality, we assume $G_A$ is the source network and $G_B$ is the target network. As DANE’s architecture and loss function are symmetric, it is able to handle bidirectional domain adaptations.

4.2 Shared Weight Graph Convolutional Network
Graph Convolutional Network (GCN) [Kipf and Welling, 2016a] represents each vertex in a graph as an embedding vector based on node feature matrix $X$ and adjacency matrix $A$. In GCN, each layer can be expressed as follows:

$$H^{(l+1)} = \sigma(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} H^{(l)} W_l)$$

where $\hat{A} = A + I_N$, $\hat{D}_{ii} = \sum_j \hat{A}_{ij}$, $H^{(l)}$ is the output of the $l$-th layer, $H^{(0)} = X$, $\sigma$ is activation function and $W_l$ are learnable parameters of the $l$-th layer. We use a shared parameter set $\theta_s = \{W_1, W_2, \ldots\}$ to embed both source and target model. As proven by Jure Leskovec et al [Donnat et al., 2018], graph convolution operations can preserve the similarity of a node pair if their local subnetworks (i.e. the subgraph consisting of $k$-hop neighbors of a node and the node itself) are similar. Therefore, GCN with such shared weight architecture can preserve local subnetwork similarity. Under structural equivalence hypothesis in complex network theory [Henderson et al., 2012; Grover and Leskovec, 2016], two nodes having similar local network neighborhoods can be considered structurally similar even if they are from two different networks. Hence, shared weight graph convolutional network architecture can project the nodes of $G_{src}$ and $G_{tgt}$ into a shared embedding space $Z$, where structurally similar nodes have similar representation vectors.

To learn compatible parameters for both $G_{src}$ and $G_{tgt}$, we apply a multi-task loss function preserving properties on two networks simultaneously:

$$L_{gcn} = L_{G_{src}} + L_{G_{tgt}}$$

where $L_{G_{src}}$ denotes the loss function on the source network and $L_{G_{tgt}}$ denotes the same function calculated on the target network.
network. We apply the first-order loss function proposed in
LINE [Tang et al., 2015]:
\[L_G = - \sum_{(i,j) \in E} \log \sigma(v_j \cdot v_i) - Q \cdot \mathbb{E}_{k \sim P_{\text{neg}}(N)} \log \sigma(-v_i \cdot v_k)\]
where \(E\) is the set of edge, \(Q\) is the number of negative samples, \(P_{\text{neg}}\) is a noise distribution where negative samples are drawn and \(\sigma\) is the sigmoid function. In this paper, we set \(P_{\text{neg}}(N) \propto d_n^{u,75}\), where \(d_n\) is degree of node \(n\). As a flexible framework, DANE is able to preserve other network properties like triadic closure [Huang et al., 2014] when we replace \(L_G\) with corresponding loss functions.

### 4.3 Adversarial Learning Regularization

In order to align the distribution of \(V_{\text{src}}\) and \(V_{\text{tgt}}\) in embedding space \(Z\), we add Adversarial Learning Regularization into our model. We train a discriminator to distinguish which network an embedding vector is from, and train the Shared Weight Graph Convolutional Network to confuse the discriminator, just like what GANs do. Such training method will force \(P(v \in V_{\text{src}}|v = z)\) and \(P(v \in V_{\text{tgt}}|v = z)\) to keep close almost everywhere in the embedding space \(Z\), which is equivalent to distribution alignment.

In this work, inspired by LSGAN [Mao et al., 2017], we design the architecture and loss function of discriminator based on Pearson \(\chi^2\) divergence to avoid the instability of adversarial learning. 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 \(V_{\text{src}}\) and output 1 otherwise. So the discriminator’s loss function is:

\[L_D = \mathbb{E}_{x \in V_{\text{src}}}[(D(x) - 0)^2] + \mathbb{E}_{x \in V_{\text{tgt}}}[(D(x) - 1)^2] \quad (4)\]

where \(D(z)\) is the output of the discriminator. In LSGAN, on the contrary, the generator confuses the discriminator unidirectionally by forcing the distribution of fake samples to approximate that of real samples. However, hoping to keep DANE’s architecture and loss function symmetric so that it can handle bidirectional domain adaptation, we design following adversarial training loss function:

\[L_{adv} = \mathbb{E}_{x \in V_{\text{src}}}[(D(x) - 1)^2] + \mathbb{E}_{x \in V_{\text{tgt}}}[(D(x) - 0)^2] \quad (5)\]

We combine the training of Shared Weight Graph Convolutional Network and Adversarial Learning Regularization together by defining the overall loss function of DANE as follows:

\[L = L_{\text{gen}} + \lambda L_{adv} \quad (6)\]

where \(\lambda\) is a hyperparameter to control the weight of regularization. In this paper, we set \(\lambda\) as 1. In each iteration, we first train the discriminator for \(k\) steps to optimize \(L_D\), followed by training our embedding model for 1 step to optimize the graph convolutional network based on Equation 6.

