Network Embedding for Cross-network Node Classification

Xiao Shen, Fu-lai Chung
Department of Computing
Hong Kong Polytechnic University
xiao.shen@connect.polyu.hk, cskchung@comp.polyu.edu.hk

ABSTRACT

Network embedding is an effective method to learn low-dimensional node vector representations which can well preserve the original network structures. Existing network embedding algorithms are mostly developed for a single network, which fail to learn generalized and comparable feature representations across networks. In this paper, we address a cross-network node classification problem by leveraging the abundant labeled information from a source network to help classify unlabeled nodes in a target network. A cross-network deep network embedding (CDNE) model is proposed to embed the nodes from the source network and the target network into a unified low-dimensional latent space. This model integrates deep network embedding and domain adaptation to learn label-discriminative and network-invariant node vector representations. The network structures, node attributes and node labels are leveraged collectively to learn similar hidden vector representations for similar nodes within a network and across different networks. Extensive experimental results demonstrate that the proposed model significantly outperforms the state-of-the-art related algorithms for node classification in the target network.

1 Introduction

Domain adaptation aims to transfer the knowledge pre-learned from a source domain to assist in solving the same task in a target domain [1]. Domain adaptation has been widely applied to computer vision [2-4] and natural language processing (NLP) [5, 6]. However, applying domain adaptation to social network analysis like classifying nodes across networks has not been sufficiently investigated. Given a source network having fully labeled nodes and a target network having very small fraction of labeled nodes but a large number of unlabeled nodes, one can take advantage of the rich labeled information from the source network to help build an accurate node classifier for the target network in a domain adaptation manner.

Network embedding is a highly effective method to learn low-dimensional node vector representations with the original network structures and properties being well preserved. Existing network embedding algorithms are generally developed for a single network [7-14], which can only capture the proximities between nodes within a network while failing to capture the proximities between nodes from different networks. Thus, single-network embedding algorithms would fail to learn generalized and aligned feature representations across different networks [15, 16]. It is indeed challenging to measure the cross-network node proximities solely based on network topological structures. At the same time, nodes in the real-world networks are often associated with rich attributes, e.g., users in social networks have profile information and papers in citation networks have titles and abstracts. Intuitively, node attributes should be more comparable than network structures across different networks, especially when node labels depend more on homophily effect [17] rather than structural identity [18]. For example, the papers belonging to the same research area, say “Information Security”, from two different citation networks, might be likely to include some common keywords in their titles, such as “Privacy, Verification, Encryption, Decryption, Cryptography”, while they might have rather distinct topological structures in different networks. The cross-network node classification task faces the following challenges: 1) nodes from different networks do not inherently have the same set of pre-defined feature vector representations; 2) the same labeled nodes across different networks can have rather distinct topological structures; 3) even though node attributes are more comparable, nodes from different networks might not share the same set of attributes and the attribute distributions across networks can be rather varied.

To address such a challenging cross-network node classification problem, we aim to leverage network structures, node attributes and node labels collectively to capture the proximities between nodes within a network and across different networks. A novel cross-network deep network embedding (CDNE) model is proposed to employ two stacked auto-encoders (SAEs), namely SAE_s and SAE_t, to embed nodes from the source network and the target network into a unified low-dimensional latent space. On one hand, network structures are leveraged to capture the proximities between nodes within a network. Firstly, SAE_s and SAE_t are employed to learn low-dimensional node vector representations by reconstructing the associated network structural proximity matrices of the source network and the target network, respectively. In addition, the pairwise constraints have been incorporated into SAE_s and SAE_t to embed more strongly connected nodes within each network closer in the latent space. On the other hand, the available attributed and labeled information can be leveraged to capture the proximities between nodes across different networks. SAE_s is firstly employed to learn label-discriminative feature vector representations for the source network nodes, by mapping nodes
belonging to the same class closer while those belonging to completely different classes far apart from each other. Then, SAE_t is employed to learn network-invariant feature vector representations for the target network nodes, by minimizing the cross-network marginal and class-conditional distribution discrepancies. As a result, the target network nodes would have similar latent vector representations w.r.t. the source network nodes associated with the same labels. Note that in SAE_s, different categories of source network nodes have already been mapped separately in the embedding space. Thus, by matching the cross-network class-conditional distributions in SAE_t, the nodes associated with the same labels no matter within the target network or across networks would have similar latent vector representations, while the nodes associated with completely different labels within the target network and across networks would have rather different latent vector representations. Such properties yield label-discriminative and network-invariant node vector representations, which significantly benefits the cross-network node classification problem. The contributions of this work can be summarized as follows:

1. To the best of our knowledge, it is the first cross-network deep network embedding model developed for the challenging cross-network node classification problem;
2. By leveraging network structures, node attributes and node labels collectively to capture the within-network and cross-network proximities, label-discriminative and network-invariant node feature vector representations can be learned;
3. Extensive experimental results in various real-world datasets demonstrate that the proposed CDNE model significantly outperforms the state-of-the-art related models/algorithms for node classification in the target network.

The rest of this paper is organized as follows. Section 2 reviews the network embedding and network transfer learning algorithms. Section 3 introduces the detailed framework of CDNE. Section 4 reports the experimental results of CDNE. Section 5 concludes this paper.

2 Related Work

In this section, we review the state-of-the-art network embedding algorithms and network transfer learning models.

2.1 Network Embedding

Network embedding generally aims to map nodes with higher proximities in a network closer to each other in the low-dimensional latent space. Line [11] and SDNE [12] seek to preserve the first-order and second-order proximities between nodes based on the first-order and second-order neighborhoods. DeepWalk [9] and node2vec [13] employ the random walk sampling strategy to generate the neighborhood of each node. Then, the Skip-Gram with Negative sampling (SGNS) model [19] is extended to learn node vector representations with the preservation of neighborhood structure. In addition, in [7, 8, 10, 20, 21], the positive pointwise mutual information (PPMI) [22] metric has been widely employed to capture high-order proximities between nodes. Then, deep learning [8, 10] or matrix-factorization [7, 20, 21] techniques have been employed to learn more similar feature vector representations for nodes which can more easily reach each other within K steps. It has been shown that the high-order proximities which can capture the global structural information are more beneficial for learning informative feature representations [7]. However, such proximities are defined in terms of the neighborhoods between nodes, thus, the proximity-based network embedding algorithms [7-14] would fail to capture proximities between disconnected nodes.

