Semi-supervised Learning with Adaptive Neighborhood Graph Propagation Network

Bo Jiang, Leiling Wang, Jin Tang and Bin Luo

School of Computer Science and Technology, Anhui University
jiangbo@ahu.edu.cn

Abstract—Graph methods have been commonly employed for visual data representation and analysis in computer vision area. Graph data representation and learning becomes more and more important in this area. Recently, Graph Convolutional Networks (GCNs) have been employed for graph data representation and learning. Existing GCNs usually use a fixed neighborhood graph which is not guaranteed to be optimal for semi-supervised learning tasks. In this paper, we propose a unified adaptive neighborhood feature propagation model and derive a novel Adaptive Neighborhood Graph Propagation Network (ANGPN) for data representation and semi-supervised learning. The aim of ANGPN is to conduct both neighborhood graph construction (from pair-wise distance information) and graph convolution simultaneously in a unified formulation and thus can learn an optimal neighborhood graph that best serves graph convolution for graph based semi-supervised learning. Experimental results demonstrate the effectiveness and benefit of the proposed ANGPN on image data semi-supervised classification tasks.

I. INTRODUCTION

Compact data representation and learning is a fundamental problem in computer vision area. Graph methods have been commonly employed for visual data representation and analysis, such as classification, segmentation, etc. The core technique in graph methods is graph representation and learning. Recently, there is an increasing attention on trying to generalize CNNs to Graph Convolutional Networks (GCNs) to deal with graph data [1] [2] [3] [4]. For example, Bruna et al. [5] employ a eigen-decomposition of graph Laplacian matrix to define graph convolution operation. Henaff et al. [6] introduce a spatially constrained spectral filters to define graph convolution. Defferrard et al. [1] propose to utilize Chebyshev expansion to approximate the spectral filters for graph convolution operation. Kipf et al. [7] present a simple Graph Convolutional Network (GCN). The main idea of GCN is to explore the first-order approximation of spectral filters to compute graph convolution more efficiently. Monti et al. [2] propose mixture model CNNs (MoNet) for graph data learning and analysis. Veličković et al. [4] propose Graph Attention Networks (GAT) for graph based semi-supervised learning which can aggregate the features of neighboring nodes in an adaptive weighting manner. Hamilton et al. [3] present a general inductive representation and learning framework (GraphSAGE) by sampling and aggregating features from a node’s local neighborhood. Klicpera et al. [8] propose to integrate PageRank propagation into GCN in layer-wise propagation. Petar et al. [9] propose Deep Graph Infomax (DGI) to learn the node representation in unsupervised manner. Jiang et al., [10] recently propose Graph Diffusion-Embedding Network (GEDNs) for graph node semi-supervised learning.

The above state-of-the-art methods can be widely used in many graph learning tasks, such as graph based semi-supervised learning, low-dimensional representation, clustering etc. In computer vision applications, for these learning tasks, existing methods usually need to use a two-stage framework, i.e., 1) obtaining/constructing a neighborhood graph from data and 2) conducting graph convolutional representation and learning on this graph. However, one main issue is that such a two-stage framework does not fully mine the correlation between graph construction and graph convolutional learning, which may lead to weak suboptimal solution. Previous works generally focus on graph convolution operation while little attention has been put on graph construction. To address this issue, Li et al. [11] present an optimal graph CNNs, in which the graph is learned adaptively by employing a metric learning method. Veličković et al. [4] propose Graph Attention Networks (GAT) for graph based semi-supervised learning. Jiang et al. [12] propose Graph Optimized Convolutional Network (GOCN) to learn the node representation in unsupervised manner. Klicpera et al. [13] propose Graph mask Convolutional Network (GmCN) by selecting neighbors adaptively in GCN operation.

In this paper, motivated by recent graph learning [14], [15] and graph optimized GCN works [8], [12], [13], we propose a novel graph neural network, named Adaptive Neighborhood Graph Propagation Network (ANGPN), for graph based semi-supervised learning task. Similar to [12], [13], ANGPN aims to conduct both neighborhood graph learning and graph convolution together and formulate graph construction and convolution simultaneously. Different from previous works [4], [12], [13], ANGPN aims to learn an optimal neighborhood graph for graph convolution from pair-wise distance information and can be used in many computer vision tasks which do not have any prior graph structures. Overall, the main contributions of this paper are summarized as follows:

- We propose a novel Adaptive Neighborhood Graph Propagation Network (ANGPN) which integrates both neighborhood graph construction and graph convolution cooperatively in a unified formulation for data representation and semi-supervised learning problem.
- We propose a novel model of Adaptive Neighborhood
Feature Propagation (ANFP) which provides a context-aware representation for data by jointly employing both unary feature of each data and contextual features from its neighbors together.

