Cross-domain Network Representations

Shan Xue[a], Jie Lu[a,*], Guangquan Zhang[a]

[a]*Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia*

Abstract

The purpose of network representation is to learn a set of latent features by obtaining community information from network structures to provide knowledge for machine learning tasks. Recent research has driven significant progress in network representation by employing random walks as the network sampling strategy. Nevertheless, existing approaches rely on domain-specifically rich community structures and fail in the network that lack topological information in its own domain. In this paper, we propose a novel algorithm for cross-domain network representation, named as CDNR. By generating the random walks from a structural rich domain and transferring the knowledge on the random walks across domains, it enables a network representation for the structural scarce domain as well. To be specific, CDNR is realized by a cross-domain two-layer node-scale balance algorithm and a cross-domain two-layer knowledge transfer algorithm in the framework of cross-domain two-layer random walk learning. Experiments on various real-world datasets demonstrate the effectiveness of CDNR for universal networks in an unsupervised way.

Keywords: network representation, transfer learning, random walk, information network, unsupervised learning, feature learning

1. Introduction

Networks generated from mature systems usually have larger numbers of entities such as nodes and edges than the emerging ones. For example, a new born online social media attracts limited numbers of users and hasn't formed massive interactions among them, from where the it gets extremely scarcer scale than the mature media like Facebook. Furthermore, in some domains such as the biological domain, it's difficult to collect sufficient data due to the costs, technique barriers, ethic reasons and so on. Traditional industries normally lack historical data when data-driven techniques haven't brought them benefits. Above scenarios lead to data deficiency which affect network analysis and learning. Previous approaches developed for network representation based on large-scale datasets are not able to be applied.

From the domain-specific view, rich data collected from real-world complex systems with large-scale network datasets. The components in a system are defined as the nodes in a network, direct interactions between nodes are defined as edges, and connection strengths are described by weights

---

*Corresponding author

Email addresses: shan.xue@student.uts.edu.au (Shan Xue), jie.lu@uts.edu.au (Jie Lu), guangquan.zhang@uts.edu.au (Guangquan Zhang)

on edges. Techniques not only analysis networks but also learn knowledge from network structures which has become a main stream in network research for artificial intelligence purposes [1, 2]. To this end, networks are preliminarily categorized based on real-world systems and their physical properties, such as social network [3, 4], biological network [5] and citation network [6]. As shown in Figure 1, social networks (a) denote users as nodes and friendship as edges; biological networks such as the Protein-Protein Interactions (PPI) network (b) models proteins as nodes and PPI as edges; and citation networks (c) represent papers as nodes and citations as edges.

From all kinds of networks, the information network [7] abstract the information flows from the original network structure, where the original nodes like users, proteins and authors are treated as the information users and suppliers and the information interchanges on friendship, PPI and citations as edges. The information network encode the network behaviors and save them into the network structure as shown in Figure 1(d). Information networks help us illustrate the entities in a physical system but raise a question on how to understand the various properties behind the different network categories especially when the topologies seem no difference as shown in Figure 1.

Network representation aims to learn a latent feature/vector space by learning from the information formed by network entities [8]. It inputs the high-dimensional network structures and outputs relatively low-dimensional representations in encoding as many community properties as possible. For the use of machine learning, network representation should output complex but highly structured latent features, to meet the smoothness requirement in learning function and to overcome the sparsity from input data [9]. To this end, a series of network representation approaches have been proposed based on the sampling strategy of random walks and the deep learning technique in the last decade. The random walk is a type of similarity measurement for a variety of problems in community detection [10, 11], which computes the local community structure information sub-linear to the size of the input network [12, 13]. A stream of short random walks is used as a basic tool for extracting information from real-world large-scale information networks [14, 15].

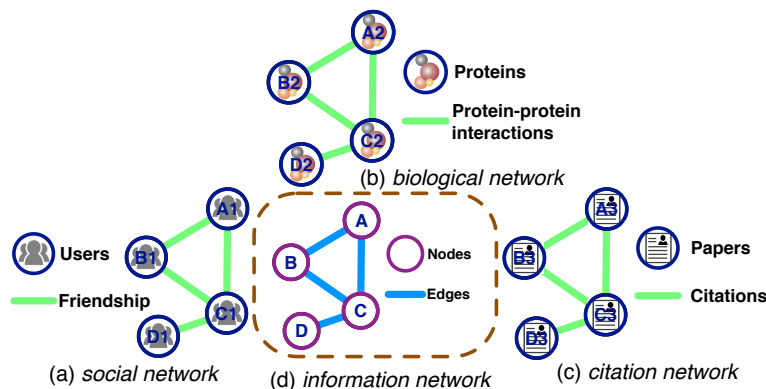The typical random walk-based network representation algorithms, such as DeepWalk [15], learn

Figure 1: Illustrations of undirected network structures formed by entities of nodes and edges. (a) Social network is formed by users {A1,B1,C1,D1} and user friendships; (b) Biological network is formed by proteins {A2,B2,C2,D2} and protein-protein interactions; and (c) Citation network is formed by papers {A3,B3,C3,D3} and citations. (d) Information network extracts information flows from (a), (b) and (c) as edges and inherit the nodes {A,B,C,D}.

sequences of nodes as a stream of short random walks to model the network structures of deep features which obviously are highly dependent on the sliced window that controls random walk learning for node sampling purpose. However, when the distance between two nodes is larger than the sliced window size, the random walk jumps to the next round. Although it could be covered by introducing a vast amount of sampling, the repetitions increase computational complexity. This explains the main reason why the networks with small structure scales are barely applicable for these algorithms. Therefore, the previous works on random walk-based network representations are limited in a domain-specific way so that the performance mainly relies on the network topological quality. Our previous work proposed a framework for transferring structures across large-scale information networks (FTLSIN) [16], however only enabled structural knowledge transfer across relational information networks and both networks should have large scales. The cases listed in the beginning of this paper will not be guaranteed satisfying latent feature spaces from the limited network structures for the further machine learning tasks within one domain.

To address above problems, we propose a novel algorithm for universal cross-domain network representations (CDNR) with the following contributions.

1) CDNR offers an effective learning solution for the network representation, where the network doesn't have enough entities that causes a random walk failure in structural sampling.
2) CDNR determines the relationships between two independent networks which would belong to irrelevant domains. Similar network patterns are detected so that links generated between the corresponding communities transfer knowledge in CDNR.
3) CDNR predicts the potential entities for the scarce network structures by employing the cross-domain two-layer random walk (*CD2L-RandomWalk*) framework from [16] and integrating two novel algorithms, cross-domain two-layer node-scale balance (*CD2L-NodeBalance*) and cross-domain two-layer knowledge transfer (*CD2L-KnowlTransfer*).

The rest of the paper is arranged as follows. In Section 2, related works are summarized. In Section 3, we state the CDNR problem. In Section 4, the proposed CDNR algorithm is explained in detail. In Section 5, two experiments are designed to evaluate the representations on real-world datasets. Our conclusions are presented in Section 6.

2. Related Works

The previously used per-node partition function [17] is expensive to compute, especially for large information networks. To overcome this disadvantage, a series of sampling strategies have been proposed [18, 19] to analyze the statistics within local structures, e.g., communities and sub-networks. These approaches are different from traditional representation learning [20, 21, 22]. The latent feature learning of the network representation captures neighborhood similarity and community membership in topologies [23, 24, 25].

DeepWalk [15] trains a neural language model on the random walks generated by the network structure. After denoting a random walk that starts from a root node, DeepWalk slides a window and maps the central node to its representation. Hierarchical Softmax factors out the probability distributions corresponding to the random walk and the representation function is updated to maximize the probability. DeepWalk has produced promising results in dealing with sparsity in scalable networks, but has relatively high computational complexity for large-scale information networks. LINE, Node2Vec and Struc2Vec are the other structure-based network representation algorithms that improve the performance of DeepWalk. LINE [7] preserves both the local network structure

and the global network structure by first-order proximity and second-order proximity respectively and can be applied to large-scale deep network structures that are directed, undirected, weighted and unweighted. Node2Vec [26] explores the diverse neighborhoods of nodes in a biased random walk procedure by employing classic search strategies. Struc2Vec [27] encodes structural similarities and generates the structural context for nodes using random walks. The above-mentioned works has contributed to network analysis by modeling a stream of short random walks.

All the previous works based on random walk to sample networks into a steam of nodes are under a common assumption of power-law distribution. The power-law distribution exists widely in real-world networks. It is a special degree distribution that follows $P(deg) \sim deg^{-a}$, where deg is a node degree and a is a positive constant [28]. A network that follows the power-law distribution is also regarded as a scale-free network with the scale invariance property [29]. The social networks, biological networks and citation networks being discussed in this paper are observed to be scale-free in nature [30]. In *log-log* axes, the power-law distribution shows a linear trend on the slope ratio of $-a$ (Figure 4 and Figure 6), which reflects that numerous edges connect small degree nodes and will not change regardless of network scale [31]. It has been observed in [15] that if a network follows the power-law distribution, the frequency at which a node undertakes in a short random walk will also follow the same distribution. Meanwhile, random walks in power-law distribution networks naturally gravitate towards high degree nodes [32].

In this paper, we propose CDNR which employs biased random walk sampling strategies to learn network structures based on previous works. However, CDNR is different from the deep transfer learning approaches for cross-domain graph-structured data, i.e., context enhanced inductive representation [33], intrinsic geometric information transfer [34] and deep inductive graph representation [35]. Deep neural network-based network representation usually need to generalize a small set of base feature for deep learning, such as network statistical properties like node degree, which lost valuable information from networks. The link predictions in *CD2L-RandomWalk* are therefore leveraged on the power-law distribution as well as the distance calculation between the two independent networks across domains. The network that has small distance to the target network is regarded as the source domain. The scale invariance property should theoretically ensure that power law-based CDNR is robust.

