This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Data Representation and Learning with Graph Diffusion-Embedding Networks

Bo Jiang, Doudou Lin, Jin Tang*, Bin Luo
School of Computer Science and Technology, Anhui University, Hefei, 230601, China
jiangbo@ahu.edu.cn, ahu_lindd@163.com, ahhftang@gmail.com, luobin@ahu.edu.cn

Abstract

Recently, graph convolutional neural networks have been widely studied for graph-structured data representation and learning. In this paper, we present Graph Diffusion-Embedding networks (GDENs), a new model for graph-structured data representation and learning. GDENs are motivated by our development of graph based feature diffusion. GDENs integrate both feature diffusion and graph node (low-dimensional) embedding simultaneously into a unified network by employing a novel ***diffusion-embedding*** *architecture. GDENs have two main advantages. First, the equilibrium representation of the diffusion-embedding operation in GDENs can be obtained via a simple closed-form solution, which thus guarantees the compactivity and efficiency of GDENs. Second, the proposed GDENs can be naturally extended to address the data with multiple graph structures. Experiments on various semi-supervised learning tasks on several benchmark datasets demonstrate that the proposed GDENs significantly outperform traditional graph convolutional networks.*

1. Introduction

Compact feature representation of data is a fundamental problem in computer vision and machine learning area. Convolutional Neural Networks (CNNs) have been widely applied for data representation and learning in many computer vision applications, in which the underlying data generally have a grid-like structure. However, in some real problems, data are not coming with grid-like structure but instead have some irregular structures. Such data are usually represented in the form of graphs. Traditional CNNs generally fail to directly address graph-structured data and the convolution operation on graph data is generally not as well-defined as on grid structure data. To solve this problem, early works aim to use Recursive Neural Networks (RNNs) [7, 29] to deal with directed acyclic graph data. As an extension of RNNs, Graph Neural Networks (GNNs) [9, 27] have also been introduced to address the general cyclic graph data.

*Corresponding author

Recently, there is an increasing attention in generalizing neural networks to deal with arbitrary graph-structured data [22, 10, 30, 30]. For example, Bruna et al. [3] propose a CNN-like neural architecture on graphs in the Fourier domain by employing eigen-decomposition of graph Laplacian. To reduce high computational complexity of graph Laplacian eigen-decomposition, Defferrard et al. [4] propose to approximate the filters by using recurrent Chebyshev polynomials. Kipf and Welling [15] further propose a simplified Graph Convolutional Network (GCN) based on the first-order approximation of spectral filters. Similar work has also been proposed in [6]. Atwood and Towsley [1] propose Diffusion-Convolutional Neural Networks (DCNNs) by employing a random walk diffusion process to include contextual information for data feature representation. Monti et al. [22] present mixture model CNNs (MoNet) which provides a kind of unified generalization of CNN architecture on graphs. Veličković et al. [30] present Graph Attention Networks (GAT) by designing an attention layer that aggregates the feature information of the neighboring nodes for data representation.

In this paper, we propose Graph Diffusion-Embedding Networks (GDENs) for graph data representation and learning. GDENs are motivated by our development on graph based feature diffusion to explore contextual information for graph node representation. Our GDENs follow the general network structure of recent GCNs [15], but compute the hidden representation of each node by employing a novel 'diffusion-embedding' architecture. The main benefits of the proposed diffusion-embedding architecture are two aspects. First, the equilibrium representation of the proposed diffusion-embedding operation can be obtained via a simple closed-form solution, which guarantees the compactivity and efficiency of GDENs. Second, the proposed diffusion-embedding architecture can be naturally extended to address the data with multiple graph structures. Overall, the main contributions are summarized as follows:

- We propose Graph Diffusion-Embedding Networks (GDENs) which integrate graph feature diffusion and embedding together for graph data representation and semi-supervised learning.

- We propose a general mechanism of graph based feature diffusion which can provide a compact contextual representation for graph node by jointly employing both unary feature information of each node and contextual feature information from its neighboring nodes simultaneously.
- Based on GDENs, we provide a novel graph neural network (called as M-GDEN) to address multiple graph data representation and learning.

As an application, we apply GDENs on various semi-supervised classification tasks on graph data. Experimental results on several widely used benchmarks demonstrate that GDENs significantly outperform the state-of-the-art graph neural network models on semi-supervised learning tasks.

2. Brief Review of GCN

As an extension of CNNs from regular grid to irregular graph, Graph Convolutional Networks (GCNs) [4, 15] have been widely studied for graph data representation and learning in recent years. Similar to the classic CNNs, GCNs contain several convolutional hidden layers that take a feature map matrix $\mathbf{H}^{(k)} \in \mathbb{R}^{n\times d_k}$ as the input and output a feature map $\mathbf{H}^{(k+1)} \in \mathbb{R}^{n\times d_{k+1}}$ by using a graph convolution operator. In general, we set $d_{k+1} \leq d_k$, and thus the convolution operation also provides a kind of low-dimensional embedding for nodes on graph.

Given an input feature matrix $\mathbf{H}^{(0)} \in \mathbb{R}^{n\times d_0}$ and graph $\mathbf{A} \in \mathbb{R}^{n\times n}$, GCN [15] conducts the following layer-wise propagation as,

$$\mathbf{H}^{(k+1)} = \sigma\big((\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{H}^{(k)}\mathbf{W}^{(k)}\big) \tag{1}$$

where $k = 0, 1, \cdots K-1$ and $\mathbf{I}$ is an identity matrix. $\mathbf{D} = \text{diag}(\mathbf{d}_1, \mathbf{d}_2 \cdots \mathbf{d}_n)$ is a diagonal matrix with $\mathbf{d}_i = \sum_{j=1}^{n} \mathbf{A}_{ij}$. $\mathbf{W}^{(k)} \in \mathbb{R}^{d_k \times d_{k+1}}$ is a layer-specific trainable weight matrix. $\sigma(\cdot)$ denotes an activation function, such as $\text{ReLU}(\cdot) = \max(0, \cdot)$.