Experimental results on several datasets demonstrate the effectiveness of ANFP on semi-supervised learning tasks.

II. ADAPTIVE NEIGHBORHOOD FEATURE PROPAGATION

In the following, we first introduce a neighborhood graph feature propagation (NFP) model. Then we extend it to Adaptive NFP (ANFP) by constructing the neighborhood graph in NFP adaptively.

A. Neighborhood feature propagation

Let $M \in \mathbb{R}^{n \times n}$ be the adjacency matrix of a neighborhood graph with $M_{ii} = 0$, i.e., if node $v_i \in \mathcal{N}(v_j)$ then $M_{ij} = 1$, otherwise $M_{ij} = 0$. Let $A = MD^{-1}$ be the row-normalization of $M$, where $D$ is a diagonal matrix with $D_{ii} = \sum_j M_{ij}$. Thus, we have $\sum_{j=1}^{n} A_{ij} = 1$, $A_{ij} \geq 0$ and $A_{ii} = 0$. Let $H = (h_1, h_2, \cdots, h_n) \in \mathbb{R}^{n \times d}$ be the feature descriptors of graph nodes. Then, the aim of our Neighborhood Graph Based Feature Propagation is to learn a kind of feature representation $F = \mathcal{P}(A, H) = (f_1, f_2, \cdots, f_n) \in \mathbb{R}^{n \times d}$ for graph nodes by incorporating the contextual information of their neighbors. Here, we propose a scheme that is similar to the one in label propagation [16] but iteratively propagates the features $H$ on neighborhood graph $A$. Formally, for each node $v_i$, we absorb a fraction of feature information from its neighbors in each propagation step as follows,

$$f_i^{(t+1)} = \alpha \sum_{j=1}^{n} A_{ij} f_j^{(t)} + (1 - \alpha)h_i$$

(1)

where $t = 0, 1, \cdots$ and $f_j^{(0)} = h_j$. Parameter $\alpha \in (0, 1)$ is the fraction of feature information that node $v_i$ receives from its neighbors. We can rewrite Eq.(1) more compactly as

$$F^{(t+1)} = \alpha AF^{(t)} + (1 - \alpha)H$$

(2)

where $F^{(0)} = H$. It is known that the above propagation will converge to an equilibrium solution as

$$F^* = (1 - \alpha)(I - \alpha A)^{-1}H$$

(3)

B. Adaptive neighborhood feature propagation

In the above feature propagation, one important aspect is the construction of neighborhood graph $A$. One popular way is to construct a human established graph, such as k-nearest neighbor graph. However, such a pre-defined graph may have no clear structure and thus also be not guaranteed to best serve the feature propagation and learning task. To overcome this issue, we propose to learn an adaptive neighborhood graph $S$ to better capture the intrinsic neighborhood relationship among data in the above NFP. We call it as Adaptive Neighborhood Feature Propagation (ANFP).

In order to do so, we first show that the above propagation has an equivalent optimization formulation [10], [12], [16]. In particular, the converged solution of Eq.(3) is the optimal solution that minimizes the following optimization problem,

$$\min_{F} \text{Tr}(F^T(I - A)F) + \mu\|F - H\|_F^2$$

(4)

where $\alpha = \frac{1}{1 + \beta}$, $\text{Tr}(\cdot)$ denotes the trace function and $\| \cdot \|_F$ denotes the Frobenious norm of matrix. Then, inspired by [14], [15], we incorporate graph learning into Eq.(4) and construct a unified model as

$$\min_{S, F} \sum_{i,j=1}^{n} D_{ij}S_{ij} + \gamma\|S\|_F^2 + \beta \text{Tr}(F^T(I - S)F) + \mu\|F - H\|_F^2$$

s.t. $\sum_{j=1}^{n} S_{ij} = 1, S_{ij} \geq 0$

(5)

where $D_{ij} = \|x_i - x_j\|_2$ denotes the Euclidean distance between original input feature $x_i$ and $x_j$ of data. The optimal $S_{ij}$ can be regarded as a confidence (or probability) that data $x_j$ is connected to $x_i$ as a neighbor. A larger distance $D_{ij}$ should be assigned with smaller confidence $S_{ij}$. The parameter $\gamma > 0$ is used to control the sparsity of learned graph $S$ [14]. We call it as Adaptive Neighborhood Feature Propagation (ANFP). The optimal $S$ and $F$ can be obtained via a simple algorithm which alternatively conducts the following Step 1 and Step 2 until convergence.