3. Problem Statement

Definition 1 (Domain [36]) A domain is denoted as $\mathcal{D} = \{\mathbf{X}, P(\mathbf{x})\}$, where $\mathbf{X}$ is the feature space and $P(\mathbf{x})$ is the marginal probability distribution that $\mathbf{x} = \{x_1, \cdots, x_n\} \in \mathbf{X}$.

Definition 2 (Network [30]) Let $G = (V, E, W)$ be a given network, where V represents one kind of entities known as nodes, E represents another kind of entities known as edges reflecting connections between nodes, $E \subseteq (V \times V)$, and W represents the possible weights on E.

Definition 3 (Cross-domain Network Representation) Suppose a source domain $\mathcal{D}^s = \{\mathbf{X}^s, P(\mathbf{x}^s)\}$ represented by $G^s = (V^s, E^s)$ and a target domain $\mathcal{D}^t = \{\mathbf{X}^t, P(\mathbf{x}^t)\}$ represented by $G^t = (V^t, E^t, W^t)$, domains are irrelevant $\mathcal{D}^s \neq \mathcal{D}^t$ if $\mathbf{X}^s \neq \mathbf{X}^t$ and $P(\mathbf{x}^s) \neq P(\mathbf{x}^t)$; and relevant if $\mathbf{X}^s = \mathbf{X}^t$ or $P(\mathbf{x}^s) = P(\mathbf{x}^t)$. CDNR employs structural information on W^t, from $G^s = (V^s, E^s)$ to $G^t = (V^t, E^t, W^t)$, to improve the target domain representations $f \colon V^t \to \mathbf{X}^t$ in a d-dimensional latent feature space.

To prepare the structural knowledge from the source domain, CDNR firstly implements a maximum likelihood optimization to generate a set of random walks $\mathcal{W}^s$ on G^s in the bottom layer of

CD2L-RandomWalk. A neighborhood $N_S(u^s) \subset G^s$ is clustered rooted at node u^s by the neighborhood sampling strategy S on the biased random walks [26]. Then, *CD2L-RandomWalk* constructs links between G^s and G^t. To this end, *CD2L-NodeBalance* balances the scales of V^s and V^t by clustering V^s into super nodes $V' \in \mathcal{V}$ in which $v^s \in V'$ share close node degrees with $v^t \in V^t$; and generate links between V^t and $\mathcal{V}$. *CD2L-KnowlTransfer* trains the maximized similarities across two domains and determines how much value should be transferred across the shortest paths $\mathcal{P}$ and E^t, where $\mathcal{P}$ are formed by the super edges E' and the values are save in W^t. In CDNR, the representations $\mathbf{X}^t$ are learned in the top layer of *CD2L-RandomWalk* and will be evaluated by a standard classification task.

3.1. Bottom-layer Random Walk: Knowledge Preparation

The bottom-layer random walk is designed for knowledge preparation in the source domain. The sampled random walks contains structural knowledge from which will be transferred to the target domain. The bottom-layer random walk introduces a biased random walk strategy to efficiently explore diverse neighborhoods and sample the nodes along the shortest path[1]. Suppose a set of random walks $\mathcal{W}^s$, each root node v^s repeats k times for sampling and each random walk is set in a length of l. In generating a random walk, suppose we are standing at node c which is the i-th node in the random walk, $1 < i < l$, the node $c-1$ denotes the $i-1$-th node and the $i+1$-th node x is chosen from $N_S(c)$ based on a probability $P(x|c) = \frac{\pi_{xc}}{Z}$ where Z is the partition function that ensures a normalized distribution [9] and $\pi_{xc} = \alpha_{pq}(x,c)$ is guided by the search bias α_{pq}. To be specific, $\alpha_{pq}(x,c)$ follows the searching rules: if the length of the shortest path between nodes x and $c-1$ is $|\mathcal{P}_{xc-1}| = 0$, then $\alpha_{pq}(x,c) = 1/p$; $\alpha_{pq}(x,c) = 1$, if $|\mathcal{P}_{xc-1}| = 1$; and $\alpha_{pq}(x,c) = 1/q$,

---

[1]The shortest path is a path between two nodes for which the sum of its edge weights is minimized.

Table 1: Summary of notations.

$\mathcal{D}^s$, $\mathcal{D}^t$	A source domain and a target domain.
$\mathbf{X}^t$, $\mathbf{x}^t$, x_i^t	A d-dimensional target domain latent feature spaces, a N^t-dimensional feature vectors, and the i-th element of $\mathbf{x}^t$.
G^s, G^t	A (un)directed unattributed unweighted network from $\mathcal{D}^s$, and a (un)directed unattributed weighted network from $\mathcal{D}^t$.
V^s, V^t, v^s, v^t	The node set of G^s, the node set of G^t, a node in V^s, and a node in V^t.
E^s, E^t, e_{ij}^s, e_{ij}^t	The edge set of G^s, the edge set of G^t, an edge between v_i^s and v_j^s, and an edge between v_i^t and v_j^t.
W^t, w_{ij}^t	The weight set on E^t, and a weight in W^t on e_{ij}^t.
$\mathcal{G}$, $\mathcal{V}$, E'	A form of super graph for G^s, a super-node set, and a super-edge set.
V', $e'_{V_i'V_j'}$	A super node in $\mathcal{V}$, and a super edge in E' connecting V_i' and V_j'.
W', $w'_{V_i'V_j'}$	A weight set, and a weight in W' on $e'_{V_i'V_j'}$.
$\mathcal{W}^s$, $\mathcal{W}^t$	The random walk sets on G^s and G^t.
$\mathcal{P}_{V_i'V_j'}$	A shortest path between V_i' and V_j' over $\mathcal{G}$.
deg^s, deg^t, deg'	A set of node degree values on G^s, a set of node degree values on G^t and a set of node degree values on $\mathcal{G}$.
E^*, W^*	A link set between G^t and $\mathcal{G}$ across domains, and a weight set on E^*.
$e^*_{v^tV'}$, $w^*_{v^tV'}$	A link in E^* that connects v^t and V', and a weight in W^* on $e^*_{v^tV'}$.
a	The slope ratio of power-law distribution.
$Deg(\cdot)$	The function calculates the node degree.
$\langle \cdot \rangle$	The average function on a value set.
$|\cdot|$	The function counts the number in a set.

if $|\mathcal{P}_{xc-1}| = 2$. The sampling strategy on the biased random walks is computationally efficient especially for real-world large-scale networks.

4. Knowledge Transfer in Cross-domain Network Representations

CDNR enables the cross-domain random walk-based network representations and assumes both networks across domains follow the power-law distribution. Representations in CDNR work under the Skip-gram framework and are optimized by maximum likelihood over biased random walks. The contributions of CDNR are realized in this section by *CD2L-RandomWalk* with the two components: *CD2L-NodeBalance* and *CD2L-KnowlTransfer*.

4.1. Cross-domain Two-layer Node-scale Balance and Link Prediction

By transferring knowledge from an external source domain, CDNR deals with the scenarios that the training sample in the target domain is insufficient to make a good network representation. Such knowledge transfer belongs to a transfer learning task [37] arises two questions: 1) Link prediction: how to construct paths between two networks across domains for *CD2L-RandomWalk*, and 2) *CD2L-NodeBalance*: how to solve the problem of unbalanced node scales.

The unbalancedness between G^s and G^t is reflected on the nodes $|V^s| > |V^t|$ and also on the connections $\langle deg^s \rangle > \langle deg^t \rangle$, where $|V^s|$ and $|V^t|$ refer to the node scales, and $\langle deg^s \rangle$ and $\langle deg^t \rangle$ refer to the average node degrees[2]. In this case, *CD2L-NodeBalance* tries to reform G^s into a smaller size based on the network structures of G^t. For the purpose of discovering sub-graph patterns [38], a concept of super node [39] is employed and we define the formation for CDNR.

Definition 4 (Super Node in Source Domain) A super node is a sub-graph of the original source network. Denoting the super-node set $\mathcal{V}$, a super node $V' \in \mathcal{V}$ consists of a group of nodes $\{v^s\} \subseteq V^s$ and the edges $\{e^s\} \subseteq E^s$ connecting to or from $\{v^s\}$. The nodes $\{v^s\}$ that clustered into a V' have close node degrees.

To cluster a set of nodes in the large-scale network, a super-node learning based on the nodes in the target domain is as follows:

$$\Phi_{Snode} \colon V^s \to \{V', e^*_{v^t V'}, w^*_{v^t V'} | V^t\} \tag{1}$$

where $e^*_{v^t V'}$ is a predicted link between $v^t \in V^t$ and $V' = \{v^s\}$ across domains, and $w^*_{v^t V'}$ is the weight on $e^*_{v^t V'}$ which indicates the similarity between v^t and V' and how much knowledge should be transferred from the source domain to the target domain in *CD2L-RandomWalk*.

CD2L-NodeBalance attempts to pair each node v^t with at least one super node V' in a minimum super-node scale $|\mathcal{V}|$ and a maximum likelihood between V^t and $\mathcal{V}$ according to Eq. (4). For each pair of (v^t, V'), we firstly initialize a link and a weight following,

$$e^*_{v^t V'} = \begin{cases} 1 & \text{if } w^*_{v^t V'} > 0 \\ 0 & \text{if } w^*_{v^t V'} = 0 \end{cases} \tag{2}$$

$$w^{*(0)}_{v^t V'} = \frac{\min(Deg(v^t), Deg(V'))}{\max(Deg(v^t), Deg(V'))} \tag{3}$$

---

[2]The average degree is a mean on the degrees of all nodes in the network.

where $Deg(v^t)$ denotes the degree of v^t, $Deg(V')$ denotes the degree of V', and V' is initialized on nodes in the same degree.

To optimize Φ_{Snode} in Eq. (4), we analysis the degree ranges over V^t and V' in $[1, \max(deg^t)]$ and $[1, \max(deg')]$ respectively and reorganize V' including merging and dividing super nodes based on the following three cases.