Besides the plain network structures, nodes in the real-world networks are often associated with rich attributes. Recently, a family of attributed network embedding algorithms [20, 21, 23-28] has been proposed to preserve both network topological proximity and node attribute affinity. For example, Chang et al. [24] proposed a heterogenous network embedding framework to learn node vector representations based on both node contents and linkage structures. Yang et al. [20] proposed a TADW model to learn network representations from both textual information and network structures based on a matrix factorization framework. Huang et al. [23] developed a LANE algorithm to incorporate label information into attributed network embedding to jointly project node labels, network structures and node attributes into a unified embedding space via eigenvector decomposition (EVD). Zhang et al. [25] proposed an ANRL algorithm which employs a neighbor enhancement autoencoder and an attribute-aware skipgram model to learn network representations from both network structures and node attributes. However, all these attributed network embedding algorithms [20, 21, 23-28] just utilize the information from a single network.

2.2 Transfer Learning Across Networks

Network transfer learning studies the transferal of useful knowledge from a source network to assist in the same prediction task in a target network. For example, Ye et al. [29] proposed a transfer learning approach to predict the signed labels of edges in a target network by leveraging the edge labeled information from a source network. Tang et al. [30] aim to classify the type of social relationships in a target network by borrowing the knowledge from a source network. Shen et al. [31] developed a CNNS model to predict the edges least likely to be active for influence propagation in a target network, by leveraging the knowledge pre-learned from a source network. Fang et al. [32] developed a network transfer learning algorithm for cross-network node classification, which utilizes Nonnegative Matrix Tri-Factorization (NMTF) technique to project the label propagation matrices of the source network and the target network into a common latent space so as to learn the shared latent structural features across networks.

To effectively transfer the knowledge from a source network to a target network for cross-network predictions, it is required that the feature vector representations should be generalized and comparable across networks. Existing network transfer learning methods either manually define the same set of explicit topological features for the source network and the target network [30, 31], or learn the common latent features by factorizing the associated matrices of the source network and the target network.
into a unified latent space [29, 32]. To the best of our knowledge, the proposed CDNE model is the first to take advantage of deep network embedding to learn label-discriminative and network-invariant node vector representations for network transfer learning.

3 Problem Statement
Let \( G^s = (V^s, E^s, Y^s) \) be a fully labeled source network, with a set of all labeled nodes \( V^s \) and a set of edges \( E^s \). \( Y^s \in \mathbb{R}^{n^s \times c} \) is a label matrix associated with \( G^s \), where \( n^s = |V^s| \) is the number of nodes in \( G^s \) and \( c \) is the number of node categories. \( Y^s_{ik} = 1 \) if node \( v^s_i \in V^s \) is associated with label \( k \); otherwise, \( Y^s_{ik} = 0 \). A node can have multiple labels.

Let \( G^t = (V^t, E^t, Y^t) \) be an insufficiently labeled target network with a set of nodes \( V^t = \{v^t_i, v^t_j\} \) and a set of edges \( E^t \), where \( n^t = |V^t| \) denotes the number of nodes in \( G^t \), \( V^t \) indicates a small set of labeled nodes and \( V^t \) represents a larger set of unlabeled nodes in \( G^t \). \( V^t \in \mathbb{R}^{n^t \times c} \) is the observable label matrix associated with \( G^t \), where \( Y^t_{ik} = 1 \) if node \( v^t_i \in V^t \) has an observable label \( k \); otherwise, \( Y^t_{ik} = 0 \).

In addition, nodes can be associated with attributes. Let \( A^s \) and \( A^t \) denote the set of node attributes in \( G^s \) and \( G^t \), respectively, where \( w^s = |A^s| \) and \( w^t = |A^t| \) represent the number of node attributes in \( G^s \) and \( G^t \). Note that the nodes from the source network and the target network might not share the same set of attributes, i.e., \( A^s \neq A^t \), but we can build a union set between \( A^s \) and \( A^t \) as \( A^s \cup A^t \), where \( w = |A| \) represents the number of union attributes between \( G^s \) and \( G^t \). Then, we construct two matrices \( A^s \in \mathbb{R}^{n^s \times w} \) and \( A^t \in \mathbb{R}^{n^t \times w} \) to represent the node attributed values associated with \( G^s \) and \( G^t \), respectively. \( A^s_{i_r} > 0, r \in \{s, t\} \) represents the value of the \( k \)-th attribute (in the union set \( A \)) associated with the \( i \)-th node in \( G^r \); while \( A^s_{i_r} = 0 \) indicates that the \( i \)-th node in \( G^s \) is not associated with the \( k \)-th attribute.

In our defined cross-network node classification problem, the network dimensionality (i.e. number of nodes), and the distributions of network connections and node attributes are varied across networks. However, the two networks should share the same set of node labels. The goal of cross-network embedding is to learn appropriate cross-network node vector representations such that the abundant labeled information from the source network can be successfully leveraged to classify unlabeled nodes in the target network.

4 Cross-Network Deep Network Embedding
In this section, we elaborate on the framework of the proposed CDNE model. As shown in Figure 1, CDNE leverages the single-network topological structures to capture the proximities between nodes within a network and utilizes the available node attributes and labels to capture the proximities between nodes across different networks. CDNE consists of two SAEs. Firstly, SAE_s is employed to learn label-discriminative node vector representations for the source network. Then, SAE_t is employed to learn network-invariant node vector representations in the target network, by minimizing the maximum mean discrepancy (MMD) between the hidden representations learned by each SAE.

Figure 1: The framework of the proposed CDNE model. SAE_s and SAE_t have the same number of hidden layers and the same dimensionality for each \( k \)-th (\( 1 \leq k \leq L \)) hidden layer, but have different input dimensionalities. \( X^s_f \in \mathbb{R}^{1 \times n^s} \) and \( X^t_f \in \mathbb{R}^{1 \times n^t} \) denote the structural-proximity vectors associated with the \( i \)-th node in \( G^s \) and the \( j \)-th node in \( G^t \), respectively. \( H^s_i(k), H^t_j(k) \) represent the latent vector representation of the \( i \)-th node in \( G^s \) and the \( j \)-th node in \( G^t \), learned by the \( k \)-th layer of SAE_s and the \( k \)-th layer of SAE_t, respectively.