Remark. To alleviate the problem of numerical instabilities and vanishing gradients, it is suggested to use the following re-normalization trick in above propagation (Eq.(1)) as [15],

$$\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} \rightarrow \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}} \tag{2}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\tilde{\mathbf{D}}_{ii} = \sum_{j=1}^{n} \tilde{\mathbf{A}}_{ij}$.

The last layer of GCN outputs the final representation $\mathbf{H}^{(K)}$ of graph nodes, which can be widely utilized for graph data analysis, such as graph classification [4], graph link prediction [16] or graph node (semi-supervised) classification [15], etc. In this paper, we focus on semi-supervised classification. For this task, a softmax activation function is further conducted on each row of the final output feature map matrix $\mathbf{H}^{(K)}$. Let $\mathbf{Z} = \text{softmax}(\mathbf{H}^{(K)}) \in \mathbb{R}^{n\times c}$ be the final output, where c denotes the number of class. Then $\mathbf{Z}$ provides a kind of label prediction for graph nodes. The weight parameters of GCN network $\mathcal{W} = \{\mathbf{W}^0, \mathbf{W}^1, \cdots \mathbf{W}^{(K)}\}$ are trained by minimizing the cross-entropy loss over all the labeled nodes L, i.e.,

$$\mathcal{L}_{\text{Semi-GCN}} = -\sum\nolimits_{i\in L}\sum\nolimits_{j=1}^{c} \mathbf{Y}_{ij}\ln\mathbf{Z}_{ij} \tag{3}$$

where L indicates the set of labeled nodes and each row $\mathbf{Y}_{i\cdot}, i \in L$ of $\mathbf{Y}$ denotes the corresponding label indication vector for the i-th labeled node [15].

3. Graph Diffusion-Embedding Networks

3.1. Motivation

In this section, we propose Graph Diffusion-Embedding Networks (GDENs). Our GDENs are motivated by our observation on layer-wise propagation rule of GCN [15]. Intuitively, the propagation rule Eq.(1) in GCN can be decomposed into two operations, i.e.,

$$\mathbf{F}^{(k)} = (\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{H}^{(k)} \tag{4}$$

$$\mathbf{H}^{(k+1)} = \sigma\big(\mathbf{F}^{(k)}\mathbf{W}^{(k)}\big) \tag{5}$$

where Eq.(4) provides a kind of *feature diffusion* on graph $\mathbf{A}$ and Eq.(5) presents a non-linear transformation (*embedding*) for feature $\mathbf{F}^{(k)}$ via projection matrix $\mathbf{W}^{(k)}$ and non-linear activation function $\sigma(\cdot)$.

Here, we focus on diffusion operation Eq.(4), which is further analyzed as follows. For simplicity, we rewrite update Eq.(4) as

$$\mathbf{F} = (\mathbf{I} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{H} \tag{6}$$

Let $\mathbf{F} = (\mathbf{f}_1 \cdots \mathbf{f}_n)$ and $\mathbf{H} = (\mathbf{h}_1 \cdots \mathbf{h}_n)$, then we can note that the representation $\mathbf{f}_i$ of each node is updated as

$$\mathbf{f}_i = \sum\nolimits_{j,j\neq i} \hat{\mathbf{A}}_{ij}\mathbf{h}_j + \mathbf{h}_i \tag{7}$$

where $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$. From Eq.(7), we can note that GCN indeed employs an one-step feature diffusion on normalized graph $\hat{\mathbf{A}}$ (biased by feature itself) to obtain contextual feature representation in layer-wise propagation. *Obviously, this one-step diffusion does not return the equilibrium representation of feature diffusion which thus may lead to weak contextual feature representation.* This motivated us to propose a general diffusion-embedding network. In the following, we first propose our general feature diffusion models. Then, we present the details of our GDEN architecture.

3.2. Graph based feature diffusion

Given a graph $\mathbf{A} \in \mathbb{R}^{n\times n}$ and feature descriptors $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2 \cdots \mathbf{h}_n) \in \mathbb{R}^{n\times d}$ of graph nodes, the aim of our graph based feature diffusion is to learn a kind of feature representation $\mathbf{F} = \mathcal{F}_d(\mathbf{A}, \mathbf{H}) = (\mathbf{f}_1, \mathbf{f}_2 \cdots \mathbf{f}_n) \in \mathbb{R}^{n\times d}$ for graph nodes by incorporating the contextual information of their neighboring nodes. In the following, we present three kinds of graph based feature diffusion models. Note that, similar diffusion models have been commonly used in ranking and label propagation tasks [34, 35, 5, 20]. Differently, here we propose to explore them for *feature diffusion and representation* problem.

(1) Graph random walk feature diffusion

One intuitive way to formulate the feature diffusion $\mathcal{F}_d(\mathbf{A}, \mathbf{H})$ is using random walk with restart (RWR) model [35, 20, 12, 13] and obtain the equilibrium representation of graph nodes based on the final equilibrium state of random walk. First, we define a transition probability matrix as

$$\mathbf{P} = \mathbf{A}\mathbf{D}^{-1} \tag{8}$$

where $\mathbf{D} = \mathrm{diag}(\mathbf{d}_1 \cdots \mathbf{d}_n)$ with $\mathbf{d}_i = \sum_j \mathbf{A}_{ij}$. Then, starting from initial $\mathbf{F}^{(0)} = \mathbf{H} = (\mathbf{h}_1 \cdots \mathbf{h}_n)$, we explore RWR model to conduct diffusion as

$$\mathbf{f}_i^{(t+1)} = \lambda \sum\nolimits_{j,j\neq i} \mathbf{P}_{ij}\mathbf{f}_j^{(t)} + (1-\lambda)\mathbf{h}_i \tag{9}$$

where $t = 0, 1, \cdots$ and $\lambda \in [0, 1]$. $1-\lambda$ is the jump probability. Each vector $\mathbf{f}_j^{(t)}$ denotes the j-th row of $\mathbf{F}^{(t)}$. The restart term is used to preserve the unary feature information of each node during diffusion. It is known that, the above iteration (Eq.(9)) will converge to the equilibrium state as,

$$\mathbf{F} = (1-\lambda)(\mathbf{I} - \lambda\mathbf{P})^{-1}\mathbf{H} \tag{10}$$

We use this converged $\mathbf{F}$ as the final equilibrium representation of graph nodes.