Denoting the range scales $n_{deg^t} = |deg^t|$ and $n_{deg'} = |deg'|$, there are three possible cases of *CD2L-NodeBalance* as follows and as shown in Figure 2. Degree sets deg^t and deg' are always ranked in a decreasing order. A v^t finds the corresponding V' that are in the same position in deg^t and deg', denoted as $Deg(v^t) \sim Deg(V')$.

Case 1: If $n_{deg^s} = n_{deg^t}$, only one V' links to v^t. In this case, *CD2L-NodeBalance* is completed in the initialization stage with $E^* = \{e^*_{v^t V'}\}$ and $W^* = \{w^*_{v^t V'}\}$.

Case 2: If $n_{deg^s} > n_{deg^t}$, more than one V' links to v^t. $W^* = \{w^*_{v^t V'}\}$ at the current stage is going to be optimized in Eq. (4). If $w^*_{v^t V'}$ turns to 0, the edge $e^*_{v^t V'}$ is deleted and the V' is merged into another super node that linked with v^t and gets the smallest weight.

Case 3: If $n_{deg^s} < n_{deg^t}$, there are at least one v^t not linked to any V'. We add a group of empty super nodes V'_{null} in a number of $n_{deg^t} - n_{deg^s}$ and evenly insert them into $\mathcal{V}$. To fill up the V'_{null}, a few nodes in $V' \neq \emptyset$ next to V'_{null} are removed and added to V'_{null}. V'_{null} then is initialized $w^{*(0)}_{v^t V'} = 0$ by Eq. (3).

In Case 2 and Case 3, Φ_{Snode} is optimized by maximizing the likelihood between V^t and $\mathcal{V}$. Starting from each v^t_i, a vector $\vec{w} = [w_1, \cdots, w_i, \cdots, w_{n_{node}}]^\top$ weights for each pair of (v^t_i, V'_j), where $i = 1, \cdots, |V^t|$, $j = 1, \cdots, n_{link}$ and $n_{link} = \max(n_{deg'}, n_{deg^t})$. If there is link between (v^t_i, V'_j), $w_i = w^*_{v^t V'}$; else wise 0.

$$\max_{\Phi_{Snode}} \sum_i \eta \sum_j \left[\log(C) - a_\oplus \log(\vec{\delta}_z \vec{w} \vec{w}^\top \vec{\delta}_z^\top) \right] \tag{4}$$

where $\vec{\delta}_z$ is a vector in size of n_{link} with the value of 0 or 1, which based on $Deg(v^t) \sim Deg(V')$ in Cases 1-3. Let $a_\oplus = \min\{a^s, a^t\}$ in which a^s and a^t are the power-law slope ratio of G^s and G^t respectively. $\eta = \frac{1}{n_{deg^t}} e^{\frac{1 - n^2_{deg'}}{n_{deg'}}} \gamma e^\lambda$ controls the range of the likelihood over *CD2L-NodeBalance*, where γ is a parameter for V^t and λ is a parameter for $\mathcal{V}$. The optimized *CD2L-NodeBalance*
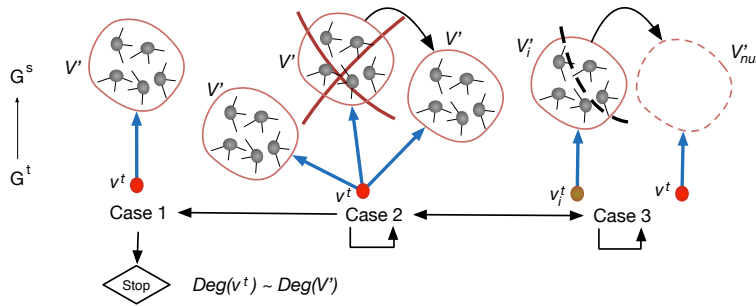

Figure 2: An illustration of the three cases in *CD2L-NodeBalance* and their interconversions: 1) Case 2 to Case 1, 2) Case 2 to Case 2, 3) Case 2 to Case 3, 4) Case 3 to Case 2 and 5) Case 3 to Case 3.

Algorithm 1 The *CD2L-NodeBalance* algorithm.

Input:
    $G^t = (V^t, E^t)$ in the target domain and $G^s = (V^s, E^s)$ in the source domain.
Initialize:
    $V'^{(0)} \leftarrow$ Cluster $v^s \in V^s$ by node degrees.
    $n_{deg^s}^{(0)}, n_{deg^t}^{(0)} \leftarrow$ Node degree scales in G^s and G^t.
    $W^{*(0)} \leftarrow$ Apply Eq. (3).
1: **while** $n_{deg^s}^{(t)} \neq n_{deg^t}^{(t)}$ **do**
2:     $W^{*(t)} \leftarrow$ Apply Eq. (4)
3:     $E^{*(t)} \leftarrow$ Apply Eq. (2)
4: **end while** $\{\epsilon = \max_{\Phi_{Snode}}^{(t)} - \max_{\Phi_{Snode}}^{(t-1)}\}$
5: **return** $E^* = E^{*(t)}$ and $W^* = W^{*(t)}$

---

results suggest the predicted links $E^* \propto \vec{\delta}_z$ where $W^* = \vec{\delta}_z \vec{w}$ in Case 2 and Case 3.

4.2. Cross-domain Two-layer Knowledge Transfer and Target Domain Edge Evolvement

    CD2L-KnowlTransfer transfers the knowledge saved in weights through the predicted links E^*. The knowledge includes three parts of weights as shown in Figure 3: a weight on the super edge that reflects the knowledge learning from the random walks in the source domain, two weights on the predicted links, and the original weight on e^t (in this paper is 1 or 0). The *CD2L-KnowlTransfer* follows:

$$\Phi_{Knowl} \colon (G^t, \mathcal{G}, W^*) \to W^t \qquad (5)$$

where $\mathcal{G}$ denotes the super graph.

    Definition 5 (Super Graph in Source Domain) A super graph $\mathcal{G} = (\mathcal{V}, E', W')$ reformed from G^s is formed by super nodes $\mathcal{V} = \{V'\}$, super edges $E' = \{e'_{V_i' V_j'}\}$ and the super weights $W' = \{w'_{V_i' V_j'}\}$ on E', where $\mathcal{F} \colon (\mathcal{W}^s, \mathcal{V}) \to E'$. If a random walk belongs to $\mathcal{W}^s$ goes through V_i' and V_j', there will be an $e'_{V_i' V_j'}$.

$$w'_{V_i' V_j'} = \sum_{v_i^s \in V_i'} \sum_{v_j^s \in V_j'} \frac{1}{d_{\mathcal{W}^s}(v_i^s, v_j^s)} \qquad (6)$$

Algorithm 2 The *CD2L-KnowlTransfer* algorithm.

Input:
    $\mathcal{W}^s$ Random walks of G^s generated in the bottom-layer of *CD2L-RandomWalk*; $G^t = (V^t, E^t, W^{t(0)})$ in the target domain; and E^* and W^* from Algorithm 1.
1: **for** e_{ij}^t in E^t **do**
2:     $w'_{V_i' V_j'} \leftarrow$ Apply Eq. (6).
3:     $\mathcal{P}_{V_i' V_j'} \leftarrow$ Construct shortest paths between V_i' and V_j'.
4:     $w_{ij}^t \leftarrow$ Update weight on e_{ij}^t by Eq. (7).
5:     $e_{ij}^t \leftarrow$ Evolve new edge if $w_{ij}^{t(0)} = 0$ and $w_{ij}^t > 0$.
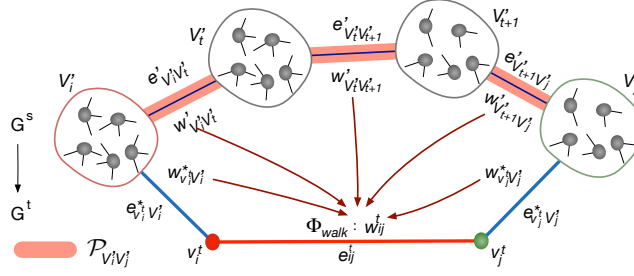6: **end for**
7: **return** $G^t = (V^t, E^t, W^t)$

Figure 3: An illustration of weight contributions on a target domain network edge in *CD2L-KnowlTransfer*.

Algorithm 3 The CDNR algorithm.

CD2L-RandomWalk

1: $\mathcal{W}^s \leftarrow$ Random walks generated from G^s in the Bottom-layer Random Walk.

2: E^*, $W^* \leftarrow$ Apply Algorithm 1.

3: $W^t \leftarrow$ Apply Algorithm 2.

Top-layer Feature Learning

1: **for** u^t in G^t **do**

2:    $N_S(u^t) \leftarrow$ Search neighborhood of u^t with W^t.

3:    $f \leftarrow$ Apply Skip-gram to optimize.

4: **end for**

5: **return** $\mathbf{X}^t \leftarrow$ A latent feature space of G^t by f.

---

where $d_{\mathcal{W}^s}(v_i^s, v_j^s)$ is the distance between nodes v_i^s and v_j^s in a random walk, and $w'_{V_i'V_j'}$ calculates every random walk going over $e'_{V_i'V_j'}$.

In having the three parts of weights $\{w'_{V_i'V_j'}, w^*_{v_i^t V_i'}, w^*_{v_j^t V_j'}, w_{ij}^{t(0)}\}$ that contribute to w_{ij}^t in *CD2L-KnowlTransfer*, the weight on $e_{ij}^t \in E^t$ in the top layer of the *CD2L-RandomWalk* are denoted as:

$$w_{ij}^t = w_{ij}^{t(0)} + \frac{1}{Z} \sum_{V_i'} \sum_{V_j'} w^*_{v_i^t V_i'} \cdot w^*_{v_j^t V_j'} \cdot \Big[\frac{1}{l_\mathcal{P}} \sum_{e'_{V_t'V_{t+1}'} \subseteq \mathcal{P}_{V_i'V_j'}} w'_{V_t'V_{t+1}'}\Big] \tag{7}$$

where $w_{ij}^{t(0)} = \{0, 1\}$ is the original weight that reflects an edge between v_i^t and v_j^t or no edge, Z is for normalization, $\mathcal{P}_{V_i'V_j'}$ is the shortest path between V_i' and V_j' over $\mathcal{G}$, $l_\mathcal{P}$ is the length of $\mathcal{P}_{V_i'V_j'}$, $e'_{V_t'V_{t+1}'} \in E'$ denotes an edge that consists in $\mathcal{P}_{V_i'V_j'}$, and $w'_{V_t'V_{t+1}'}$ is the weight on $e'_{V_t'V_{t+1}'}$.