In several state-of-the-art network embedding algorithms [7, 8, 10, 20, 21], the PPMI metric [22] has been employed to measure the topological proximities between nodes based on their network connections. Also, the SGNS model [19] which has been widely utilized in the random walk based network embedding algorithms [9, 13, 25, 33], has been proved to be equivalent to performing factorization on the PPMI matrix [7, 20, 22]. Motivated by this, we employ the PPMI metric to capture the network structural proximities between nodes within a network. Given a network \( G^r = (V^r, E^r) \), one can employ the random walk sampling strategy [8, 33] to obtain a \( k \)-step transition probability matrix \( T^r(k) \in \mathbb{R}^{n^r \times n^r} \), where \( T^r(k)_{ij} \) indicates the probability of visiting \( v^r_j \) from \( v^r_i \) after exactly \( k \) steps in network \( G^r \). Based on a series of \( k \)-step transition probability matrices up to the maximum \( K \)-step, i.e., \( \{T^r(k)\}_{k=1}^K \), we can aggregate an overall transition probability matrix by weighting closer neighborhoods more, as in [10]: \( T^r = \sum_{k=1}^{K} T^r(k)/k \). Then, the PPMI metric [22] can be employed to measure the structural proximities between nodes as:

\[ X^s_{ij} = \max \left( \log \frac{\hat{f}^r_{ij}}{\sum_{k=1}^{K} T^s(k)_{ij}/ns}, 0 \right) \]

where \( \hat{f}^r \) indicates the row-wise normalized aggregated transition probability matrix. The \( i \)-th row vector of \( X^r \), i.e., \( X^r_i \in \mathbb{R}^{1 \times n^r} \) captures the structural proximity from \( v^r_j \) towards all the nodes in \( G^r \). \( X^s_{ij} > 0 \) iff \( v^s_i \) has a strong network connection towards \( v^s_j \) within \( K \) steps in \( G^r \); otherwise, \( X^s_{ij} = 0 \).

4.1 SAE_s in Source Network
Given the aggregated PPMI matrix of the source network as the input, a \( l \)-layer SAE_s is constructed as follows:
where (1) and (2) represent the encoding and decoding process of SAE_s, respectively. \( H^s(k) = X^s \in R^{n_s \times n_t} \) is the input PPMI matrix of SAE_s, \( H^s(k) = X^s \in R^{n_s \times d(k)}, \forall 1 \leq k \leq l \) denotes the hidden matrix representation learned by the k-th layer of SAE_s, and \( d(k) \) is the hidden dimensionality of the k-th layer of SAE_s. The i-th row vector of \( H^s(k) \), denoted as \( H^s_i(k) \), represents the latent feature vector representation of node \( v^s_i \) learned by the k-th layer of SAE_s. \( \hat{H}^s(k) \) is the reconstructed matrix of \( H^s(k) \) and \( \hat{H}^s_i(k) = H^s_i(k) \). In addition, \( W^s_1(k) \in R^{d(k) \times d(k-1)} \), \( B^s_1(k) \in R^{n_s \times d(k)} \), \( W^s_2(k) \in R^{d(k) \times d(k)} \), and \( B^s_2(k) \in R^{n_s \times d(k)} \) refer to the encoding weight, encoding bias, decoding weight and decoding bias matrices associated with the k-th layer of SAE_s, respectively. \( f \) is a non-linear activation function. In this work, the sigmoid function \( f(x) = 1/(1 + e^{-x}) \) is employed.

### 4.1.1 Preserving Single-Network Structural Proximities

By minimizing the reconstruction errors of SAE_s, nodes with more similar neighborhood structure would have more similar latent vector representations. In addition, to address the network sparsity issue, we follow [12] to incorporate a penalty matrix \( P^s(k) \) into the reconstruction errors as:

\[
\mathcal{R}^s(k) = \frac{1}{2n^s} \| P^s(k) \odot (\hat{H}^s(k-1) - H^s(k-1)) \|_F^2 \tag{3}
\]

where if \( H^s(k-1) > 0 \), \( P^s(k) = \beta > 1 \); and if \( H^s(k-1) = 0 \), \( P^s(k) = 1 \). \( P^s(k) \) specifies the ratio of penalty on the reconstruction errors of non-zero input elements over that of zero input elements. In addition, we design the following pairwise constraint to make more strongly connected nodes (i.e. with higher network structural proximities) have more similar vector representations:

\[
\mathcal{C}^s(k) = \frac{1}{2n^s d^s} \sum_{i,j=1}^{n^s} (X^s_i + X^s_j) \| H^s_i(k) - H^s_j(k) \|_2^2 \tag{4}
\]

\[
\operatorname{Tr}( (H^s(k))^T L X_s H^s(k)) / n^s
\]

where \( \operatorname{Tr}(. \) denotes the trace of a matrix, \( L^s = D^s - (X^s + (X^s)^T) \) is the Laplacian matrix of \( X^s + (X^s)^T \), and \( D^s \) is a diagonal matrix with the diagonal entry as \( (D^s)_{ii} = \sum_{j=1}^{n^s} X^s_{ij} + \sum_{j=1}^{n^s} X^s_{ji} \). By minimizing (3) and (4), the network structural proximities between nodes within the source network can be well preserved by the latent node vector representations.

### 4.1.2 Label-Discriminative Features

Next, a matrix \( O^s \in R^{n^s \times n_t} \) is defined to represent whether two nodes in \( G^s \) share common labels or not, where \( O^s_{ij} = -1 \) if \( v_i^s \) and \( v_j^s \) do not share any common labels, and \( O^s_{ij} = 1 \) if \( v_i^s \) and \( v_j^s \) share the number of common labels shared by \( v_i^s \) and \( v_j^s \). Then, the following pairwise constraint is devised to learn label-discriminative node vector representations:

\[
\mathcal{M}^s(k) = \frac{1}{2} \| \frac{1}{n^s} \sum_{i,j=1}^{n^s} O^s_{ij} \| H^s_i(k) - H^s_j(k) \|_2^2 \tag{5}
\]

\[
\frac{1}{2} - \frac{1}{n^s} 1^T H^s(k) - \frac{1}{n^s} 1^T H^s(k) \|_2^2
\]

where \( O^s \) can be divided into a positive part and a negative part as: \( O^s_+ = \max(O^s, 0) \) and \( O^s_- = -\min(O^s, 0) \). Then, (5) can be rewritten as \( \mathcal{M}^s(k) = \operatorname{Tr}( (H^s(k))^T L O^s H^s(k)) / n^s \), where \( L_{ij} = L^s_{ij} + L^s_{ji} \), \( L^s_{ij} = D^s_{ij} - O^s_+ \). \( L^s_{ij} = D^s_{ij} - O^s_- \) is the associated Laplacian matrix of \( O^s_+ \) and \( D^s_{ij} \) is a diagonal matrix of \( O^s_- \) with the diagonal entries as \( (D^s_{ij})_{ii} = \sum_{j=1}^{n^s} O^s_{ij} \). Similarly, \( L^s_{ij} = D^s_{ij} - O^s_- \) is the associated Laplacian matrix of \( O^s_- \). Minimizing (5) makes the nodes sharing more common labels have more similar latent vector representations, while making the nodes belonging to completely different categories have rather different latent vector representations.