Remark. It is noted that, GCN update rule Eq.(4 or 7) is similar to the one-step iteration ($t = 0$) of Eq.(9) with $\alpha = 0.5$ (on row normalize graph $\mathbf{P}$). Therefore, from diffusion aspect, the proposed diffusion Eq.(9 or 10) provides an equilibrium and flexible contextual feature representation by exploring graph structure $\mathbf{A}$ and the input feature $\mathbf{H}$. We will incorporate this feature diffusion in our GDEN layer-wise propagation, as shown in §3.3 in detail.

(2) Graph Laplacian feature diffusion

In additional to the above RWR model, we can also formulate feature diffusion $\mathcal{F}_d(\mathbf{A}, \mathbf{H})$ by employing Laplacian regularization models [34]. In particular, we propose to compute the optimal diffused representation $\mathbf{F}$ by minimizing the following Laplacian regularization problem,

$$\min_{\mathbf{F}} \ \frac{1}{2}\sum_{i,j=1}^{n} \mathbf{A}_{ij}\|\mathbf{f}_i - \mathbf{f}_j\|_2^2 + \lambda\sum_{i=1}^{n}\|\mathbf{f}_i - \mathbf{h}_i\|_2^2 \tag{11}$$

The first smoothness term conducts feature diffusion (or propagation) on graph $\mathbf{A}$ while the second fitting term encourages to preserve the original feature information $\mathbf{h}_i$ of each node in diffusion process. It is known that, Eq.(11) has a closed-form solution which is given by

$$\mathbf{F} = \lambda(\mathbf{D} - \mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{H} \tag{12}$$

Similarly, we can also compute the optimal diffused representation $\mathbf{F}$ by minimizing the following normalized Laplacian regularization problem [35, 20],

$$\min_{\mathbf{F}} \ \frac{1}{2}\sum_{i,j=1}^{n} \mathbf{A}_{ij}\|\frac{\mathbf{f}_i}{\sqrt{\mathbf{d}_i}} - \frac{\mathbf{f}_j}{\sqrt{\mathbf{d}_j}}\|_2^2 + \lambda\sum_{i=1}^{n}\|\mathbf{f}_i - \mathbf{h}_i\|_2^2 \tag{13}$$

The optimal closed-form solution is given by

$$\mathbf{F} = (1-\gamma)(\mathbf{I} - \gamma\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})^{-1}\mathbf{H} \tag{14}$$

where $\gamma = \frac{1}{\lambda+1}$.

Remark. Note that, the above Laplacian regularization models have been explored for graph node ranking and label propagation [34, 35, 5, 11]. Here we propose to explore them for feature diffusion task whose aim is to learn a contextual feature representation for graph node while preserve the unary information of each node.

Table 1 summarizes the feature diffusion operators based on the proposed three diffusion models. In additional to these methods, some other diffusion models [5, 20] can also be explored here. Generally, these models have three main desired properties. First, they conduct feature diffusion while preserving the information of original feature $\mathbf{H}$ in feature diffusion process. Second, they all have an explicit equilibrium representation which can be computed via a simple closed-form solution and thus can be computed efficiently. Third, they can be naturally extended to address multiple graphs, as shown in §3.4 in detail.

Table 1. Summary of the proposed feature diffusion methods.

Model	Diffusion operator $\mathcal{F}_d(\mathbf{A}, \mathbf{H})$
RWR Eq.(9)	$(1-\lambda)(\mathbf{I} - \lambda\mathbf{A}\mathbf{D}^{-1})^{-1}\mathbf{H}$
LapReg Eq.(11)	$\lambda(\mathbf{D} - \mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{H}$
NLapReg Eq.(13)	$(1-\gamma)(\mathbf{I} - \gamma\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})^{-1}\mathbf{H}$

3.3. GDEN architecture

In this section, we present our Graph Diffusion-Embedding Networks (GDENs) based on the proposed feature diffusion operators. Similar to the structure of GCNs [15], GDENs contain one input layer, several hidden layers and one final perceptron layer. For hidden layer, it takes a feature matrix $\mathbf{H}^{(k)} \in \mathbb{R}^{n\times d_k}$ as the input and outputs a feature map matrix $\mathbf{H}^{(k+1)} \in \mathbb{R}^{n\times d_{k+1}}$ by using diffusion and embedding operators. The diffusion operator aims to learn a contextual feature representation for graph nodes by exploring graph structure while the embedding operator is used to learn a low-dimensional representation for graph nodes.

Formally, given an input feature matrix $\mathbf{H}^{(0)} \in \mathbb{R}^{n\times d_0}$ and graph $\mathbf{A} \in \mathbb{R}^{n\times n}$, GDENs conduct the following layer-wise propagation as,

$$\mathbf{H}^{(k+1)} = \sigma\big(\mathcal{F}_d(\mathbf{A}, \mathbf{H}^{(k)})\mathbf{W}^{(k)}\big) \tag{15}$$

where $k = 0, 1, \cdots K-1$ and $\mathcal{F}_d(\mathbf{A}, \mathbf{H}^{(k)})$ denotes the contextual (diffused) feature representation based on graph $\mathbf{A}$ and feature map matrix $\mathbf{H}^{(k)}$. We can use any kind of diffusion models $\mathcal{F}_d(\mathbf{A}, \mathbf{H}^{(k)})$ listed in Table 1. $\mathbf{W}^{(k)} \in \mathbb{R}^{d_k\times d_{k+1}}$ is a layer-specific trainable weight matrix to conduct linear embedding. Function $\sigma(\cdot)$ denotes an activation function, such as $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$.