In *CD2L-KnowlTransfer* above, G^t is enriched in network structures by putting extra weights on the original edges and also evolves possible edges.

4.3. Top-layer Random Walk and Network Representations

CDNR represents G^t in the top layer of *CD2L-RandomWalk* after *CD2L-NodeBalance* and *CD2L-KnowlTransfer*. CDNR learns the latent feature space by $f: V^t \rightarrow \mathbf{X}^t$ in the Skip-gram framework.

Given a node u^t in the target domain with the window size r, we obtain a cross-domain Skipgram for G^t by maximizing the following log-likelihood function of f in observing a neighborhood of $N_S(u^t)$,

$$\max_f \sum_{u^t \in V^t} \log Pr(N_S(u^t)|f(u^t)) \tag{8}$$

where $\mathcal{W}^t$ is learned on $P(x^t|u^t) = \frac{\pi_{x^t u^t}}{Z}$ that $\pi_{x^t u^t} = \alpha_{pq}(x^t, u^t) \cdot w^t_{x^t u^t}$, while $\mathcal{W}^s$ is learned on $\pi_{x^s u^s} = \alpha_{pq}(x^s, u^s)$.

In summary, Algorithm 3 of CDNR is formed by *CD2L-RandomWalk* and *Top-layer Feature Learning*. The main advantage of CDNR is that when the network representation is poor because it lacks structures, the *CD2L-RandomWalk* enables knowledge transfer from external domains and CDNR doesn't need to rebuild a network representation model. CDNR offers an efficient cross-domain learning with a relatively low computational cost of $O(\langle deg^t \rangle |V^t|)$ on *CD2L-NodeBalance*, $O(|E^t|)$ on *CD2L-KnowlTransfer* and $O(\langle deg^t \rangle^2 |V^t|)$ on network representation in line with Node2Vec [26].

5. Experiments

This section evaluates the effectiveness of the CDNR compared to the baseline algorithms of network representations in both single-label classifications (Section 5.2) and multi-label classifications (Section 5.3).

5.1. Baseline Algorithms

This experiment evaluates the performance of the unsupervised CDNR on the target networks. The representation outputs are applied to a standard supervised learning task, i.e., linear SVM classification [40]. The experiments choose a simple classifier because we want to put less emphasis on classifiers in evaluate the network representation performance. The baseline algorithms are chosen from the previous domain-specific network representations and a deep inductive graph representation as follows.

- **DeepWalk** (Perozzi *et al.* 2014) [15] is the first random walk-based network representation algorithm. By choosing DeepWalks, we exclude the matrix factorization approaches which have already been demonstrated to be inferior to DeepWalk.
- **LINE** (Tang *et al.* 2015) [7] learns latent feature representations from large-scale information networks by an edge-sampling strategy in two separate phases of first- and second-order proximities. We excluded a recent Graph Factorization algorithm [41] because LINE demonstrated better performance in the previous experiment.
- **Node2Vec** (Grover *et al.* 2016) [26] learns continuous feature representations of nodes using a biased random walk procedure to capture the diversity of connectivity patterns observed in networks with the biased parameter α which is controlled by parameters of p and q.
- **Struc2Vec** (Ribeiro *et al.* 2017) [27] learns node representations from structural identity by constructing a hierarchical graph to encode structural similarities and generating a structural context for nodes.
- **DeepGL** (Rossi *et al.* 2018) [35] learns interpretable inductive graph representations by relational functions for each representing feature and achieve inductive transfer learning across networks. It inputs a 3-dimensional base features to a CNN and outputs the representation in d dimensions where d depends on learning.

5.2. Experiment on Single-label Dataset

5.2.1. Single-label Datasets

Two academic citation networks are selected as the datasets. Both of them are used for the multi-class classification problem [42]. Nodes are denoted as papers in these networks.

Table 2: Single-label classification dataset statistics.

Domain	Datasets	Num. of Nodes	Num. of Edges	Num. of Categories
Source	DBLP	60,744	52,890	4
Target	M10	10,310	77,218	10

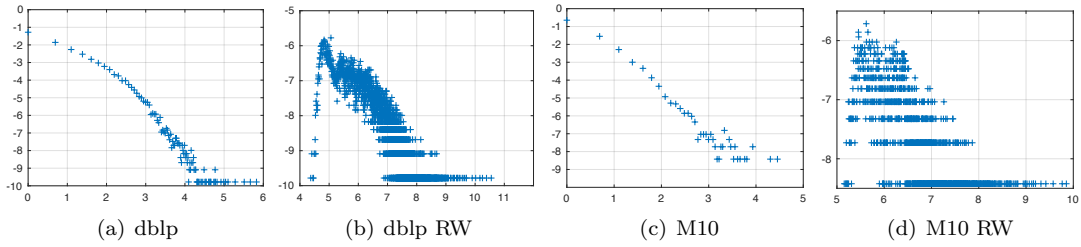(a) dblp   (b) dblp RW   (c) M10   (d) M10 RW

Figure 4: Power-law distribution of the single-label classification datasets and their random walks on the networks. The X-axial is denoted as $\log(Deg)$ of the network and the Y-axial is denoted as $\log(Pr(Deg))$. Each power-law distribution pair of the network and its random walks should follow the same pattern so that random walks over the network can conduct skip-gram based network representations. For example, (a) and (b) are formed as a power-law distribution pair following the same pattern by which random walks on the dblp network are guaranteed a network representation on dblp.

- **DBLP dataset**[3] (source network) consists of bibliographic data in computer science. Each paper may cite or be cited by other papers, naturally forming a citation network. The network in this dataset abstracts a list of conferences from four research areas, *i.e.,* database, data mining, artificial intelligence and computer vision.
- **CiteSeer-M10 dataset**[4] (target network) is a subset of CiteSeerX data which consists of scientific publications from 10 distinct research areas, *i.e.,* agriculture, archaeology, biology, computer science, financial economics, industrial engineering, material science, petroleum chemistry, physics and social science.

5.2.2. Experiment Setup

For the evaluations, we randomly partition the dataset in the target domain into two non-overlapping sets for training and testing by nine groups of training percentages, $\{0.1, 0.2, \cdots, 0.9\}$. We repeat the above steps 10 times and thus obtain 10 copies of the training data and testing data. The reported experimental results are the average of the 10 runs and their variance.

---

[3]http://arnetminer.org/citation (V4 version is used)
[4]http://citeseerx.ist.psu.edu/

The parameters of CDNR are set in line with typical values used for DeepWalk [15], LINE [7], Node2Vec [26] and Struc2Vec [27]. For networks in both the source domain and the target domain, let the dimensions of feature representation be $d = 128$, the walk length be $l = 80$, the number of walks of every source node be $k = 10$, the window size be $r = 10$, $workers = 8$, and the search bias α be with $p = 1$ and $q = 1$. Let the learning rate ρ start from 0.025 as in [7] and the convergence track on 0.1 in our experiment. For Struc2Vec, let OPT1 (reducing the length of degree sequences), OPT2 (reducing the number of pairwise similarity calculations) and OPT3 (reducing the number of layers) all in values of True, and the maximum number of layers be 6. The parameters in *CD2L-NodeBalance* is set as $\gamma = 100$ and $\lambda = 100$. In these settings, the total number of random walks over an input network is $w = SampleSize \times k$ and the size of the random walks is $w \times l$. For DeepGL [35], the operator is chosen from {mean, sum, maximum, Hadamard, Weight L^p, RBF} which gets best results in base feature learning; L^p is set in 1; feature similarity threshold is set in 0.01; maximum depth of layer is set in 10; and convergence for feature diffusion is set in 0.01.

5.2.3. Single-label Classification

(a) Node2Vec on dblp (b) CDNR on M10

(c) PCA on M10 (d) LLE on M10 (e) Laplacian on M10

Figure 5: Network representation on dblp and M10 in a 2-dimensional latent feature space.

We use Macro-F1 and Micro-F1 [43] to evaluate classification performances and the results are shown in Table 3. The F1 scores are designed to evaluate the effectiveness of category assignments [44].

$$F_1(r, p) = \frac{2rp}{r + p} \tag{9}$$

We use the indicators of true positive (tp), false positive (fp) and false negative (fn) to measure

12

the standard recall (r) and precision (p). In $Micro\_F_1$, let $r = \frac{\sum tp}{\sum tp + \sum fn}$ and $p = \frac{\sum tp}{\sum tp + \sum fp}$. The Micro_F1 score computes the global $n \times m$ binary decisions, where n is the number of total test nodes, and m is the number of categories of binary labels. In $Micro\_F_1$, let $r = \frac{1}{m} \sum \frac{tp}{tp + fn}$ and $p = \frac{1}{m} \sum \frac{tp}{tp + fp}$. The Macro_F1 score computes the binary decisions on individual categories and then averages the categories.

Representation Analysis. Figure 5 (a) illustrates the feature spaces of dblp by CDNR bottom-layer random walk (Node2Vec) and Figure 5 (b) illustrates the feature spaces of dblp by CDNR. These two illustrations show almost the same distribution and obtain good mappings in a low dimension compared to PCA (Figure 5 (c)), LLE (Figure 5 (d)) and Laplacian (Figure 5 (e)) based network representations.