#### 4.1.3 Loss Function of SAE_s

By integrating the reconstruction errors (3), the pairwise constraint on strongly connected nodes (4), the pairwise constraint on labeled nodes (5), and a L2-norm regularization to prevent overfitting \( \Omega^s(k) = \frac{1}{2} (\| W^s_1(k) \|_F^2 + \| W^s_2(k) \|_F^2) \), the overall loss function of SAE_s is defined as:

\[
J^s(k) = \mathcal{R}^s(k) + \alpha^s(k) \mathcal{C}^s(k) + \beta^s(k) \mathcal{M}^s(k) + \lambda^s(k) \Omega^s(k) \tag{6}
\]

where \( \alpha^s(k), \beta^s(k) \) and \( \lambda^s(k) \) are the trade-off parameters to balance the effects of different terms.

### 4.2 SAE_t in Target Network

#### 4.2.1 Preserving Single-Network Structural Proximities

Similar to SAE_s, a l-layer SAE is constructed in SAE_t. In addition, the hidden dimensionality at each k-th layer of SAE_t, i.e., \( d(k), (1 \leq k \leq l) \), are set as the same as in SAE_s. The input of SAE_t is the aggregated PPMI matrix of the target network, i.e., \( H^t(0) = X^t \in R^{n^t \times n_t} \). Thus, the input dimensionality of SAE_t is different from SAE_s, i.e., \( n^t \neq n^s \). Similar to SAE_s, we devise the reconstruction errors (7) and the pairwise constraint on strongly connected nodes (8) to preserve the structural proximities between nodes within the target network, as follows:

\[
\mathcal{R}^t(k) = \frac{1}{2n^t} \| P^t(k) \odot (\hat{H}^t(k-1) - H^t(k-1)) \|_F^2 \tag{7}
\]

\[
\mathcal{C}^t(k) = \frac{1}{n^t} \operatorname{Tr}( (H^t(k))^T L X_t H^t(k)) \tag{8}
\]

#### 4.2.2 Network-Invariant Features

Next, to learn feature representations aligned across networks, we need to match the distributions of the node vector representations learned for the target network with that of the source network. The widely utilized nonparametric MMD metric [34-36] is employed to measure the cross-network distribution differences. Firstly, the empirical marginal MMD [35] between the source network and the target network is defined as:

\[
\mathcal{M}^t_M^{(k)} = \frac{1}{2} \| \frac{1}{n^s} 1^T H^s(k) - \frac{1}{n^t} 1^T H^t(k) \|_2^2 \tag{9}
\]
where $1^s \in R^{1 \times n_s}$ and $1^t \in R^{1 \times n_t}$ denote two ones-vectors. By minimizing $M_{\delta}^{(k)}$, the marginal distributions between the source network and the target network can be matched.

Secondly, the class-conditional MMD [35] between the source network and the target network is defined as:

$$M_{\mathcal{C}}^{(k)} = \sum_{j=1}^{2} \frac{1}{2} \left( \sum_{i=1}^{n_s} v_{ij}^s \mu_{ij}^{(k)} - \sum_{i=1}^{n_t} v_{ij}^t \mu_{ij}^{(k)} \right)^2$$

(10)

where the first term and the second term represent the average feature vector representation of the nodes associated with label $j$ in networks $G^t$ and $G^s$, respectively, learned by the $k$-th layer of SAE_t and the $k$-th layer of SAE_s.

Note that unlike the source network nodes with completely observable labels, the target network nodes just have very limited observable labels. Directly utilizing the sparse observable label matrix $Y^t$ in (10) would fail to obtain sufficient statistics to update the weights of SAE_t and cannot well match the cross-network class-conditional distributions. To address this, we predict pseudo-labels for the unlabeled target network nodes based on the available attributes associated with the nodes from the source network and the target network. Since the original high-dimensional attributes might contain noises, we employ Principal Components Analysis (PCA) [37] to extract the low-dimensional (i.e. 128-D in the experiments) attribute vector representations. Then, we utilize a one-vs-rest logistic regression (LR) classifier to predict labels of the unlabeled nodes in the target network based on the low-dimensional attribute vector representations and all the observable labels in the source network and the target network. In addition, instead of using binary labels, we utilize fuzzy labels to represent the degree of membership of each node belonging to a specific class. Let $\bar{Y}^t$ denote the predicted label matrix of the target network, where if $v_{ij}^t \in V_1^t$, $\bar{Y}^t_{ij} = Y_{ij}^t \in \{0,1\}$; and if $v_{ij}^t \in V_0^t$, $0 < \bar{Y}^t_{ij} < 1$ represents the predicted probability of $v_{ij}^t$ to be labeled with $j$. Next, replacing $Y^t$ by $\bar{Y}^t$ which includes both observable binary labels and predicted fuzzy labels, we can re-compute the class-conditional MMD as:

$$M_{\mathcal{C}}^{(k)} = \sum_{j=1}^{2} \frac{1}{2} \left( \sum_{i=1}^{n_s} v_{ij}^s \mu_{ij}^{(k)} - \sum_{i=1}^{n_t} v_{ij}^t \mu_{ij}^{(k)} \right)^2$$

(11)

By minimizing (11), both observable and predicted labeled target network nodes would be aligned to the source network nodes associated with the same labels. In addition, it should be noted that minimizing (5) in SAE_s has already pulled the nodes belonging to different classes far apart from each other. Hence, minimizing (11) in SAE_t would simultaneously make different categories of target network nodes have rather different feature vector representations. Thus, the label-discriminative and network-invariant node vector representations can be simultaneously obtained by CDNE.

### 4.2.3 Loss Function of SAE_t

By integrating the reconstruction errors (7), the pairwise constraints on strongly connected nodes (8), the marginal MMD (9), the conditional MMD (11), and a L2-norm regularization

### Algorithm 1: CDNE

**Input:** Source network $G^s = (V^s, E^s, X^s, Y^s, A^s)$ and target network $G^t = (V^t, E^t, X^t, Y^t, A^t)$.

1. Greedy layer-wised training for SAE-s:
   - Set $H_t^{(0)} = X^t$ for $k = 1$.
     1.1. Leverage $H^{(k-1)}$ as input to $k$-th layer of SAE-s;
     1.2. Given $H^{(k-1)}, X^s, Y^s$, optimize $k$-th layer of SAE-s by finding $\theta^{(k)}$.
     2. Predict fuzzy labels for nodes in $V_0^t$ based on attribute vector representations and obtain $\bar{Y}^t$.
   - Set $H_t^{(0)} = X^t$ for $k = 1$.
     3.1. Leverage $H^{(k-1)}$ as input to $k$-th layer of SAE-t;
     3.2. Given $H^{(k-1)}, X^t, \bar{Y}^t, H^{(k-1)}, Y^s$, optimize $k$-th layer of SAE-t by finding $\theta^{(k)}$.
   - End for