The last layer of GDENs output the final representation of graph nodes $\mathbf{H}^{(K)}$, which can be used for graph classification [4], graph link prediction [16] and graph node semi-supervised classification task [15]. For semi-supervised learning task, we add a softmax activation function on each row of the final output feature map matrix $\mathbf{H}^{(K)}$ as,

$$\mathbf{Z}_{ij} = \mathrm{softmax}(\mathbf{H}^{(K)}) = \frac{\exp(\mathbf{H}^{(K)}_{ij})}{\sum_{j=1}^{c}\exp(\mathbf{H}^{(K)}_{ij})} \tag{16}$$

where c denotes the number of class. The final obtained output $\mathbf{Z} \in \mathbb{R}^{n\times c}$ gives the final label prediction for all graph nodes in which each row $\mathbf{Z}_i$ of matrix $\mathbf{Z}$ denotes the corresponding label prediction vector for the i-th node. The optimal weight parameters $\mathcal{W} = \{\mathbf{W}^0, \mathbf{W}^1, \cdots \mathbf{W}^{(K)}\}$ of GDENs are obtained by minimizing the following cross-entropy loss function over all the labeled nodes L, i.e.,

$$\mathcal{L}_{\text{Semi-GDEN}} = -\sum\nolimits_{i\in L}\sum\nolimits_{j=1}^{c}\mathbf{Y}_{ij}\ln\mathbf{Z}_{ij} \tag{17}$$

where L indicates the set of labeled nodes and $\mathbf{Y}_{i\cdot}, i \in L$ denotes the corresponding label indication vector for the i-th labeled node, i.e.,

$$\mathbf{Y}_{ij} = \begin{cases} 1 & \text{if node } i \text{ belongs to } j\text{-th class} \\ 0 & \text{otherwise} \end{cases}$$

Figure 1 shows the training loss values across different epochs. One can note that, GDENs obtain obviously lower cross-entropy loss values than GCN [15] at convergence, which clearly demonstrates the higher predictive accuracy of GDENs model. Also, the convergence speeds of GDENs are faster than GCN, indicating the efficiency of GDENs. Figure 2 demonstrates the 2D visualizations of the feature map output by the first hidden layer of GCN and GDEN (with normalized Laplacian diffusion), respectively. Different colors denote different classes. Intuitively, one can observe that the data of different classes are distributed more clearly in our GDEN representation, which demonstrates the benefits of GDEN on conducting graph data representation and learning.

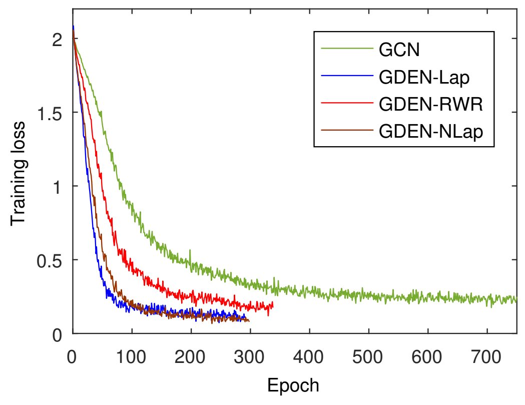


Figure 1. Demonstration of cross-entropy loss values across different epochs on Cora dataset.

3.4. GDENs on multiple graphs

One important property of the proposed GDENs is that they can naturally address the structured data with multiple graph representations. This is because we can extend Laplacian regularized diffusion models (Eq.(11) and Eq.(13)) on multiple graphs to provide multiple graph feature diffusion operators. Formally, given an input feature matrix $\mathbf{H}^{(0)} \in \mathbb{R}^{n\times d_0}$ with multiple graph representations $\mathcal{A} = \{\mathbf{A}^{(1)}\cdots\mathbf{A}^{(m)}\}$, we can obtain an optimal feature diffusion $\mathbf{F} = \mathcal{F}_d(\mathbf{A}^{(1)}, \mathbf{A}^{(2)}\cdots\mathbf{A}^{(m)}; \mathbf{H})$ on graph set $\mathcal{A}$ by optimizing

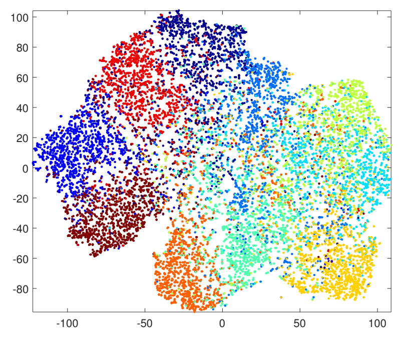
$$\min_{\mathbf{F}}\ \frac{1}{2}\sum_{v=1}^{m}\Big(\sum_{i,j=1}^{n}\alpha_v^r\mathbf{A}^{(v)}_{ij}\|\mathbf{f}_i - \mathbf{f}_j\|_2^2\Big) + \lambda\sum_{i=1}^{n}\|\mathbf{f}_i - \mathbf{h}_i\|_2^2$$
$$s.t.\ \sum\nolimits_{v=1}^{m}\alpha_v = 1, \alpha_v \geq 0 \tag{18}$$

where $\alpha = (\alpha_1, \alpha_2\cdots\alpha_m)$ denote the important weights of different graphs which are learned adaptively. The parameter $r > 1$ is used to control the weight distribution, as suggested in previous work [32]. First, we rewrite Eq.(18) more compactly as

$$\min_{\mathbf{F}}\ \sum\nolimits_{v=1}^{m}\alpha_v^r\mathrm{Tr}(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}) + \lambda\mathrm{Tr}(\mathbf{F}^T\mathbf{F} - 2\mathbf{F}^T\mathbf{H})$$
$$s.t.\ \sum\nolimits_{v=1}^{m}\alpha_v = 1, \alpha_v \geq 0 \tag{19}$$