Effectiveness of search priority in random walks. In Table 3, DeepWalk and Struc2Vec demonstrate worse performance than LINE, Node2Vec and our CDNR, which can be explained by their inability to reuse samples, a feat that can be easily achieved using the random walk. The outstanding performance of Node2Vec among baseline algorithms indicates that the exploration strategy is much better than the uniform random walks learned by DeepWalk and LINE. The parameter of search bias α adds flexibility in exploring local neighborhoods prior to the global network. The poor performance of DeepWalk and LINE mainly occurs because the network structure is rather sparse, feature noise, and contains limited information. CDNR performs best on the M10 network, as dblp is also a citation network that naturally share similar network patterns with M10. Such patterns are captured by CDNR and transfered to M10. On average, there are smaller variances in the performance of CDNR on the dblp2M10 learning task.

Importance of information from source domain. Table 3 shows that CDNR outperforms the domain-specific baseline algorithms, which use topological information from the source domain to learn the network representation in the target domain. When a top layer is working base on the *CD2L-RandomWalk*, the information in the source network is transferred to the source network by adjusting the weights on the edges of the target network. This procedure achieves better performance and shows the significance of transferring topological information from the external domains.

5.3. Experiment on Multi-label Datasets

5.3.1. Datasets

We select five real-world large-scale networks of different kinds as the experimental datasets, consisting of three online social networks (Blog3, Facebook), two citation networks (arXivCit-HepPh, arXivCit-HepTh) and one biological network (PPI). All of them are for the multi-class multi-label classification problem. In the online social networks, nodes represent users and the users' relationships are denoted as edges. In the citation networks, papers are denoted as nodes and edges describe the citations in this experiment. In the biological network, genes are denoted as nodes and edges represent the relationships between the genes.

- **Blog3 (BlogCatalog3) dataset**[5] is a social blog directory which manages bloggers and their blogs. Both the contact network and selected group membership information is included. The network has 10,312 nodes, 333,983 undirected edges and 39 different labels. Nodes are classified according to the interests of bloggers.

---

[5]http://socialcomputing.asu.edu/datasets/BlogCatalog3

Table 3: CDNR single-label classification results on the target domain of M10.

	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	DeepWalk	0.1758 ±0.0086	0.1833 ±0.0100	0.1897 ±0.0122	0.2049 ±0.0126	0.2051 ±0.0128	0.2216 ±0.0111	0.2236 ±0.0170	0.2420 ±0.0133	0.2431 ±0.0220
	LINE	0.2338 ±0.0102	0.2362 ±0.0170	0.2623 ±0.0110	0.2821 ±0.0141	0.3269 ±0.0150	0.3244 ±0.0087	0.3561 ±0.0193	0.3508 ±0.0184	0.4128 ±0.0486
	Node2Vec	0.3342 ±0.0099	0.4166 ±0.0110	0.4714 ±0.0153	0.5213 ±0.0127	0.5550 ±0.0176	0.5843 ±0.0092	0.6216 ±0.0215	0.6353 ±0.0115	0.6535 ±0.0324
	Struc2Vec	0.1742 ±0.0186	0.1673 ±0.0194	0.1701 ±0.0171	0.1622 ±0.0215	0.1695 ±0.0145	0.1669 ±0.0215	0.1685 ±0.0120	0.1626 ±0.0188	0.1784 ±0.0281
	DeepGL	0.6557 ±0.0187	0.6465 ±0.0176	0.6739 ±0.0161	0.6600 ±0.0197	0.6787 ±0.0164	0.6724 ±0.0139	0.6871 ±0.0184	0.6786 ±0.0160	0.6911 ±0.0403
	CDNR dblp2M10	**0.7507 ±0.0143**	**0.7728 ±0.0114**	**0.8052 ±0.0154**	**0.8245 ±0.0074**	**0.8363 ±0.0051**	**0.8526 ±0.0116**	**0.8587 ±0.0128**	**0.8772 ±0.0173**	**0.8720 ±0.0179**
Macro-F1	DeepWalk	0.2523 ±0.0117	0.2667 ±0.0051	0.2768 ±0.0072	0.2945 ±0.0120	0.2935 ±0.0081	0.3077 ±0.0086	0.3101 ±0.0158	0.3294 ±0.0123	0.3359 ±0.0220
	LINE	0.3160 ±0.0113	0.2984 ±0.0127	0.3421 ±0.0144	0.3596 ±0.0249	0.4070 ±0.0382	0.4275 ±0.0548	0.4498 ±0.0383	0.4277 ±0.0302	0.4773 ±0.0486
	Node2Vec	0.4326 ±0.0147	0.4748 ±0.0156	0.5338 ±0.0153	0.5900 ±0.0153	0.6092 ±0.0290	0.6388 ±0.0314	0.6866 ±0.0202	0.6981 ±0.0572	0.6568 ±0.0261
	Struc2Vec	0.2718 ±0.0126	0.2129 ±0.0092	0.2220 ±0.0143	0.1889 ±0.0147	0.2067 ±0.0058	0.1871 ±0.0150	0.1803 ±0.0098	0.1723 ±0.0271	0.1749 0.0165
	DeepGL	0.6704 ±0.0151	0.6003 ±0.0222	0.6154 ±0.0247	0.5732 ±0.0305	0.5848 ±0.0309	0.5607 ±0.0264	0.5918 ±0.0324	0.5910 ±0.0388	0.5914 0.0407
	CDNR dblp2M10	**0.7558 ±0.0138**	**0.6939 ±0.0168**	**0.7269 ±0.0174**	**0.7174 ±0.0149**	**0.7301 ±0.0256**	**0.7540 ±0.0238**	**0.7679 ±0.0325**	**0.7722 ±0.0609**	**0.7745 ±0.0698**

14

Table 4: Multi-label classification dataset statistics.

Datasets	Network	Num. of Nodes	Num. of Edges	Ave. Degree	Num. of Categories	Labels
Blog3	Social	10,312	333,983	64.776	39	Interests
Facebook	Social	4,039	88,234	43.691	10	Groups
PPI	Biological	3,890	37,845	19.609	50	States
arXivCit-HepPh	Citation	34,546	421,578	24.407	11	Years
arXivCit-HepTh	Citation	27,777	352,807	25.409	11	Years

(a) Blog3   (b) Blog3 RW   (c) Facebook   (d) Facebook RW   (e) PPI

(f) arXivHepPh   (g) arXivHepPh RW   (h) arXivHepTh   (i) arXivHepTh RW   (j) PPI RW
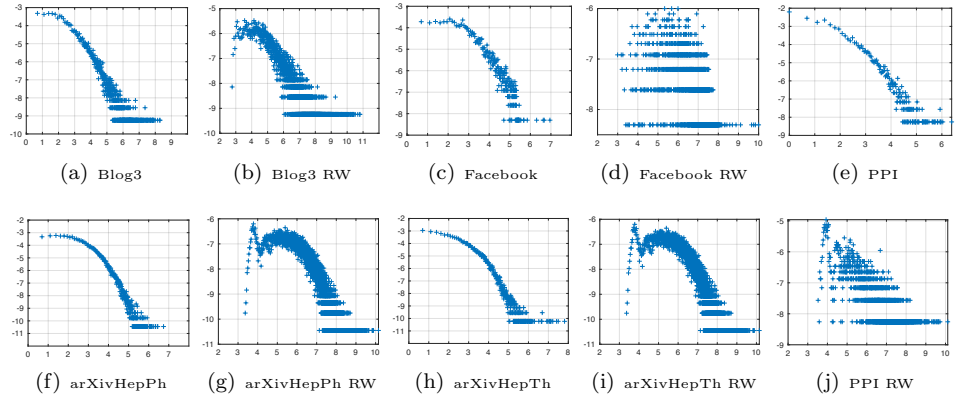
Figure 6: Power-law distributions of multi-label classification datasets and their random walks.

- **Facebook dataset**[6] consists of circles (i.e., friends lists) from Facebook. This dataset contains user profiles as node features, and circles as edge features and ego networks. The network has 4,039 nodes, 88,234 undirected edges and 10 different labels representing groups of users.
- **PPI (Protein-Protein Interactions) dataset**[7] is a subgraph of the PPI network for Homo Sapiens, which obtains labels from hallmark gene sets and represents biological states. The network has 3,890 nodes, 76,584 undirected edges and 50 different labels.
- **arXivCit-HepPh (arXiv High-energy Physics Citation Network) dataset**[8] **and arXivCit-HepTh (arXiv High-energy Physics Theory Citation Network) dataset**[9] are abstracted from the e-print arXiv. arXivCit-HepPh covers all the citations within a dataset of 34,546 papers (regarded as nodes) with 421,578 directed edges. arXivCit-HepTh covers all the citations within a dataset of 27,777 papers (regarded as nodes) with 352,807 directed edges. If a paper v_i cites paper v_j, the graph contains a directed edge from v_i to v_j. The data consist of papers from the period January 1993 to April 2003, categorized by year.

The networks chosen in the experiment follow the power-law distribution [31], as do the random walks on the networks [15], as shown in Figure 6.

5.3.2. Experiment Setup

Table 5: Networks selected as the source domain and target domain for CDNR by distance.

Source Domain	Target Domain
Blog3	PPI
arXivCit-HepTh	PPI
arXivCit-HepPh	PPI
Facebook	PPI
Blog3	Facebook

This experiment summarizes the network statistics in Table 4. Node degree reflects the connection capability of the node. A network is selected as a source domain or a target domain follows $|V^s| > |V^t|$ and $\langle deg^s \rangle > \langle deg^t \rangle$. These selections are shown in Table 5. The experiment setup for the multi-label classification evaluation is as same as the setup in the single-label dataset experiment.

5.3.3. Multi-label Classification

In the multi-label classification setting, every node is assigned one or more labels from a finite set Y. In the training phase of the CDNR node feature representations, we observe a fraction of the nodes and all their labels, and predict the labels for the remaining nodes. The multi-label classification in our experiment inputs the network representations to a *one-against-all* linear SVM classifier [44]. We use the F1 score of Macro-F1 and Micro-F1 to compare performance [43] in Tables 6-9.