**Output:** Label-discriminative and network-invariant node vector representations for $G^s$ and $G^t$, i.e., $H_s^{(l)}$ and $H_t^{(l)}$.

$$\Omega^{(k)} = \frac{1}{2} (||W_{1}^{(k)}||_2^2 + ||W_{2}^{(k)}||_2^2),$$

the overall loss function of SAE_t is defined as:

$$\mathcal{J}^{(k)} = \mathcal{R}^{(k)} + \alpha^{(k)} \mathcal{C}^{(k)} + \mu^{(k)} \mathcal{M}_{\delta}^{(k)} + \gamma^{(k)} \mathcal{M}_{\mathcal{C}}^{(k)} + \chi^{(k)} H_t^{(k)}$$

(12)

where $\alpha^{(k)}, \mu^{(k)}, \gamma^{(k)}$ and $\chi^{(k)}$ are the trade-off parameters to balance the effects of different terms. To optimize CDNE, SAE_s and SAE_t should be trained in sequence, as shown in Algorithm 1. To optimize SAE_s or SAE_t, one can use stochastic gradient descent (SGD) to optimize each $k$-th layer of a SAE, and employ a greedy layer-wise training approach [38] until reaching the deepest $l$-th layer so as to learn the deepest hidden matrix representation.

### 5 Experiments

In this section, we conduct experiments to investigate the following questions: 1) whether existing single-network embedding algorithms can be tailored to the cross-network node classification problem? 2) what types of information would be beneficial/adverse to cross-network node classification? 3) how effective is the proposed CDNE model over the baselines? 4) how sensitive is CDNE over the parameters?
The proposed CDNE model was evaluated in five real-world network datasets, as shown in Table 1. Blog1 and Blog2 are two disjoint subnetworks extracted from the BlogCatalog dataset [39], where a node represents a blogger and an edge indicates the friendship between two bloggers. Each node is associated with some attributes, i.e., keywords extracted from blogger’s self-description. Each node has a label indicating the blogger’s interested group. Since these two networks were extracted from the same original network, they share the same set of node attributes, and the attribute distributions between the two networks are very similar. To enlarge the cross-network distribution discrepancy, in each network, we randomly altered 30% of non-zero attribute values to be zeroes and randomly altered 30% of zero attribute values to be “1” so as to simulate the incomplete and noisy attributed information across networks.

In addition, Citation1, DBLPv7 and ACMv9 are three citation networks, where each node represents a paper and each edge indicates the citation of one paper to another. We extracted Citation1, DBLPv7 and ACMv9 from the datasets provided by ArnetMiner [40], which are from different original sources, i.e., Microsoft Academic Graph, DBLP and ACM respectively. Also, in our pre-processing step, the papers we keep in Citation1, DBLPv7 and ACMv9 were published before year 2008, between year 2004 and 2008, and after year 2010, respectively. One paper can have multiple labels indicating its research topics. The labels include “Database, Artificial Intelligence, Computer Vision, Information Security, and Networking”. The sparse bag-of-words features extracted from the title of each paper were utilized as its attributes. Since these three networks were extracted from different sources and formed in different time periods, they do not share a common set of node attributes and the attribute distributions across these networks are inherently varied to some extent.

### 5.1.1 Datasets

The proposed CDNE model was evaluated in five real-world network datasets, as shown in Table 1. Blog1 and Blog2 are two disjoint subnetworks extracted from the BlogCatalog dataset [39], where a node represents a blogger and an edge indicates the friendship between two bloggers. Each node is associated with some attributes, i.e., keywords extracted from blogger’s self-description. Each node has a label indicating the blogger’s interested group. Since these two networks were extracted from the same original network, they share the same set of node attributes, and the attribute distributions between the two networks are very similar. To enlarge the cross-network distribution discrepancy, in each network, we randomly altered 30% of non-zero attribute values to be zeroes and randomly altered 30% of zero attribute values to be “1” so as to simulate the incomplete and noisy attributed information across networks.

In addition, Citation1, DBLPv7 and ACMv9 are three citation networks, where each node represents a paper and each edge indicates the citation of one paper to another. We extracted Citation1, DBLPv7 and ACMv9 from the datasets provided by ArnetMiner [40], which are from different original sources, i.e., Microsoft Academic Graph, DBLP and ACM respectively. Also, in our pre-processing step, the papers we keep in Citation1, DBLPv7 and ACMv9 were published before year 2008, between year 2004 and 2008, and after year 2010, respectively. One paper can have multiple labels indicating its research topics. The labels include “Database, Artificial Intelligence, Computer Vision, Information Security, and Networking”. The sparse bag-of-words features extracted from the title of each paper were utilized as its attributes. Since these three networks were extracted from different sources and formed in different time periods, they do not share a common set of node attributes and the attribute distributions across these networks are inherently varied to some extent.

### 5.1.2 Implementation Details

In the CDNE model, we built a 2-layer SAE for both SAE_s and SAE_t, with the number of hidden dimensions set as $d(1) = 256$ and $d(2) = 128$ for the 1-st layer and the 2-nd layer of SAE, respectively. In SAE_s and SAE_t, we set the maximum step of neighbors as $K=6$, the ratio of reconstruction penalty as $\beta = 4$, the weight of $L2$-norm regularization as $\lambda^{(k)} = \lambda = 0.05$, and the weight of pairwise constraints on strongly connected nodes as $a^{(1)} = a^{(2)} = a = 4$ and $a^{(1)} = a^{(2)} = a/2$. In SAE_s, we set the weight of pairwise constraints on labeled nodes as $\varphi^{(1)} = \varphi = 2$ and $\varphi^{(2)} = 0$. In SAE_t, we set the weight of marginal MMD as $\mu^{(1)} = \mu = 2$ and $\mu^{(2)} = \mu/2$. In addition, we set the weight of conditional MMD as $\gamma^{(1)} = \gamma = 50$, $\gamma^{(2)} = \gamma/2$ for the two cross-network node classification tasks between Blog1 and Blog2, and set $\gamma = 40$ for the six cross-network node classification tasks among Citation1, DBLPv7 and ACMv9.

### 5.1.3 Baselines

No existing cross-network embedding algorithm has been developed for the cross-network node classification task. In order to systematically investigate what factors would be beneficial or adverse to cross-network node classification, we carefully select different types of related algorithms as the baselines, including:

1. **Attribute-only**: PCA [37] is a straightforward baseline which extracts low-dimensional attribute vector representations from the original attribute matrix.