where $\mathbf{L}^{(v)} = \mathbf{D}^{(v)} - \mathbf{A}^{(v)}$ and $\mathbf{D}^{(v)} = \mathrm{diag}\{\mathbf{d}^{(v)}_1\cdots\mathbf{d}^{(v)}_n\}$ and $\mathbf{d}^{(v)}_i = \sum_j\mathbf{A}^{(v)}_{ij}$. The equivalent Lagrangian form is

$$\min_{\mathbf{F}}\sum\nolimits_{v=1}^{m}\alpha_v^r\mathrm{Tr}(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}) + \lambda\mathrm{Tr}(\mathbf{F}^T\mathbf{F} - 2\mathbf{F}^T\mathbf{H})$$
$$+\ \xi\Big(\sum\nolimits_{v=1}^{m}\alpha_v - 1\Big) \tag{20}$$
$$s.t.\ \alpha_v \geq 0$$


(a) GCN result

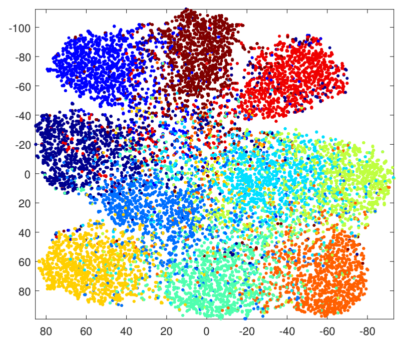

(b) GDEN result

Figure 2. t-SNE [21] visualizations of the feature maps output by the first hidden layer of GCN and GDEN, respectively on the CIFAR10 dataset. Different classes are marked by different colors. Note that, the data distribution of different classes is demonstrated more clearly in our GDEN representation.

where ξ is the Lagrangian multiplier. Fix α and obtain the optimal $\mathbf{F}$ by setting the derivation w.r.t $\mathbf{F}$ to zero and obtain

$$\mathbf{F} = \lambda\big(\lambda\mathbf{I} + \sum\nolimits_{v=1}^{m} \alpha_v^r \mathbf{L}^{(v)}\big)^{-1}\mathbf{H} \tag{21}$$

Fix $\mathbf{F}$ and we similarly obtain the optimal as α.

$$\alpha = \frac{(1/\mathrm{Tr}(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}))^{\frac{1}{r-1}}}{\sum_{v=1}^{m}(1/\mathrm{Tr}(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}))^{\frac{1}{r-1}}} \tag{22}$$

Therefore, the optimal $\mathbf{F}$ can be obtained by alternatively updating $\mathbf{F}$ and α as follows until convergence,

$$\mathbf{F} \leftarrow \lambda\big(\lambda\mathbf{I} + \sum\nolimits_{v=1}^{m} \alpha_v^r \mathbf{L}^{(v)}\big)^{-1}\mathbf{H} \tag{23}$$

$$\alpha \leftarrow \frac{(1/\mathrm{Tr}(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}))^{\frac{1}{r-1}}}{\sum_{v=1}^{m}(1/\mathrm{Tr}(\mathbf{F}^T\mathbf{L}^{(v)}\mathbf{F}))^{\frac{1}{r-1}}} \tag{24}$$

Empirically, the algorithm converges quickly, generally less than 10 iterations. Similarly, we can also derive multiple graph feature diffusion based on normalized Laplacian regularization diffusion model (Eq.(13)).

Based on the above multiple graph feature diffusion operators, we can extend GDEN on multiple graphs and present our Multiple GDENs (M-GDENs). Formally, given an input feature matrix $\mathbf{H}^{(0)} \in \mathbb{R}^{n\times d_0}$ and multiple graphs $\mathcal{A} = \{\mathbf{A}^{(1)} \cdots \mathbf{A}^{(m)}\}$, M-GDENs conduct the following layer-wise propagation as,

$$\mathbf{H}^{(k+1)} = \sigma\big(\mathcal{F}_d(\mathbf{A}^{(1)} \cdots \mathbf{A}^{(m)}; \mathbf{H}^{(k)})\mathbf{W}^{(k)}\big) \tag{25}$$

where $k = 0, 1, \cdots K-1$ and $\mathcal{F}_d(\mathbf{A}^{(1)} \cdots \mathbf{A}^{(m)}; \mathbf{H}^{(k)})$ denotes the contextual (diffused) feature representation which is obtained by computing the optimal solution of multiple graph diffusion Eq.(18) via update algorithm Eqs.(23,24).

3.5. Comparison with related works

We provide the detail comparisons with some recent works including Graph Convolutional Network (GCN) [15], Diffusion Convolutional Neural Network (DCNN) [1], Diffusion Convolutional Recurrent Neural Network (DCRN-N) [19] and Graph Attention Networks (GAT) [30]. Both GCN [15] and GAT [30] use a one-step diffusion operation while DCRNN [19] utilizes a finite K-step truncation of diffusion on graph. In DCNN [1], it uses a standard random walk based diffusion process. One main limitation for these methods is that the equilibrium representation of feature diffusion is not obtained, which thus may lead to weak contextual feature representation in their networks. Also, these models can not be used directly for the data with multiple graph structures. In contrast, in GDENs, we explore *regularized diffusion* (Eq.(9), Eq.(11) and Eq.(13)) in our diffusion-embedding architecture for contextual feature representation. The main benefits are three aspects. First, they all have an explicit equilibrium representation which encourage GDENs to generate more compact feature representations for graph data. Second, they conduct feature diffusion while preserve the information of original input feature in feature representation process. Thus, they present an effective representation by integrating both unary feature information of graph nodes and contextual feature information from neighboring nodes simultaneously. Third, they can be naturally extended to address multiple graphs.