---

[6]https://snap.stanford.edu/data/egonets-Facebook.html
[7]https://downloads.thebiogrid.org/BioGRID
[8]http://snap.stanford.edu/data/cit-HepPh.html
[9]http://snap.stanford.edu/data/cit-HepTh.html

Table 6: CDRN multi-label classification results of Micro-F1 on the target domain network of PPI.

Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	0.2849 ±0.0181	0.2854 ±0.0116	0.2845 ±0.0193	0.2803 ±0.0170	0.2725 ±0.0168	0.2736 ±0.0200	0.2629 ±0.0241	0.2778 ±0.0215	0.2621 ±0.0344
LINE	0.2900 ±0.0062	0.2772 ±0.0077	0.2807 ±0.0083	0.2715 ±0.0104	0.2702 ±0.0113	0.2649 ±0.0166	0.2710 ±0.0163	0.2494 ±0.0251	0.2398 ±0.0195
Node2Vec	0.3073 ±0.0171	0.2955 ±0.0104	0.3024 ±0.0139	0.3028 ±0.0120	0.3028 ±0.0102	0.2995 ±0.0186	0.3021 ±0.0288	0.2967 ±0.0197	0.3005 ±0.0283
Struc2Vec	0.2693 ±0.0228	0.2713 ±0.0187	0.2696 ±0.0188	0.2515 ±0.0187	0.2603 ±0.0133	0.2499 ±0.0212	0.2493 ±0.0148	0.2419 ±0.0156	0.2338 ±0.0287
DeepGL	0.3055 ±0.0062	0.3063 ±0.0083	0.3028 ±0.0083	0.2947 ±0.0054	0.2987 ±0.0063	0.2975 ±0.0128	0.2911 ±0.0128	0.2890 ±0.0180	0.2764 ±0.0178
CDNR Blog3 2PPI	0.3386 ±0.0062	0.3390 ±0.0086	0.3423 ±0.0061	0.3420 ±0.0072	0.3404 ±0.0079	0.3414 ±0.0073	0.3350 ±0.0125	0.3371 ±0.0199	0.3312 ±0.0238
CDNR arXivCit -HepPh 2PPI	0.3412 ±0.0041	0.3425 ±0.0075	0.3410 ±0.0052	0.3431 ±0.0057	0.3460 ±0.0069	0.3474 ±0.0112	0.3431 ±0.0103	0.3429 ±0.0107	0.3330 ±0.0192
CDNR arXivCit -HepTh 2PPI	**0.3420 ±0.0036**	0.3426 ±0.0057	0.3434 ±0.0044	**0.3462 ±0.0049**	0.3441 ±0.0042	**0.3553 ±0.0101**	**0.3450 ±0.0106**	**0.3457 ±0.0150**	**0.3521 ±0.0260**
CDNR Facebook 2PPI	0.3415 ±0.0053	**0.3442 ±0.0035**	**0.3454 ±0.0035**	0.3448 ±0.0066	**0.3468 ±0.0065**	0.3410 ±0.0102	0.3447 ±0.0119	0.3443 ±0.0151	0.3444 ±0.0189

17

Table 7: CDRN multi-label classification results of Macro-F1 on the target domain network of PPI.

Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	0.3416 ±0.0140	0.3378 ±0.0138	0.3364 ±0.0208	0.3406 ±0.0171	0.3306 ±0.0159	0.3336 ±0.0241	0.2949 ±0.0288	0.2825 ±0.0185	0.2041 ±0.0386
LINE	0.3058 ±0.0094	0.3003 ±0.0113	0.3008 ±0.0069	0.2940 ±0.0120	0.2868 ±0.0138	0.2826 ±0.0176	0.2733 ±0.0173	0.2462 ±0.0262	0.1822 ±0.0198
Node2Vec	0.3490 ±0.0193	0.3442 ±0.0141	0.3510 ±0.0205	0.3500 ±0.0126	0.3432 ±0.0140	0.3414 ±0.0201	0.3274 ±0.0240	0.3006 ±0.0248	0.2310 ±0.0385
Struc2Vec	0.2892 ±0.0197	0.2926 ±0.0232	0.3019 ±0.0227	0.2784 ±0.0267	0.2851 ±0.0152	0.2626 ±0.0202	0.2589 ±0.0177	0.2399 ±0.0287	0.1712 ±0.0262
DeepGL	0.3213 ±0.0065	0.3290 ±0.0072	0.3235 ±0.0107	0.3155 ±0.0067	0.3136 ±0.0095	0.3086 ±0.0117	0.2970 ±0.0147	0.2783 ±0.0220	0.2115 ±0.0132
CDNR Blog3 2PPI	0.3519 ±0.0108	0.3551 ±0.0105	0.3539 ±0.0073	0.3514 ±0.0060	0.3469 ±0.0097	0.3431 ±0.0060	0.3282 ±0.0154	0.3063 ±0.0223	0.2389 ±0.0276
CDNR arXivCit -HepPh 2PPI	0.3532 ±0.0106	0.3582 ±0.0099	0.3536 ±0.0085	0.3509 ±0.0057	0.3531 ±0.0098	0.3456 ±0.0085	0.3360 ±0.0128	0.3130 ±0.0144	0.2512 ±0.0270
CDNR arXivCit -HepTh 2PPI	0.3570 ±0.0079	0.3575 ±0.0091	0.3568 ±0.0044	**0.3565 ±0.0091**	0.3523 ±0.0090	**0.3556 ±0.0137**	0.3368 ±0.0150	**0.3234 ±0.0249**	**0.2682 ±0.0330**
CDNR Facebook 2PPI	**0.3576 ±0.0086**	**0.3595 ±0.0065**	**0.3574 ±0.0052**	0.3553 ±0.0068	**0.3573 ±0.0080**	0.3432 ±0.0079	**0.3423 ±0.0145**	0.3164 ±0.0164	0.2578 ±0.0217

Table 8: CDNR multi-label classification results of Micro-F1 on the target domain network of Facebook.

Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	0.8078 ±0.0449	0.8727 ±0.0177	0.8933 ±0.0062	0.9050 ±0.0059	0.9153 ±0.0060	0.9198 ±0.0061	0.9307 ±0.0039	0.9301 ±0.0103	0.9334 ±0.0175
LINE	0.4627 ±0.0026	0.4654 ±0.0104	0.4719 ±0.0026	0.4739 ±0.0035	0.4765 ±0.0035	0.4761 ±0.0033	0.4760 ±0.0067	0.4787 ±0.0066	0.4755 ±0.0075
Node2Vec	0.9352 ±0.0072	0.9401 ±0.0032	0.9398 ±0.0051	0.9419 ±0.0047	0.9442 ±0.0057	0.9454 ±0.0063	0.9468 ±0.0092	0.9466 ±0.0079	0.9502 ±0.0098
Struc2Vec	0.4152 ±0.0237	0.4521 ±0.0144	0.4716 ±0.0061	0.4994 ±0.0059	0.5161 ±0.0078	0.5381 ±0.0096	0.5461 ±0.0115	0.5639 ±0.0241	0.5530 ±0.0175
DeepGL	0.9535 ±0.0161	0.9483 ±0.0114	0.9531 ±0.0059	0.9515 ±0.0099	0.9552 ±0.0087	0.9546 ±0.0087	0.9555 ±0.0039	0.9612 ±0.0089	**0.9640 ±0.0072**
CDNR Blog32Facebook	**0.9584 ±0.0038**	**0.9550 ±0.0036**	**0.9561 ±0.0049**	**0.9565 ±0.0044**	**0.9568 ±0.0032**	**0.9617 ±0.0050**	**0.9606 ±0.0042**	**0.9627 ±0.0067**	0.9623 ±0.0085

Table 9: CDNR multi-label classification results of Macro-F1 on the target domain network of Facebook.

Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	0.7655 ±0.0185	0.7915 ±0.0242	0.7858 ±0.0331	0.8052 ±0.0308	0.7902 ±0.0306	0.8138 ±0.0327	0.8213 ±0.0504	0.7678 ±0.0317	0.7822 ±0.0378
LINE	0.5063 ±0.0053	0.5040 ±0.0189	0.5083 ±0.0093	0.5129 ±0.0061	0.5091 ±0.0092	0.5040 ±0.0077	0.5020 ±0.0137	0.4981 ±0.0117	0.4961 ±0.0109
Node2Vec	0.8310 ±0.0256	0.8331 ±0.0226	0.8206 ±0.0262	**0.8373 ±0.0359**	0.8343 ±0.0354	0.8214 ±0.0479	0.8192 ±0.0487	0.8018 ±0.0277	0.8104 ±0.0498
Struc2Vec	0.3701 ±0.0156	0.3937 ±0.0157	0.3926 ±0.0174	0.4160 ±0.0155	0.4377 ±0.0235	0.4525 ±0.0131	0.4532 ±0.0144	0.4755 ±0.0260	0.4583 ±0.0347
DeepGL	**0.8810 ±0.0330**	0.8660 ±0.0328	0.8724 ±0.0381	0.8748 ±0.0343	0.8794 ±0.0395	0.8856 ±0.0313	0.8578 ±0.0350	**0.8732 ±0.0514**	**0.8757 ±0.0537**
CDNR Blog32Facebook	0.8749 ±0.0301	**0.8831 ±0.0135**	**0.8866 ±0.0304**	0.8766 ±0.0360	**0.8890 ±0.0291**	**0.8876 ±0.0294**	**0.8910 ±0.0334**	0.8443 ±0.0468	0.8415 ±0.0482

Experimental results from the algorithmic perspective. A general observation drawn from the results is that the learned feature representations from other networks improve or maintain performance compared to the domain-specific network representation baseline algorithms. CDNR outperforms DeepWalk, LINE, Node2Vec, Struc2Vec and DeepGL in all datasets with a gain of 19.29%, 49.57%, 15.66%, 58.83% and 10.06%, respectively. CDNR outperforms DeepWalk, LINE, Node2Vec and Struc2Vec on the PPI dataset and the Facebook dataset in 100% of the experiment, and outperforms DeepGL on the PPI dataset in 100% and the Facebook dataset in 88.89% of the experiment. The losses of CDNR to DeepGL on the training percentages of {80%,90%} might caused by classifier and training sample selection and NN-based DeepGL shows robustness than other algorithms.