2. **Plain network structure**: DeepWalk [9] employs Depth-first Sampling to generate truncated random walks. By regarding a node in a network as a word in a document, it extends Skip-Gram model to learn low-dimensional node vector representations with neighborhood preservation. DNE-APP [10] utilizes a semi-supervised SAE to reconstruct the aggregated $K$-step PPMI matrix and map nodes with higher proximities closer in the embedding space.

3. **Attribute & Network structure**: NetTr [32] is a network transfer learning algorithm which employs NMTF to project the label propagation matrices of the source network and the target network into a common latent space so as to learn the shared latent structural features. The latent structural features and low-dimensional attributes extracted by PCA are concatenated as its final node vector representations. ANRL [25] leverages a neighborhood enhancement autoencoder and an attribute-aware skip-gram model to learn network representations from both network structures and node attributes. Its best model variant, i.e., “WAN” was chosen.

4. **Network structure & Attribute & Label**: LANE [23] employs EVD to jointly project node labels, network structures and node attributes into a unified embedding space so as to capture the correlations among them. The proposed CDNE model leverage two SAEs to embed the cross-network nodes into a unified low-dimensional latent space. The single-network structures are utilized to capture within-network proximities, while the available node attributes and labels are leveraged to capture cross-network proximities.

For fair comparisons, we set the same dimensionality of the node vector representations learned by each baseline as $d=128$. In addition, in our experiments, we utilize the PPMI matrix with the same $K$-step to capture network connections, in DNE-APP, LANE and CDNE. Besides, DeepWalk and ANRL both leverage the Skip-Gram model [19], which has been proved to be equivalent as performing factorization on the PPMI matrix [7, 20, 22]. Note that DeepWalk, DNE-APP, ANRL and LANE were originally developed for a single-network scenario. To tailor them to cross-network node classification, we construct a unified network containing all the nodes in the source network and the target network.

### Table 1: Statistics of the real-world network datasets.

| Dataset   | #Nodes | #Edges | #Attributes | #Union Attributes | #Labels |
|-----------|--------|--------|-------------|-------------------|--------|
| Blog1     | 2300   | 33471  | 8189        | 8189              | 6      |
| Blog2     | 2896   | 53836  | 8189        |                   |        |
| Citation1 | 8935   | 15113  | 5349        |                   |        |
| DBLPv7    | 5484   | 8130   | 4412        |                   |        |
| ACMv9     | 9360   | 15602  | 5571        |                   |        |

1. https://github.com/xhuang31/LANE
2. https://www.aminer.cn/citation
Table 2: Cross-network node classification between Blog1 and Blog2 when only 1% of labeled nodes are available in the target network. “T” indicates that only leveraging the labeled nodes in the target network for training, while “S+T” indicates that leveraging the labeled nodes from both the source network and the target network for training. The highest F1 scores among all the comparing algorithms are shown in Boldface.

| Metrics | $g^s$ | $g^t$ | Algorithms |
|---------|-------|-------|------------|
|         | CDNE  | NetTr | PCA        | DeepWalk | DNE-APP | ANRL | LANE |
|         | S+T   | T     | S+T        | T        | S+T     | T    | S+T  |
| Micro-F1| Blog1  | Blog2 | 0.6650     | 0.5061   | 0.2508  | 0.5063 | 0.3272 | 0.2856 | 0.3531 | 0.2721 | 0.3579 | 0.4837 | 0.2858 | 0.5503 |
|         | Blog2  | Blog1 | 0.6437     | 0.5232   | 0.2338  | 0.5243 | 0.3434 | 0.2483 | 0.3676 | 0.2879 | 0.3615 | 0.4667 | 0.2513 | 0.5457 |
| Macro-F1| Blog1  | Blog2 | 0.6611     | 0.4905   | 0.2398  | 0.4987 | 0.2717 | 0.2544 | 0.2980 | 0.2317 | 0.2987 | 0.4309 | 0.2157 | 0.5391 |
|         | Blog2  | Blog1 | 0.6398     | 0.5143   | 0.2253  | 0.5167 | 0.2979 | 0.2323 | 0.3251 | 0.2360 | 0.3156 | 0.4623 | 0.1667 | 0.5356 |

Table 3: Cross-network node classification among Citation1, DBLPv7 and ACMv9 when only 1% of labeled nodes are available in the target network.

| Metrics | $g^s$ | $g^t$ | Algorithms |
|---------|-------|-------|------------|
|         | CDNE  | NetTr | PCA        | DeepWalk | DNE-APP | ANRL | LANE |
|         | S+T   | T     | S+T        | T        | S+T     | T    | S+T  |
| Micro-F1| DBLPv7 | Citation1 | 0.8110 | 0.5927 | 0.4663 | 0.6080 | 0.6773 | 0.4847 | 0.7128 | 0.6107 | 0.5381 | 0.6710 | 0.4534 | 0.5911 |
|         | ACMv9 | 0.8122 | 0.5886 | 0.6071 | 0.4260 | 0.3100 | 0.5787 | 0.6165 | 0.5517 | 0.4761 | 0.6630 | 0.4063 | 0.5878 | 0.5932 |
|         | Citation1 | DBLPv7 | 0.7310 | 0.5984 | 0.4256 | 0.5960 | 0.5837 | 0.4078 | 0.6165 | 0.5517 | 0.4761 | 0.6630 | 0.4063 | 0.5878 |
|         | ACMv9 | 0.7471 | 0.5641 | 0.5915 | 0.3974 | 0.5983 | 0.5970 | 0.4668 | 0.5672 | 0.4989 | 0.6465 | 0.4268 | 0.5707 | 0.5625 |
|         | Citation1 | ACMv9 | 0.7997 | 0.5769 | 0.4435 | 0.5915 | 0.5983 | 0.5970 | 0.4668 | 0.5672 | 0.4989 | 0.6465 | 0.4268 | 0.5707 | 0.5456 |
|         | DBLPv7 | ACMv9 | 0.7801 | 0.5651 | 0.5707 | 0.5903 | 0.5970 | 0.4668 | 0.5672 | 0.4989 | 0.6465 | 0.4268 | 0.5707 | 0.5456 |
| Macro-F1| DBLPv7 | Citation1 | 0.7925 | 0.5559 | 0.3975 | 0.5803 | 0.6125 | 0.4297 | 0.6509 | 0.5609 | 0.4307 | 0.6385 | 0.3581 | 0.5625 |
|         | ACMv9 | 0.7901 | 0.5551 | 0.3975 | 0.5803 | 0.6125 | 0.4297 | 0.6509 | 0.5609 | 0.4307 | 0.6385 | 0.3581 | 0.5625 |
|         | Citation1 | DBLPv7 | 0.6956 | 0.5500 | 0.3180 | 0.5538 | 0.4996 | 0.2724 | 0.5429 | 0.4945 | 0.3660 | 0.6306 | 0.2496 | 0.5530 |
|         | ACMv9 | 0.7251 | 0.4987 | 0.5352 | 0.3502 | 0.5429 | 0.4945 | 0.3660 | 0.6306 | 0.2496 | 0.5530 | 0.2496 | 0.5530 | 0.5667 |
|         | Citation1 | ACMv9 | 0.7952 | 0.5337 | 0.3735 | 0.5532 | 0.5903 | 0.3721 | 0.6625 | 0.5566 | 0.4100 | 0.6225 | 0.3107 | 0.5402 |
|         | DBLPv7 | ACMv9 | 0.7774 | 0.5141 | 0.5269 | 0.5332 | 0.3407 | 0.5332 | 0.3407 | 0.5332 | 0.6052 | 0.5090 | 0.5090 | 0.5090 |