### 4.4 Theoretical Analysis

In this section, we show that the better embedding spaces and distributions are aligned, the better the downstream model \(M\) performs on target network. We provide a theoretical analysis by exploring the relation between alignment effects and the upper bound of the difference between the loss function value on model \(M\) on the source network and the target network when handling a node classification task.

We assume that \(M\) outputs the conditional distribution of a node’s label \(y\) based on its representation vector \(v\) and model parameter \(\theta\), denoted as \(P(y|v; \theta)\). \(P(y|v; \theta)\) gives the possibility that a node has a label given the embedding vector of the node. The loss function of the model on \(G_{\text{src}}\), denoted as \(L_{\text{src}}\), is defined as follows:

\[L_{\text{src}} = \mathbb{E}(D(\hat{P}_{\text{src}}(y|v), P(y|v; \theta))) \quad (7)\]

where \(\hat{P}_{\text{src}}(y|v)\) is the groundtruth, and \(D(P_1, P_2)\) measures the distance of two distributions. Assuming that \(G_{\text{src}}\) contain as many nodes as being able to approximate \(v\) as a continuous variable, we have the following equation:

\[L_{\text{src}} = \int Z p_{\text{src}}(z) \cdot D(\hat{P}_{\text{src}}(y|z), P(y|z; \theta))dz \quad (8)\]

In a similar way, the performance of the same model on \(G_{\text{tgt}}\) can be measured by following loss function:

\[L_{\text{tgt}} = \mathbb{E}(D(\hat{P}_{\text{tgt}}(y|v), P(y|v; \theta)))
= \int Z p_{\text{tgt}}(z) \cdot D(\hat{P}_{\text{tgt}}(y|z), P(y|z; \theta))dz \quad (9)\]

We introduce a theorem:

**Theorem 1** If following inequalities are satisfied:

\[D(\hat{P}_{\text{src}}(y|z), \hat{P}_{\text{tgt}}(y|z)) < c, \forall z \in Z \quad (10)\]

\[|p_{\text{src}}(z) - p_{\text{tgt}}(z)| < \epsilon, \forall z \in Z \quad (11)\]

where \(D(P_1, P_2)\) measures the distance of two distributions and satisfy triangular inequality:

\[D(P_1, P_3) \leq D(P_1, P_2) + D(P_2, P_3) \quad (12)\]

Then we will have following inequality:

\[L_{\text{tgt}} - L_{\text{src}} \leq \epsilon L_{\text{src}} + c + c \cdot \epsilon \quad (13)\]

**Proof**

\[L_{\text{tgt}} - L_{\text{src}} = \int Z p_{\text{tgt}}(z) \cdot D(\hat{P}_{\text{tgt}}(y|z), P(y|z; \theta))dz
- \int Z p_{\text{src}}(z) \cdot D(\hat{P}_{\text{src}}(y|z), P(y|z; \theta))dz\]

\[L_{\text{tgt}} - L_{\text{src}} \leq \epsilon \int Z p_{\text{src}}(z) \cdot D(\hat{P}_{\text{src}}(y|z), P(y|z; \theta))dz
+ \int Z p_{\text{src}}(z) \cdot D(\hat{P}_{\text{src}}(y|z), \hat{P}_{\text{tgt}}(y|z))dz
+ \epsilon \int Z p_{\text{src}}(z) \cdot D(\hat{P}_{\text{src}}(y|z), \hat{P}_{\text{tgt}}(y|z))dz \quad (14)\]
This theorem ensures that an embedding algorithm can support domain adaptation better when it: (1) makes $p_{src}(z)$ and $p_{tgt}(z)$ closer; (2) makes $P_{src}(y|z)$ and $P_{tgt}(y|z)$ closer. Obviously, the former objective can be achieved via distribution alignment. Simultaneously, the latter objective is achieved via embedding space alignment. Because DANE applies shared weight GCN, it represents two nodes having similar local network neighborhoods with similar embedding vectors. Meanwhile, under structural equivalence hypothesis, two nodes having similar local network neighborhoods are likely to have same label. Consequently, embedding space alignment can help keep $P_{src}(y|z)$ and $P_{tgt}(y|z)$ close almost everywhere in $Z$.

5 Experiments

5.1 Experiment Settings

Datasets

Paper Citation Networks. Paper Citation Networks consist of two different networks $A$ and $B$, where each node is a paper. The label of each paper is its field. The feature of each node is a word frequency vectors constructed from the abstract of papers.

Co-author Networks. Co-author Networks consist of two different networks $A$ and $B$, where each node is an author. Each author is assigned with one or more label denoting research topics. The feature of each node is a word frequency vector constructed from the keyword of the author’s papers.

Relevant Methods and Evaluation Metrics

We compare our model with well-recognized network embedding algorithms including DeepWalk, LINE, Node2vec, unsupervised GraphSAGE [Perozzi et al., 2014; Tang et al., 2015; Grover and Leskovec, 2016; Hamilton et al., 2017]. When using GraphSAGE, we train it on the source network and directly use the optimized parameter to embed the nodes from target network without further training. We evaluate these methods by testing their performance on a domain adaptation task of node classification. We train a classifier based on L2-regularized logistic regression via SGD algorithm with the embeddings from source network, then test the performance of the classifier on target network. We adopt micro and macro F1 score to evaluate the performance.