4. Experiments

4.1. Datasets

We test our GDENs on five datasets including three standard citation network benchmark datasets (Citeseer, Cora and Pubmed [28]) and two widely used image datasets (CIFAR10 [17] and SVHN [23]. The details of these datasets and their usages in our experiments are introduced below.

Citeseer dataset. It is a citation network which contains 3327 nodes and 4732 edges. The nodes of this network are falling into six categories and each node has a 3703 dimension feature descriptor.

Cora dataset. Similar to Citeseer dataset, it contains 2708 nodes and 5429 edges. Each node has a 1433 dimension descriptor and all nodes are falling into six categories.

Pubmed dataset. This is a larger network data which contains 19717 nodes and 44338 edges in all. Each node has a 500 dimension feature descriptor and all the nodes are falling into three categories.

CIFAR10 dataset. It contains 10 classes of 50000 RGB images with the same size 32×32 [17]. In our experiments, we select 1500 images per class and use 15000 images in all for evaluation. For each image, we first extract a CNN feature descriptor for it. Then, we construct a k nearest neighbor graph ($k = 3$) with nodes denoting images and edges representing the neighborhood relationship between images. The weight of each edge is computed by Heat kernel function.

SVHN dataset. It contains 73257 training and 26032 test RGB images with the same size 32×32 [23]. Each image contains several digits and the task is to classify the digit in the center of image. Similar to CIFAR10, we select 1500 images per class and use 15000 images in all for evaluation. We extract a CNN feature for each image and construct a k nearest neighbor graph ($k = 3$) for images. The edge weight is computed by Heat kernel function.

4.2. Experimental setup

For Cora, Citeseer and Pubmed datasets, we follow the experimental setup in work [15, 30]. For image dataset CIFAR10 [17] and SVHN [23], we randomly select 1000, 2000 and 4000 images as labeled samples and use the remaining data as unlabeled samples. For unlabeled samples, we select 1000 images for validation purpose and use the remaining 13000, 12000 and 10000 images as test samples. All the accuracy results are averaged over 10 runs with different data splits. We implement our GDEN with three versions, i.e., 1) GDEN-RWR that utilizes random walk with restart (RWR) based diffusion operation in G-DEN; 2) GDEN-Lap that utilizes graph Laplacian diffusion operation in GDEN; 3) GDEN-NLap that utilizes normalized Laplacian diffusion operation in GDEN. The optimal parameter λ in GDEN-RWR, GDEN-Lap and γ in GDEN-NLap are determined based on the validation loss values. Similar to [15], we set the number of diffusion-embedding layers in GDENs to 2. The number of units in each hidden layer is set to 40. We present additional experiments on performance of GDENs with different hidden unit numbers and convolutional layers in §4.4. We train GDENs for a maximum of 2000 epochs (training iterations) using Adam [14] with a learning rate of 0.01. Similar to [30], we stop training if the validation loss does not decrease for 100 consecutive epochs. All the network weights $\{\mathbf{W}^{(0)}\cdots\mathbf{W}^{(K)}\}$ are initialized using Glorot initialization [8].

4.3. Comparison to state-of-the-art methods

We compare our method with some other graph (neural network) based semi-supervised learning methods. The compared methods contain i) two graph based semi-supervised learning methods including Label Propagation (LP) [36] and Manifold Regularization (ManiReg) [2]; ii) two deep neural network based semi-supervised learning methods including Planetoid [33] and DeepWalk [26]; iii) three graph neural network methods including Graph Convolutional Network (GCN) [15], Diffusion Convolutional Neural Networks (DCNNs) [1] and Graph Attention Networks (GAT) [30]. The codes of DCNN, GCN and GAT have been provided by authors and we use them in our experiments. For fair comparison, the parameter settings of GCN are the same as our GDEN to obtain the average better performance on all datasets.

Table 2. Comparison results on citation network datasets

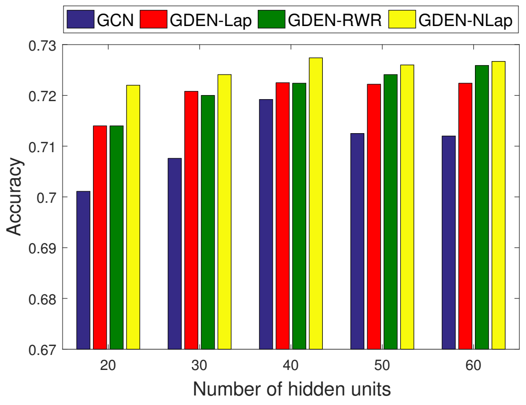
Methond	Citeseer	Cora	Pubmed
ManiReg [2]	60.1%	59.5%	70.7%
LP [36]	45.3%	68.0%	63.0%
DeepWalk [26]	43.2%	67.2%	65.3%
Planetoid [33]	64.7%	75.7%	77.2%
DCNN [1]	64.5%	76.7%	75.3%
GCN [15]	70.3%	83.6%	78.3%
GAT [30]	71.0%	83.2%	78.0%
GDEN-RWR	**72.8%**	82.0%	78.7%
GDEN-Lap	72.6%	84.7%	**78.9**%
GDEN-NLap	70.3%	**85.1%**	78.7%

Table 2 summarizes the comparison results on three network benchmark datasets[1]. The best results are marked by bold. Here we can note that, 1) GDENs generally perform better than GCN [15], DCNN [1] and GAT [30]. That is, comparing with some other competing graph convolution architectures, the proposed diffusion-embedding architectures in GDENs are more effective for graph data representation and learning. 2) GDENs perform better than other semi-supervised learning methods, such as Planetoid [33] and DeepWalk [26], which indicates the effectiveness and benefits of GDENs on conducting graph based semi-supervised learning tasks. Table 3 summarizes the comparison results on two widely used image datasets (CIFAR10 [17] and SVHN [23]). The highest results are marked by bold. Overall, we can note that 1) GDENs perform better than re-

[1]GCN results in Table 2 are slightly different from that reported in work [15], because we use a different hidden layer setting to make it consistent with GDEN and also obtain average better results on all the network and image datasets.