network. Then, by utilizing the unified network for training, the cross-network proximities between nodes from the source network and the target network can be directly captured in the unified network.

5.2 Cross-network Node Classification

For cross-network node classification, we firstly randomly split the target network nodes into a training set and a testing set, where the training fraction should be rather small. Then, we train a one-vs-rest LR classifier based on the fully labeled nodes in the source network and the scarce labeled nodes in the target network. Next, we leverage the classifier to predict the labels of the testing nodes in the target network. The Micro-F1 and Marco-F1 scores [41] are employed as two evaluation metrics. The reported F1 scores are the average over the same 5 random splits, for each comparing algorithm.

Firstly, as shown in Tables 2 and 3, in all the eight cross-network node classification tasks, the proposed CDNE model always significantly outperform all the baselines. For example, for cross-network node classification from Blog1 to Blog2, CDNE achieved a 20.84% higher Micro-F1 score and a 22.63% higher Macro-F1 score than the best baseline, i.e., LANE (S+T).

Secondly, for the baselines which solely consider plain network structures, i.e., DeepWalk and DNE-APP, leveraging the cross-network information for training would lead to even much worse performance than only utilizing 1% of labeled nodes in the target network for training. This could be explained by the fact that network structures are not generalized across different networks. Thus, only considering such incomparable structural features would lead to negative transfer in cross-network node classification. On the other hand, network structures are indeed comparable within a single network. As a result, these single-network embedding algorithms based on plain network structures can perform well for single-network node classification, while would be rather unsuitable for cross-network node classification.

Thirdly, we can see that all the baselines considering node attributes, i.e., PCA, ANRL and LANE, can be better tailored to the cross-network node classification task, i.e., leveraging the cross-network information yields much better performance than only leveraging the target-network information for training. This demonstrates that node attributes are indeed comparable across different networks. Moreover, we found that PCA (S+T) performs even better than NetTr. This is because NetTr employs NMTF to project nodes from both the source network and the target network into a unified latent space so as to learn the common latent structural features. However, it is not guaranteed that the same labeled nodes from different networks would have similar topological structures, when node labels depend on homophily effect rather than structural identity. Thus, NetTr concatenating such incomparable network structural features and node attributes
would degrade the performance of the PCA baseline which only utilizes node attributes.

Next, we discuss the performance of CDNE w.r.t. PCA, ANRL and LANE. Firstly, the result of PCA (S+T) reflects the accuracy of fuzzy labels predicted by a LR classifier based on cross-network attributes. We can see that CDNE always achieves a significant gain on PCA (S+T) in all the cross-network node classification tasks. This is because the attributed information can be noisy and incomplete across different networks, thus, PCA (S+T) solely based on cross-network attributes would have a greater limitation as the attribute discrepancy become larger across networks. In the proposed CDNE model, we not only leverage the attributed information to measure the proximities between nodes across different networks, but also utilize the single-network structures to measure the proximities between nodes within each network. By integrating such cross-network attributed proximities and within-network structural proximities, CDNE can alleviate the negative effects caused by the noisy or incomplete cross-network attributed information. Secondly, the better performance of CDNE over ANRL (S+T) could be explained by two folds. On one hand, CDNE leverages the labeled information to align the same labeled nodes across networks to have similar latent features, while such labeled information has not been considered by ANRL. On the other hand, ANRL utilizes a neighbor enhancement autoencoder to model the attributed information, which makes it focus more on preserving the attributed proximities between connected nodes rather than the disconnected ones. Nodes across different networks do not have network connections, thus, ANRL would have the limited ability to well capture the cross-network attributed proximities. Thirdly, both LANE and CDNE have considered network structures, node attributes and node labels. However, LANE (S+T) performs much worse than CDNE. This could be explained by the fact that the deep learning framework adopted by CDNE is more powerful to capture the non-linear properties of the complex network structures than the linear matrix factorization approach utilized by LANE.

5.3 Parameter Sensitivity

In this subsection, we analyze the sensitivity of the parameters $K, \phi, \mu, \gamma, l, d$ on the performance of CDNE for cross-network node classification.

Parameter $K$ denotes the maximum number of step neighbors utilized to capture the structural proximities between nodes within a network. As shown in Figure 2(a), as $K$ increases, both Micro-F1 and Macro-F1 scores would increase. This reflects that high-order proximities are indeed helpful for learning informative feature vector representations for node classification, such same finding has also been observed in [7, 10].

Parameter $\phi$ denotes the weight of pairwise constraint on labeled nodes in the source network. As shown in Figure 2(b), $\phi > 0$ always lead to much higher F1 scores than $\phi = 0$. This indicates that mapping nodes sharing more common labels closer while those belonging to completely different categories far apart is effective for learning label-discriminative feature vector representations for node classification.

Parameters $\mu$ and $\gamma$ represent the weight of the marginal MMD and the class-conditional MMD constraints in SAE_t, respectively. As shown in Figures 2(c) and 2(d), the positive values of $\mu$ and $\gamma$ always lead to better performance than zero values. This demonstrates that incorporating the MMD constraints is effective for learning network-invariant node vector representations. Moreover, we can see that the effectiveness of the conditional MMD is much more significant than that of the marginal MMD. This is because by minimizing the conditional MMD, we can make each category of nodes across networks have similar latent vector representations, which would also indirectly minimize the marginal MMD between the source network and the target network.