Experimental results from the dataset perspective. The general results on the PPI dataset (Tables 6 and 7) reflect the difficulty of cross-domain learning. Considering the domain similarities, a cross-domain adaption from either the social networks or the citation networks to the biological network as shown in our experiment would not be recommended in transfer learning. However, CDNR is capable of capturing useful structural information from network topologies and removing noise from the source domain networks in an unsupervised feature-learning environment, so CDNR on PPI still shows a slight improvement and almost retains its representation performances. Therefore, cross-domain network knowledge transfer learning works in unsupervised network representations. CDNR is less influenced by domain selections when the transferable knowledge is mainly contributed by network topologies.

Examining the results in detail shows that the source domain networks of arXivCit-HepTh and Facebook provide a larger volume of information to the PPI target domain network than other pairs of CDNR experiments, which promote knowledge transfer across domains. The citation networks of arXivCit-HepPh and arXivCit-HepTh transfer 11 categories of Years to PPI (biological network, 50 categories of States, network average degree of 19.609) with a network average degree of 24.407 and 25.409 respectively. The social networks of Blog3 and Facebook transfers 39 categories of Interests with the network average degree of 64.776 and 43.691 respectively. The show that unsupervised CDNR works especially well in dense networks, however, domains share rare natural similarities still can't guarantee a good knowledge transfer (Blog32PPI: Interests to States).

In addition, the general results on the Facebook dataset (Tables 8 and 9) show promising improvements by CDNR compared to other baseline algorithms. Unsupervised representations of CDNR allow learning from small categories to large categories, and in a heterogeneous label space. CDNR uses its *CD2L-RandomWalk* learning algorithm to capture the useful topologies in a large-scale information network.

5.4. Statistical Significance

To demonstrate that CDNR is indeed statistically superior to the baseline algorithms, we summarize our results for all classification evaluation tasks in Table 10 by pairwise t-test at a confidence level of $\alpha = 0.05$. The statistical significance is validated on every paired CDNR and baseline algorithm. On the single-label datasets, for example CDNR from the dblp dataset to the M10 dataset (CDNRdblp2M10) is compared with DeepWalk, LINE, Node2Vec, Struc2Vec and DeepGL by pairwise t-test.

9.02E-18 in line 11 column 8 of Table 10 is a mean significance value averaged from nine significance values on {10%, $\cdots$, 90%} training percentages. Each of these significance values is t-tested between CDNRBlog32Facebook and LINE. Since the CDNR multi-label dataset experiment is conducted across five datasets, the statistical significance is validated for each scenario; for example

Table 10: Pairwise t-test results of CDNR versus baseline algorithms.

	CDNRdblp2M10					CDNRBlog32PPI				
	DeepWalk	LINE	Node2Vec	Struc2Vec	DeepGL	DeepWalk	LINE	Node2Vec	Struc2Vec	DeepGL
Micro-F1	3.78E-13	7.29E-12	7.39E-07	8.13E-11	5.47E-07	3.66E-09	2.73E-07	1.03E-08	1.88E-08	8.98E-08
Macro-F1	1.04E-11	1.15E-08	4.31E-04	6.22E-10	5.07E-06	2.92E-04	3.17E-10	5.51E-03	7.26E-09	2.93E-09
	CDNRarXivCit-HepPh2PPI					CDNRarXivCitHepTh2PPI				
	DeepWalk	LINE	Node2Vec	Struc2Vec	DeepGL	DeepWalk	LINE	Node2Vec	Struc2Vec	DeepGL
Micro-F1	1.38E-08	2.77E-07	1.75E-08	2.65E-08	9.47E-08	1.12E-07	1.53E-06	2.51E-08	1.92E-07	2.02E-06
Macro-F1	7.00E-04	6.25E-09	3.45E-03	1.79E-08	4.82E-09	7.15E-04	1.04E-07	3.18E-03	1.74E-07	4.45E-07
	CDNRFacebook2PPI					CDNRBlog32Facebook				
	DeepWalk	LINE	Node2Vec	Struc2Vec	DeepGL	DeepWalk	LINE	Node2Vec	Struc2Vec	DeepGL
Micro-F1	2.20E-08	5.29E-07	1.18E-09	4.18E-08	4.61E-07	2.04E-03	9.02E-18	5.54E-07	2.95E-09	4.57E-03
Macro-F1	6.35E-04	7.98E-09	1.27E-03	1.25E-08	2.60E-08	7.94E-07	1.35E-12	2.65E-06	2.87E-09	9.85E-01

CDNRBlog32PPI is CDNR from Blog3 to PPI, and 3.66E-09 in line 3 column 7 is averaged from the nine significance values by pairwise t-testing CDNRBlog32PPI and DeepWalk.

In Table 10, each value less than $\alpha = 0.05$ indicates that the difference is statistically significant. The results in Table 10 confirm that CDNR statistically outperforms DeepWalk, LINE, Node2Vec, Struc2Vec and DeepGL in all cases expect CDNRBlog32Facebook on Macro-F1 in DeepGL which caused by the inferior results on {80%,90%} training percentages.

5.5. Parameter Sensitivity

In this experiment, CDNR sets the representational dimension d in 128 following the setting in baselines of DeepWalk, LINE, Node2Vec and Struc2Vec. However, in DeepGL, representational dimensions are determined by deep neural network training. The benefit of DeepGL, according to [35], is that it is able to determine the appropriate number of features automatically, as opposed to setting it to some fixed value. Unlike other techniques, DeepGL derives new feature layers as long as new and informative features are found. There is at least one new feature from the current layer remaining after pruning. In order to address the emphasis that dimensional parameter d puts on network representations, we set up a parameter sensitivity testing in this part.

CDNR on M10, PPI and Facebook setting in different feature learning dimensions, d={2, 3, 4,8,16,32,64,128,256}, are evaluated in Figure 7. On average, representations trend to converge when $d > 16$ and get a convergence on $d = 128$.

In Tables 3 and 6-9, the representation performance of CDNR in $d = 128$ has been proved significant compared with DeepGL in d_{DeepGL} that varies in learning. In the format of [Minimum d_{DeepGL}, Average d_{DeepGL}, Maximum d_{DeepGL}], the DeepGL feature dimensions on M10, PPI and Facebook are $[57, 66, 80]$, $[51, 61, 70]$, $[56, 64, 82]$, respectively. In this part, the representation dimensional sensitivities in CDNR are compared with DeepGL-base. To set up, DeepGL is tested on the 3-dimensional base features; correspondingly, CDNR is evaluated on the representations in $d = 3$; and the performances in $d = 3$ are compared with the CDNR performance in $d = 128$. The sensitivity results in Figure 8 show that CDNR on $d = 128$ achieves the best performance than CDNR $d = 3$ and DeepGL-base; DeepGL significantly relies on operators which represent large variance in the red bold line; and DeepGL-base can barely reach a convergence.

In conclusion, for a common real-world network with sparse structure, CDNR outperforms DeepGL by outputting a dense feature matrix with relatively smaller dimensional size. When

(a) dblp2M10   (b) Blog32PPI   (c) arXivCit-HepPh2PPI

(d) arXivCit-HepTh2PPI   (e) Facebook2PPI   (f) Blog32Facebook
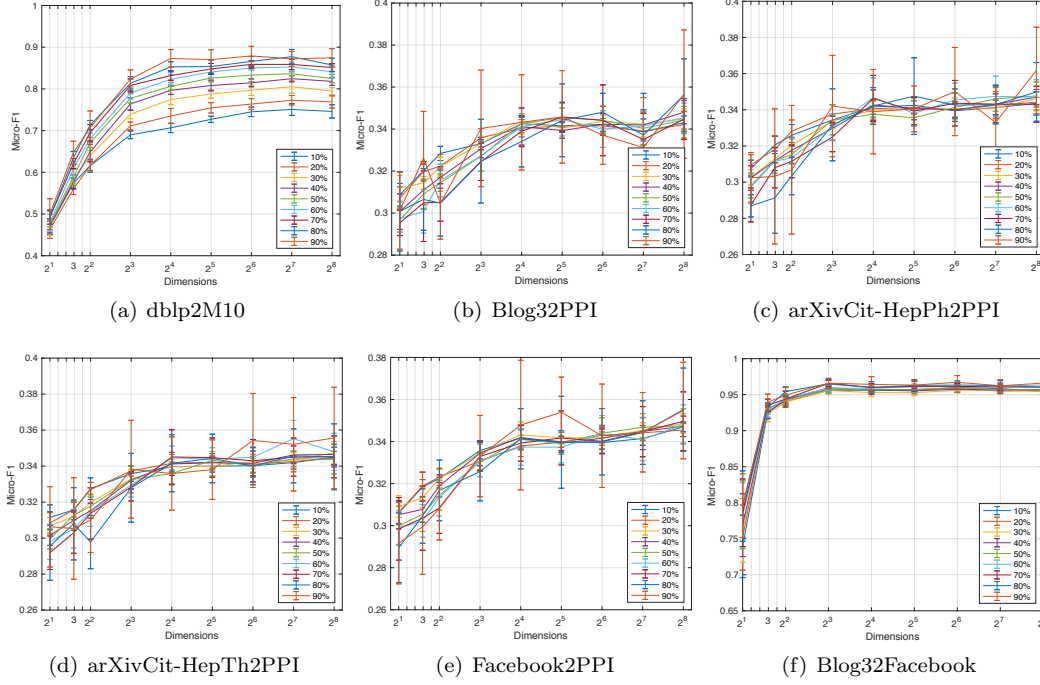
Figure 7: CDNR dimensional sensitivity on $d = \{2, 3, 4, 8, 16, 32, 64, 128, 256\}$. Lines in nine colors denote dimensional sensitivity on training sample percentages {10%,20%,30%,40%,50%,60%,70%,80%,90%}. The error bars in each line on d points reflect the variance in 10-time testing.
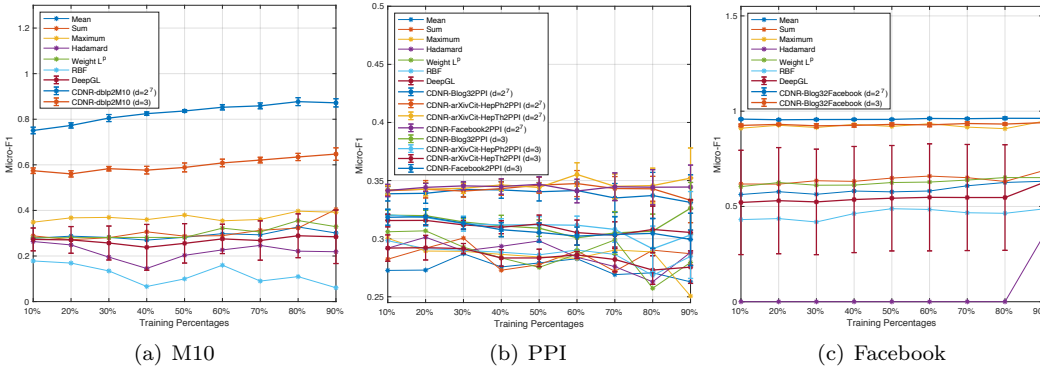