Step 1. Solve $S$ while fixing $F$, the problem becomes

$$\min_{S} \sum_{i,j=1}^{n} D_{ij}S_{ij} + \gamma\|S\|_F^2 + \beta \text{Tr}(F^T(I - S)F)$$

s.t. $\sum_{j=1}^{n} S_{ij} = 1, S_{ij} \geq 0$

(7)

which is equivalent to

$$\min_{S} \sum_{i,j=1}^{n} (F^T - \beta FF^T)_{ij}S_{ij} + \gamma\|S\|_F^2$$

s.t. $\sum_{j=1}^{n} S_{ij} = 1, S_{ij} \geq 0$

(8)

This problem has a simple closed-form solution which is given as [14],

$$S_{ij} = \max \left\{ -\frac{1}{2\gamma} (F^T - \beta FF^T)_{ij} + \eta, 0 \right\}$$

(9)

where $\eta = \frac{1}{2} + \frac{1}{2\gamma} \sum_{j=1}^{k} D_{ij}^x$.

Remark. Without loss of generality, here we suppose that $D_{i1}^x, D_{i2}^x, \cdots, D_{in}^x$ are ordered from small to large values, as discussed in work [14].

Step 2. Solve $F$ while fixing $S$, the problem becomes

$$\min_{F} \text{Tr}(F^T(I - S)F) + \mu\|F - H\|_F^2$$

(10)

This is similar to Eq.(10) and the optimal solution is

$$F = (1 - \alpha)(I - \alpha S)^{-1}H$$

(11)

1For efficiency consideration, we use the original feature $X$ to compute $D^x$ which is fixed in the following training process. Here, one can also use the current learned representation $H$ to obtain/update more accurate $D^x$. 
III. ANGPN ARCHITECTURE

In this section, we present our Adaptive Neighborhood Graph Propagation Network (ANGPN) based on the above ANFP model formulation. Similar to the overall architecture of traditional GCN [7], we employ one input layer, several hidden propagation layers and one final perceptron layer for ANGPN. They are introduced below.

A. Hidden propagation layer

Given an input feature matrix \( H^{(k)} \in \mathbb{R}^{n \times d_k} \) and pair-wise distance matrix \( D^x \in \mathbb{R}^{n \times n} \), ANGPN hidden layer outputs feature map matrix \( H^{(k+1)} \in \mathbb{R}^{n \times d_{k+1}} \) by using the above ANFP and non-linear transformation. More specifically, let \( F = \mathcal{P}(D^x, H) \) be the optimal solution of ANFP, then our ANGPN conducts layer-wise propagation in hidden layer as

\[
H^{(k+1)} = \sigma(\mathcal{P}(D^x, H^{(k)}) W^{(k)}) \tag{12}
\]

where \( D^x \) denotes the distance matrix and \( W^{(k)} \in \mathbb{R}^{d_k \times d_{k+1}} \) is the \( k \)-th layer-specific trainable weight matrix. \( \sigma(\cdot) \) denotes an activation function.

Directly calculating ANFP solution \( F = \mathcal{P}(D^x, H) \) is time consuming due to (A) inversion operation in computing \( F \) (Eq.(11)) and (B) alternative computation of Step 1 and Step 2 until convergence. To reduce the issue of high computational complexity, similar to [12], we derive an approximate algorithm to conduct them in the following. For inversion operation (A), we adopt the strategy similar to GCN [7] and approximate the optimization of \( F \) via an one-step power iteration algorithm. More concretely, instead of calculating Eq.(11) directly, we use the following update to compute \( F \).

\[
F = \alpha S H + (1 - \alpha) H = \left( \alpha S + (1 - \alpha) I \right) H \tag{13}
\]

For computation (B), we can also use a truncated \( T \)-step alternative iteration to optimize ANFP approximately. Based on the above analysis, we summarize the whole propagation algorithm to compute \( \mathcal{P}(D^x, H^{(k)}) \) in hidden propagation layer of ANGPN in Algorithm 1.