Parameter $l$ denotes the number of layers of SAE in SAE_s and SAE_t. As shown in Figure 2(e), a 2-layer SAE can achieve better performance than a shallow architecture, i.e., 1-layer basic autoencoder. However, a deeper architecture which is more than 2 layers would yield even worse results. Parameter $d$ indicates the number of dimensions of node vector representation learned by CDNE. As shown in Figure 3(f), $d \in \{128, 256\}$ would lead to better performance.

6 Conclusions

Given a target network with scarce labeled nodes and a source network with fully labeled nodes, one can leverage the abundant labeled information from the source network to help classify the unlabeled nodes in the target network. In this paper, the first cross-network deep network embedding model, i.e., CDNE, is proposed to address the challenging cross-network node classification task. CDNE employs two SAEs to embed the nodes from the source network and the target network into a unified low-dimensional latent space. On one hand, CDNE captures the
structural proximities between nodes within a network and maps nodes with higher structural proximities to have more similar latent vector representations. On the other hand, CDNE utilizes the associated node attributes to predict fuzzy labels for the unlabeled nodes in the target network. Then, both the observed binary labels and the predicted fuzzy labels would be leveraged to guide the cross-network embedding process so as to align the same labeled nodes across networks to have similar latent vector representations. We found that in the cross-network node classification task, network topological structures are specific for a single network while node attributes are more comparable across different networks. Thus, by integrating the network-specific topological structures, and relatively network-invariant node attributes and node labels, both the within-network as well as cross-network proximities can be well preserved, and label-discriminative and network-invariant node vector representations can be learned by CDNE. Extensive experimental results demonstrate that the proposed CDNE model achieves significant gains over the related algorithms for node classification in the target network.

REFERENCES

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345-1359, 2010.

[2] Y. Chen, S. Song, S. Li, L. Yang, and C. Wu, "Domain space transfer extreme learning machine for domain adaptation," *IEEE Transactions on Cybernetics*, 2018.

[3] J. Li, K. Lu, Z. Huang, L. Zhu, and H. T. Shen, "Transfer independently together: A generalized framework for domain adaptation," *IEEE Transactions on Cybernetics*, 2018.

[4] S. Mao, X. Shen, and F.-l. Chung, "Deep domain adaptation based on multi-layer joint kernelized distance," in *Proceedings of the ACM SIGKDD Conference on Research and Development in Information Retrieval*, 2018, pp. 1049-1052.

[5] F. Wu and Y. Huang, "Sentiment domain adaptation with multiple sources," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 301-310.

[6] F. Wu, Z. Yuan, and Y. Huang, "Collaboratively training sentiment classifiers for multiple domains," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 7, pp. 1370-1383, 2017.

[7] S. Cao, W. Lu, and Q. Xu, "Grape: Learning graph representations with global structural information," in *Proceedings of ACM Conference on Information and Knowledge Management (CIKM)*, 2015, pp. 891-900.

[8] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, pp. 1145-1152.

[9] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 701-710.

[10] X. Shen and F.-l. Chung, "Deep network embedding with aggregated proximity preserving," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Network Analysis and Mining*, 2017, pp. 40-44.

[11] T. Sugiyama, M. Toda, and Y. Sugiyama, "Linearized classification with metric learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 513-520.

[12] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1225-1234.

[13] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 855-864.

[14] X. Shen and F.-l. Chung, "Deep network embedding for graph representation learning in signed networks," *IEEE Transactions on Cybernetics*, 2018.

[15] M. Heimann and D. Koutra, "On generalizing neural node embedding methods to multi-network problems," in *Proceedings of Mining and Learning with Graphs Workshop*, 2017.

[16] M. Heimann, H. Shen, and D. Koutra, "Node representation learning for multiple networks: The case of graph alignment," *arXiv preprint arXiv:1802.06257*, 2018.

[17] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, no. 1, pp. 415-444, 2001.

[18] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 385-394.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111-3119.

[20] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *IJCAI*, 2015, pp. 2111-2117.

[21] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Homophily, structure, and content augmented network representation learning," in *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, 2016, pp. 609-618.

[22] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Proceedings of Advances in Neural Information Processing Systems*, 2014, pp. 2177-2185.

[23] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *Proceedings of the ACM International Conference on Web Search and Data Mining*, 2017, pp. 731-739.

[24] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 119-128.

[25] Z. Zhang, H. Yang, J. Bu, S. Zhou, P. Yu, J. Zhang, M. Ester, and C. Wang, "ANRL: Attributed Network Representation Learning via Deep Neural Networks," in *IJCAI*, 2018, pp. 3155-3161.

[26] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2016*, pp. 1899-1901.

[27] J. Liang, P. Jacobs, I. Sun, and S. Parthasarathy, "Semi-supervised embedding in attributed networks with outliers," in *Proceedings of the 2018 SIAM International Conference on Data Mining*, 2018, pp. 153-161.

[28] H. Gao and H. Huang, "Deep Attributed Network Embedding," in *IJCAI*, 2018, pp. 3364-3370.

[29] J. Ye, H. Cheng, Z. Zhu, and M. Chen, "Predicting positive and negative links in signed social networks by transfer learning," in *Proceedings of the International Conference on World Wide Web*, 2013, pp. 1477-1488.

[30] J. Tang, T. Lou, and J. Kleinberg, "Inferring social ties across heterogeneous networks," in *Proceedings of the ACM International Conference on Web Search and Data Mining*, 2012, pp. 743-752.

[31] X. Shen, F.-l. Chung, and S. Mao, "Leveraging cross-network information for graph sparsification in influence maximization," in *Proceedings of the ACM SIGKDD Conference on Research and Development in Information Retrieval*, 2017, pp. 801-804.

[32] M. Fang, J. Yin, and X. Zhu, "Transfer learning across networks for collective classification," in *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 2013, pp. 161-170.

[33] B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena, "Don’t walk, skip! Online learning of multi-scale neighborhood embeddings," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Network Analysis and Mining*, 2017, pp. 258-265.

[34] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *Proceedings of Advances in Neural Information Processing Systems*, 2007, pp. 513-520.

[35] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2200-2207.

[36] Y.-H. Hubert Tsai, Y.-R. Yeh, and Y.-C. Frank Wang, "Learning cross-domain landmarks for heterogeneous domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5081-5090.

[37] A. Mackiewicz and W. Ratajczak, "Principal components analysis (PCA)," in *Computers and Geosciences*, vol. 19, pp. 303-342, 1993.

[38] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2007, pp. 153-160.

[39] J. Li, X. Hu, J. Tang, and H. Liu, "Unsupervised streaming feature selection in social media," in *Proceedings of the ACM International Conference on Information and Knowledge Management*, 2015, pp. 1041-1050.

[40] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 990-998.

[41] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to information retrieval*. Cambridge University Press, 2008.