(a) M10   (b) PPI   (c) Facebook

Figure 8: Dimensional sensitivity analysis between CDNR and DeepGL-base. The thin lines denoted by operator = {Mean, Sum, Maximum, Hadamard, Weighted L^p, RBF} are tested on the 3-dimensional ($d = 3$) base features generated by DeepGL; the red bold line is drawn from the average DeepGL-base experimental results; the orange bold line represents the CDNR experimental results when the dimensional parameter d is set in 3; and the blue bold line represents the CDNR experimental results in $d = 128$. Experiments on PPI dataset get more CDNR results by transferring knowledge from different source domains.

evaluating against DeepGL, it explains that DeepGL outputs a sparse feature matrix in contrast to other approaches even if it is larger. However, fixing the number of features is arbitrary in the case of DeepGL. Therefore, CDNR shows less dimensional sensitivity in cross-domain feature learning.

6. Conclusions

This work proposed a solution for a new random walk-based CDNR problem. Compared to previous network representation approaches, CDNR enables effective knowledge transfer from the external domains. Two key components flexibly tackle the challenges. The algorithm is general for universal real-world networks and is computational efficient for knowledge transferring from large-scale networks with runtime that is linear in the number of edges of the target network. CDNR has all the desired properties: flexible with any kind of networks for variety of domains and learning scenarios, effective for sampling network structures from source domain, efficient for learning from gained knowledge, and accurate with a mean improvement in F1 score of 30.68%. Future works of similarity learning across domains and between networks based on network patterns will be studied to address the limitation in CDNR.

References

References

[1] J. Xuan, X. Luo, G. Zhang, J. Lu, Z. Xu, Uncertainty analysis for the keyword system of web events, IEEE Transactions on Systems, Man, and Cybernetics: Systems 46 (6) (2016) 829–842.

[2] H. Wang, J. Wu, P. Zhang, Y. Chen, Learning shapelet patterns from network-based time series data, IEEE Transactions on Industrial Informatics.

[3] R. Chaker, Z. Al Aghbari, I. N. Junejo, Social network model for crowd anomaly detection and localization, Pattern Recognition 61 (2017) 266–281.

[4] K. Choi, K.-A. Toh, H. Byun, Incremental face recognition for large-scale social network services, Pattern Recognition 45 (8) (2012) 2868–2883.

[5] P. Jancura, E. Marchiori, Dividing protein interaction networks for modular network comparative analysis, Pattern Recognition Letters 31 (14) (2010) 2083–2096.

[6] J. Lu, J. Xuan, G. Zhang, X. Luo, Structural property-aware multilayer network embedding for latent factor analysis, Pattern Recognition 76 (2018) 228–241.

[7] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, ACM, 2015, pp. 1067–1077.

[8] L. Tang, H. Liu, Relational learning via latent social dimensions, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 817–826.

[9] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (8) (2013) 1798–1828.

[10] J. D. Noh, H. Rieger, Random walks on complex networks, Physical Review Letters 92 (11) (2004) 118701.

[11] D. Lai, H. Lu, C. Nardini, Enhanced modularity-based community detection by random walk network preprocessing, Physical Review E 81 (6) (2010) 066118.

[12] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowledge and Information Systems 42 (1) (2015) 181–213.

[13] J. Leskovec, J. J. Mcauley, Learning to discover social circles in ego networks, in: Proceedings of the 26th Advances in Neural Information Processing Systems, 2012, pp. 539–547.

[14] W. Liu, L. Lü, Link prediction based on local random walk, Europhysics Letters 89 (5) (2010) 58007.

[15] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 701–710.

[16] S. Xue, J. Lu, G. Zhang, L. Xiong, A framework of transferring structures across large-scale information networks, in: Proceedings of 2018 International Joint Conference on Neural Networks, 2018, pp. 1137–1142.

[17] A. Bandyopadhyay, D. Gamarnik, Counting without sampling: Asymptotics of the log-partition function for certain statistical physics models, Random Structures & Algorithms 33 (4) (2008) 452–479.

[18] M. Kurant, A. Markopoulou, P. Thiran, Towards unbiased BFS sampling, IEEE Journal on Selected Areas in Communications 29 (9) (2011) 1799–1809.

[19] L. H. Lelis, L. Otten, R. Dechter, Predicting the size of depth-first branch and bound search trees., in: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, 2013, pp. 594–600.

[20] P. Paalanen, J.-K. Kamarainen, J. Ilonen, H. Kälviäinen, Feature representation and discrimination based on Gaussian mixture model probability densitiespractices and algorithms, Pattern Recognition 39 (7) (2006) 1346–1358.

[21] P. Zhu, W. Zuo, L. Zhang, Q. Hu, S. C. Shiu, Unsupervised feature selection by regularized self-representation, Pattern Recognition 48 (2) (2015) 438–446.

[22] S. Ding, L. Lin, G. Wang, H. Chao, Deep feature learning with relative distance comparison for person re-identification, Pattern Recognition 48 (10) (2015) 2993–3003.

[23] C. Yang, Z. Liu, D. Zhao, M. Sun, E. Y. Chang, Network representation learning with rich text information, in: Proceedings of the 24th International Joint Conference on Artificial Intelligence, 2015, pp. 2111–2117.

[24] S. Pan, J. Wu, X. Zhu, C. Zhang, Y. Wang, Tri-party deep network representation, in: Proceedings of the 25th International Joint Conference on Artificial Intelligence, 2016, pp. 1895–1901.

[25] C. Tu, W. Zhang, Z. Liu, M. Sun, Max-margin deepwalk: Discriminative learning of network representation., in: Proceedings of the 25th International Joint Conference on Artificial Intelligence, 2016, pp. 3889–3895.

[26] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 855–864.

[27] L. F. Ribeiro, P. H. Saverese, D. R. Figueiredo, Struc2vec: Learning node representations from structural identity, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 385–394.

[28] M. E. Newman, Power laws, Pareto distributions and Zipf's law, Contemporary Physics 46 (5) (2005) 323–351.

[29] A.-L. Barabási, Scale-free networks: A decade and beyond, Science 325 (5939) (2009) 412–413.

[30] A.-L. Barabási, M. Pósfai, Network science, Cambridge University Press, 2016.

[31] L. A. Adamic, B. A. Huberman, Power-law distribution of the world wide web, Science 287 (5461) (2000) 2115–2115.

[32] L. A. Adamic, R. M. Lukose, A. R. Puniyani, B. A. Huberman, Search in power-law networks, Physical Review E 64 (4) (2001) 046135.

[33] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Proceedings of the 31st Advances in Neural Information Processing Systems, 2017, pp. 1025–1035.

[34] J. Lee, H. Kim, J. Lee, S. Yoon, Transfer learning for deep learning on graph-structured data, in: Proceedings of the 31st Association for the Advancement of Artificial Intelligence, 2017, pp. 2154–2160.

[35] R. A. Rossi, R. Zhou, N. Ahmed, Deep inductive graph representation learning, IEEE Transactions on Knowledge and Data Engineering.

[36] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, G. Zhang, Transfer learning using computational intelligence: A survey, Knowledge-Based Systems 80 (2015) 14–23.

[37] S. J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on Knowledge and Data Engineering 22 (10) (2010) 1345–1359.

[38] H. Wang, P. Zhang, X. Zhu, I. W.-H. Tsang, L. Chen, C. Zhang, X. Wu, Incremental subgraph feature selection for graph classification, IEEE Transactions on Knowledge and Data Engineering 29 (1) (2017) 128–142.

[39] T. Guo, X. Zhu, Super-graph classification, in: Proceedings of the 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2014, pp. 323–336.

[40] J. A. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Processing Letters 9 (3) (1999) 293–300.

[41] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, A. J. Smola, Distributed large-scale natural graph factorization, in: Proceedings of the 22nd International Conference on World Wide Web, ACM, 2013, pp. 37–48.

[42] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes, Pattern Recognition 44 (8) (2011) 1761–1776.

[43] Y. Yang, X. Liu, A re-examination of text categorization methods, in: Proceedings of the 22nd Annual International ACM SIGIR conference on Research and Development in Information Retrieval, ACM, 1999, pp. 42–49.

[44] C.-W. Hsu, C.-J. Lin, A comparison of methods for multiclass support vector machines, IEEE Transactions on Neural Networks 13 (2) (2002) 415–425.