**Algorithm 1** ANGPN propagation layer

1: **Input:** Feature matrix \( H^{(k)} \in \mathbb{R}^{n \times d_k} \) and ordered distances \( D \in \mathbb{R}^{n \times n} \), parameters \( \gamma, \beta \) and \( \alpha \), maximum iteration \( T \)
2: **Output:** Feature map \( H^{(k+1)} \)
3: Initialize \( F = H^{(k)} \)
4: Compute \( \eta \) as \( \eta = \frac{1}{\gamma} + \frac{1}{2\beta} \sum_{j=1}^k D^x_{ij} \)
5: for \( t = 1, 2 \ldots T \) do
6: \quad Compute \( S \) as
\[
S_{ij} = \max \left\{ -\frac{1}{\gamma}(D^x - \beta FF^T)_{ij} + \eta, 0 \right\}
\]
7: \quad Compute \( F \) as
\[
F = \left( \alpha S + (1 - \alpha) I \right) H^{(k)}
\]
8: end for
9: Return \( \mathcal{P}(D^x, H^{(k)}) = F \)

B. Perceptron layer

For graph node classification, the last layer of ANGPN outputs the final label prediction \( Z \in \mathbb{R}^{n \times c} \) for graph nodes where \( c \) denotes the number of class. We add a softmax activation function on each row of the final output feature map \( H^{(K)} \in \mathbb{R}^{n \times c} \) as,

\[
Z = \text{softmax}(H^{(K)}) \tag{14}
\]

where each row \( Z_i \) of matrix \( Z \) denotes the corresponding label prediction vector for the \( i \)-th node. The optimal network weight parameters \( \{ W^{(0)}, W^{(1)}, \ldots, W^{(K)} \} \) of ANGPN are trained by minimizing cross-entropy loss function defined as [7],

\[
\mathcal{L}_{\text{Semi-ANGPN}} = - \sum_{i \in L} \sum_{j=1}^c Y_{ij} \ln Z_{ij} \tag{15}
\]

where \( L \) indicates the set of labeled nodes and \( Y_{ij}, i \in L \) denotes the corresponding label indication vector for the \( i \)-th labeled node.

C. Computational complexity

Empirically, the optimization in each layer of ANGPN generally does not lead to very high computational cost because it can be solved via a simple (efficient) algorithm (Algorithm 1) in practical. Overall, ANGPN has similar time consuming with GAT [4]. However, ANGPN outperforms GAT (shown in Table I). Theoretically, in each iteration, the complexity of our graph construction and convolution are \( O(kn) + O(dn^2) \) and \( O(n^2d) \), where \( n, d \) denote the graph size and feature dimension in each layer. The overall complexity is \( O(T(kn + 2dn^2)) \), where \( T \) is number of maximum steps. In our experiments, we set \( T = 2 \) which can return desired better results.

IV. EXPERIMENTS

A. Datasets

We test our method on four benchmark datasets, including SVHN [17], 20News [18], CIFAR [19] and CoraML [20]. The details of these datasets and their usages in our experiments are introduced below.

**SVHN.** This dataset contains 73257 training and 26032 testing images [17]. Each image is a \( 32 \times 32 \) RGB color image which contains multiple number of digits and the task is to recognize the digit in the image center. In our experiments, we select 500 images for each class and obtain 5000 images in all for our evaluation. We have not use all of images because it requires large storage and high computational complexity for graph convolution operation in our ANGPN and other related GCN based methods. We extract a commonly used CNN feature descriptor for each image data.

**CIFAR.** This dataset contains 50000 natural images which are falling into 10 classes [19]. Each image in this dataset is a \( 32 \times 32 \) RGB color image. In our experiments, we select 500 images for each class and use 5000 images in all for our evaluation. For each image, we extract a widely used CNN feature descriptor for it.
ANGPN consistently performs better than recent graph learning methods including DeepWalk [23], Graph Convolutional Network (GCN) [7], Graph Attention Networks (GAT) [4], Deep Graph Informax(DGI) [9] and GraphSAGE [3]. The codes of these comparison methods were provided by authors and we use them directly on our data setting in experiments. In addition, as discussed before, the core aspect of the proposed ANGPN is that it conducts graph construction and graph convolution cooperatively to boost their respective performance. To demonstrate the effectiveness of this cooperative learning, we construct a baseline method, NGPN, which first constructs the graph and conduct feature propagation on it. Then, ANGPN w.r.t. model depth. Also, ANGPN always performs better than GCN under different model depths, which further demonstrates the advantages of ANGPN by conducting graph construction and graph convolution cooperatively and thus can boost their respective performance.