Table 3. Comparison results on SVHN and CIFAR10 datasets

Dateset	SVHN			CIFAR10		
No. of label	1000	2000	4000	1000	2000	4000
ManiReg [2]	79.38±0.53%	82.14±0.39%	84.24±0.42%	66.49±0.49%	70.26±0.37%	73.62±0.30%
LP [36]	64.82±0.92%	67.05±0.60%	69.84±0.45%	57.45±0.99%	59.30±1.07%	60.97±0.57%
DeepWalk [26]	72.85±0.58%	74.69±0.34%	76.38±0.43%	57.09±0.41%	59.93±0.29%	62.51±0.29%
Planetoid [33]	79.48±0.71%	82.09±0.49%	83.96±0.40%	64.24±0.61%	68.01±0.44%	71.34±0.41%
GCN [15]	81.07±0.52%	81.90±0.36%	82.42±0.40%	68.86±0.50%	71.55±0.29%	71.55±0.29%
GAT [30]	76.45±0.29%	77.38±0.24%	77.71±0.34%	64.33±0.65%	65.21±0.40%	65.94±0.36%
GDEN-L	82.18±0.48%	84.04±0.42%	85.44±0.29%	69.08±0.53%	72.17±0.27%	75.03±0.31%
GDEN-RWR	82.72±0.53%	84.76±0.33%	**86.36±0.41%**	68.60±0.52%	72.12±0.42%	75.19±0.35%
GDEN-NL	**82.88±0.54%**	**84.80±0.39%**	86.24±0.36%	**69.08±0.53%**	**72.56±0.29%**	**75.46±0.21%**



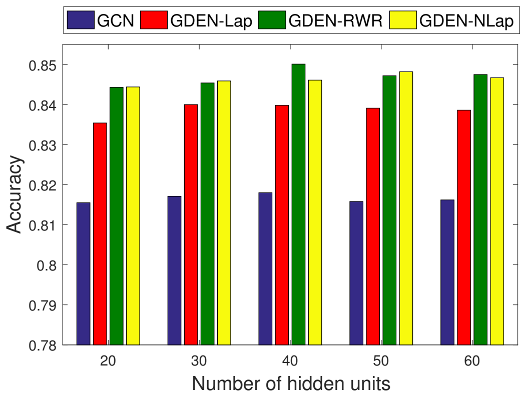


Figure 3. Results of two-layer GDENs across different unit number in hidden layers (LEFT: CIFAR10 dataset; RIGHT: SVHN dataset).

cent GCN models (GCN [15], GAT [30]) on both datasets, which further demonstrates the effectiveness of the proposed diffusion-embedding architecture to enhance the ability of GDENs for graph data learning. 2) GDENs outperform the other graph based semi-supervised learning methods and obtain the best performances on both datasets, indicating the better performance of the proposed GDENs based semi-supervised learning on image data. 3) GDEN-NLap usually performs better than GDEN-RWR and GDEN-Lap on these two datasets.

4.4. Evaluation on GDENs setting

We evaluate the performance of GDENs under different network settings. As a baseline method, we also report the performance of GCN [15] with the same setting.

Analysis on number of hidden units. We first evaluate the performance of GDENs across different number of hidden units. Figure 3 summarizes the results on dataset CIFAR10 [17] and SVHN [23], respectively. We can observe that GDENs generally perform insensitively w.r.t. the number of units in the hidden layer.Also, GDENs always outperform GCN on all settings of unit numbers in hidden layer.

Analysis on model depth. We then investigate the influence of model depth (number of hidden layers) on final performance of GDENs. Figure 4 shows the results on CIFAR10 [17] and SVHN [23], respectively. Note that GDENs can obtain desired better results using a simple two-layer setting. This phenomenon also occurs in many other graph neural networks, such as GCN [15] and GAT [30]. GDENs are generally insensitive w.r.t. model depth. It maintains better performs under different numbers of hidden layers. Also, GDENs generally outperform GCN on all settings of model depth, indicating the better performance of GDENs.

4.5. Evaluation on M-GDEN

Our final evaluation is to verify the effectiveness of the proposed M-GDEN on the data with multiple graphs. Here, we evaluate the performance of the proposed M-GDEN for semi-supervised learning on two datasets, i.e., MSRC-v1 [31] and Caltech101-7 [18, 25]. MSRC-v1 contains 210 images which are falling into 7 classes. Following the experimental setting in work [25], we construct five graphs (5 nearest neighbor graph) for this dataset based on five different kinds of visual descriptors. Caltech101-7 [18, 24] is an object recognition data set containing 101 categories of images. We follow the experimental setup of previous work [24] and select the widely used 7 classes. We construct six graphs (5 nearest neighbor graph) for this dataset

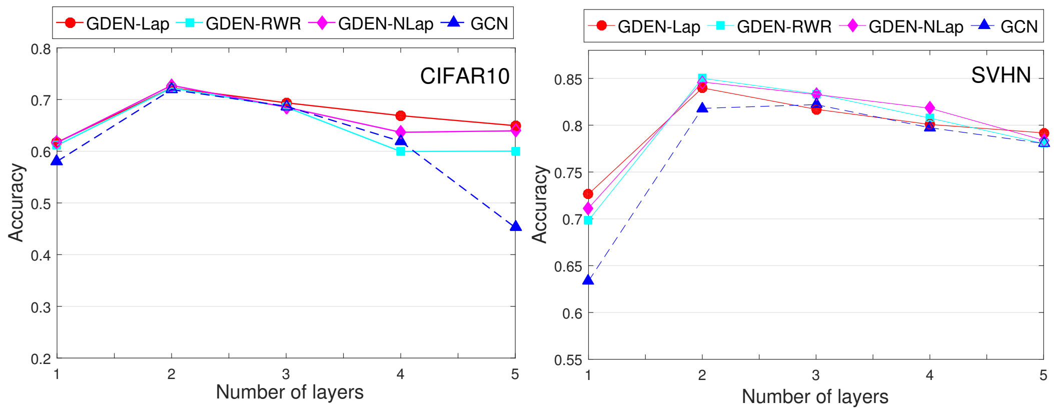


Figure 4. Results of GDENs across different network layers (LEFT: CIFAR10 dataset; RIGHT: SVHN dataset).