Hyperparameter Set-up

To be fair, for all methods we set the embedding dimension to 128 on Paper Citation Networks, and 32 on Co-author Networks. For methods applying negative sampling, we set negative sampling number as 5. For methods employing GCN, we use same activation function and 2-layer architecture.

Figure 2: Node classification performance of DANE ($\lambda = 0$) and DANE ($\lambda = 1$)

Figure 3: Performance vs Training Batches (Paper citation network $A \rightarrow B$). The orange line performs more steady than the blue line.

Table 2: Micro and macro F1 score of different network embedding methods in unsupervised domain adaptation

<table>
<thead>
<tr>
<th>Network Name</th>
<th>Nodes</th>
<th>Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Citation A</td>
<td>2277</td>
<td>8245</td>
</tr>
<tr>
<td>Paper Citation B</td>
<td>3121</td>
<td>7519</td>
</tr>
<tr>
<td>Co-author A</td>
<td>1500</td>
<td>10184</td>
</tr>
<tr>
<td>Co-author B</td>
<td>1500</td>
<td>10606</td>
</tr>
</tbody>
</table>

Table 3: Network size of two datasets.

Figure 3: Performance vs Training Batches (Paper citation network $A \rightarrow B$). The orange line performs more steady than the blue line.

each node is a word frequency vectors constructed from the abstract of papers.

4366
5.2 Domain Adaptation

In this section, we firstly compare DANE with other baselines by training classifiers on source network embedding and testing their performance on target network embedding. Table 2 shows the result of all methods on Paper Citation Networks and Co-author Networks. DANE outperforms all other methods in knowledge transferring. Besides, we can find following phenomenons:

All deep network embedding algorithms outperforms other methods. We propose that this is because the local sub-network is suitable for measuring the similarity cross-network node pairs. DeepWalk, LINE and Node2vec all preserve k-order proximity, which consider two nodes within k hops are similar. However, the distance between two nodes from different networks is infinity. Consequently, the structural similarity of cross-network node pairs can not be preserved, resulting in poorly aligned embedding spaces and performance close to random guess on target network.

Meanwhile, with same architecture of GCN, DANE outperform GraphSAGE. This phenomenon reflects the contribution of the combination of distribution alignment and the multi-task loss function.

5.3 Ablation Test

In this section, by comparing DANE with its variants, we indicate the importance of DANE’s unique design.

First, to analyze the importance of distribution alignment, we compare complete DANE, denoted as DANE (λ = 1), and its variant which has no adversarial learning regularization, denoted as DANE (λ = 0). The result is shown in Fig 2. DANE (λ = 1) performs better in both transferring directions on two networks comparing to its own variant that has no adversarial learning regularization. This phenomenon indicates the importance of adversarial learning regularization.

Second, we replace the LSGAN-based adversarial learning regularization with GAN-based regularization while preserving all other hyper-parameter settings. Fig 3 is a line chart reflecting the relation between number of training batches and model’s performance on target network. Compared with the LSGAN-based DANE, the GAN-based DANE suffers from a more serious performance drop after reaching the peak.\(^2\)

5.4 Embedding Visualization

In this section, to understand the advantage of distribution alignment better, we visualize the embedding of two paper citation networks generated by DANE (λ = 1), DANE (λ = 0) and a randomly initialized GCN. The randomly initialized GCN also embeds two networks via shared parameters. We use t-SNE [Maaten and Hinton, 2008] package to reduce the dimensionality of embedding vectors to 2. The result is shown in Fig 4. The color of each node represent its label.

Due to shared parameter architecture, all three algorithms achieve that most nodes having similar representations have same labels. This phenomenon indicates the advantage of share parameter architecture: it can align embedding space without requirement for training. After further training without distribution alignment, DANE (λ = 0) succeeds in making the distribution of two network’s embedding similar. Furthermore, when adding distribution alignment, DANE (λ = 1) can generate embeddings with more similar distributions in the area nearby boundary line of different labels than DANE (λ = 0). Therefore, DANE will have better performance in model transferring task.

6 Conclusion and Future Works

In this paper, we formulate the task of unsupervised network embedding supporting domain adaptation on multiple domain compatible networks. To the best of our knowledge, we are the first team to propose this significant task. Meanwhile, we propose Domain Adaptive Network Embedding (DANE) to handle this task. Specifically, we apply a shared weight graph convolutional network architecture with constraints of adversarial learning regularization. Empirically, we verify DANE’s performance in a variety of domain compatible network datasets. The extensive experimental results of node classification and embedding visualization indicate the advantages of DANE. For future work, one intriguing direction is to generalize DANE to solve the domain adaptation on heterogeneous networks. Also, improving the framework of DANE to solve semi-supervised and supervised domain adaptation will be beneficial for diverse scenarios.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 61876006 and No. 61572041).
References