based on six kinds of visual feature descriptors. Table 4 summarizes the semi-supervised classification results under different rates of labeled data on these two datasets, respectively. As a baseline, we also report the classification results of GCN method [15]. Since GCN can not be used to deal with multiple graphs, we implement it in two variants, i.e., i) GCN(v) that conducts convolution on the v-th individual graph $\mathbf{A}^{(v)}$; ii) GCN-A that conducts convolution on the averaged graph $\bar{\mathbf{A}} = \frac{1}{m}\sum_{v=1}^{m}\mathbf{A}^{(v)}$. Here, one can note that, comparing with baseline methods, our method obtains the best performance on both datasets, which demonstrates the effectiveness of M-GDEN on multiple graphs.

Table 4. Semi-supervised classification results on multiple graphs.

Dateset	MSRC-v1			Caltech101-7		
Rate	10%	20%	30%	10%	20%	30%
GCN(1)	47.86	49.25	49.21	82.13	84.93	84.88
GCN(2)	70.77	76.26	80.00	82.47	83.73	84.92
GCN(3)	69.46	83.33	84.29	85.62	86.03	87.63
GCN(4)	67.14	72.59	84.29	92.06	94.78	94.74
GCN(5)	67.14	75.7	73.33	90.27	93.82	94.09
GCN(6)	-	-	-	89.54	92.76	93.74
GCN-A	80.36	87.14	88.73	91.05	94.09	94.68
M-GDEN	**84.70**	**89.59**	**90.70**	**92.69**	**95.77**	**96.74**

5. Conclusion and Future Work

We present Graph Diffusion-Embedding Networks (G-DENs) for graph-structured data representation and learning. GDENs integrate both feature diffusion and embedding simultaneously in a unified network by introducing a new diffusion-embedding architecture. The diffusion-embedding architecture in GDENs can produce an equilibrium representation for graph nodes by exploring graph structure, which makes GDENs be able to obtain a more compact representation for graph data. Furthermore, GDENs provide a straightforward mechanism to address structured data with multiple graphs. Experimental results on several widely used benchmarks demonstrate that GDENs significantly outperform the state-of-the-art graph neural network models on various semi-supervised learning tasks.

In our future, we will develop some approximate algorithms to compute the inversion operation in GDENs and thus to make GDENs more efficiently in practical.

Acknowledgment

This work is supported in part by NSFC Key Projects of International (Regional) Cooperation and Exchanges under Grant (61860206004); National Natural Science Foundation of China (61602001, 61872005, 61671018); Natural Science Foundation of Anhui Province (1708085QF139); Natural Science Foundation of Anhui Higher Education Institutions of China (KJ2016A020).

References

[1] James Atwood and Don Towsley. Diffusion-convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1993–2001, 2016.

[2] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(Nov):2399–2434, 2006.

[3] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014.

[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[5] Michael Donoser and Horst Bischof. Diffusion processes for retrieval revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1320–1327, 2013.

[6] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, pages 2224–2232, 2015.

[7] Paolo Frasconi, Marco Gori, and Alessandro Sperduti. A general framework for adaptive processing of data structures. *IEEE transactions on Neural Networks*, 9(5):768–786, 1998.

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.

[9] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *IEEE International Joint Conference on Neural Networks*, pages 729–734, 2005.

[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[11] Bo Jiang, Chris Ding, Bio Luo, and Jin Tang. Graph-laplacian pca: Closed-form solution and robustness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3492–3498, 2013.

[12] Bo Jiang, Zhouqin He, Chris Ding, and Bin Luo. Saliency detection via a multi-layer graph based diffusion model. *Neurocomputing*, 314:215–223, 2018.

[13] Tae Hoon Kim, Kyoung Mu Lee, and Sang Uk Lee. Generative image segmentation using random walks with restart. In *European conference on computer vision*, pages 264–275, 2008.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[16] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[17] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[18] Yeqing Li, Feiping Nie, Heng Huang, and Junzhou Huang. Large-scale multi-view spectral clustering via bipartite graph. In *AAAI*, pages 2750–2756, 2015.

[19] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018.

[20] Song Lu, Vijay Mahadevan, and Nuno Vasconcelos. Learning optimal seeds for diffusion-based salient object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2797, 2014.

[21] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9:2579–2605, 2008.

[22] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5423–5434, 2017.

[23] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011.

[24] Feiping Nie, Guohao Cai, and Xuelong Li. Multi-view clustering and semi-supervised classification with adaptive neighbours. In *AAAI*, pages 2408–2414, 2017.

[25] Feiping Nie, Jing Li, Xuelong Li, et al. Parameter-free auto-weighted multiple graph learning: A framework for multiview clustering and semi-supervised classification. In *IJCAI*, pages 1881–1887, 2016.

[26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[27] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[28] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93, 2008.

[29] Alessandro Sperduti and Antonina Starita. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8(3):714–735, 1997.

[30] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[31] John Winn and Nebojsa Jojic. Locus: Learning object classes with unsupervised segmentation. In *IEEE International Conference on Computer Vision*, pages 756–763, 2005.

[32] Tian Xia, Dacheng Tao, Tao Mei, and Yongdong Zhang. Multiview spectral embedding. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(6):1438–1446, 2010.

[33] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*, pages 40–48, 2016.

[34] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pages 321–328, 2004.

[35] Dengyong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet, and Bernhard Schölkopf. Ranking on data manifolds. In *Advances in neural information processing systems*, pages 169–176, 2004.

[36] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *International conference on Machine learning*, pages 912–919, 2003.