**B. Experimental setting**

For all datasets, we randomly select 10%, 20% and 30% samples in each class as labeled data for training the network and use the other 5% of labeled data for validation purpose. The remaining samples are used as the unlabeled test samples. All the reported results are averaged over five runs with different groups of training, validation and testing data splits. The number of hidden convolution layers in ANGPN is set as 2. The number of units in each hidden layer is set as 50. Some additional experiments across different number of convolution layers are presented in Parameter analysis. We train our ANGPN for a maximum of 10000 epochs (training iterations) by using an ADAM algorithm [24] initialized by Glorot initialization [25] with learning rate 0.005. The parameters \(\{\beta, \alpha\}\) in ANGPN are set to \(\{0.3, 0.5\}\), respectively.

**C. Comparison results**

**Baselines.** We compare against some other related graph approaches which contain i) graph based semi-supervised learning method Label Propagation (LP) [22], Manifold Regularization (ManiReg) [21], and ii) graph neural network methods including DeepWalk [23], Graph Convolutional Network (GCN) [7], Graph Attention Networks (GAT) [4], Deep Graph Informax (DGI) [9] and GraphSAGE [3]. The codes of these comparison methods were provided by authors and we use them directly on our data setting in experiments. In addition, as discussed before, the core aspect of the proposed ANGPN is that it conducts graph construction and graph convolution cooperatively to boost their respective performance. To demonstrate the effectiveness of this cooperative learning, we construct a baseline method, NGPN, which first constructs the graph \(S\) via Eq.(15) with \(\beta = 0\) and then conducts feature propagation on \(S\) via Eq.(19).

**Comparison results.** Table I summarizes the comparison results on four benchmark datasets. The best results are marked as bold in Table I. We can note that, (1) ANGPN outperforms the baseline method GCN [7] on all datasets, demonstrating the desired ability of the proposed ANGPN network on conducting semi-supervised learning tasks by learning neighborhood graph adaptively in data representation and learning. (2) ANGPN consistently performs better than recent graph network model GAT [4], Deep Graph Informax (DGI) [9] and GraphSAGE [3]. (3) ANGPN can obtain better performance than other graph based semi-supervised learning methods, such as LP [22], ManiReg [21] and DeepWalk [23]. It demonstrates the effectiveness of ANGPN on conducting semi-supervised learning tasks.

Table II shows the results of ANGPN vs. NGPN on four datasets. Here, we can note that, ANGPN consistently performs better than the baseline model NGPN. This further demonstrates the advantages of ANGPN by conducting graph construction and graph convolution cooperatively and thus can boost their respective performance.

**D. Parameter analysis**

In this section, we evaluate the performance of ANGPN model with different network settings. Figure 2 shows the performance of ANGPN method across different number of convolutional layers on four datasets, respectively. One can note that ANGPN can obtain better performance with different number of layers, which indicates the insensitivity of ANGPN w.r.t. model depth. Also, ANGPN always performs better than GCN under different model depths, which further demonstrates the benefit and better performance of ANGPN comparing with the baseline method.

**V. Conclusion**

This paper proposes a novel Adaptive Neighborhood Graph Propagation Network (ANGPN) for graph-based semi-supervised learning problem. ANGPN integrates neighborhood graph construction and graph convolution architecture together in a unified formulation, which can learn an optimal neighborhood graph structure that best serves the proposed graph propagation network for data representation and semi-supervised learning problem. Experimental results on four benchmark datasets demonstrate the effectiveness and advantage of ANGPN model on various semi-supervised learning tasks. In the future, we will explore ANGPN model for some other machine learning tasks, such as graph embedding, data clustering etc.
### Table I: Comparison results of different methods on four benchmark datasets. The best results are marked as bold.

| Dataset     | SVHN          | CIFAR         |
|-------------|---------------|---------------|
| No. of label| 10% 20% 30%   | 10% 20% 30%   |
| ManiReg [21]| 71.91 ± 1.50  | 76.43 ± 0.63  |
|             | 78.67 ± 0.71  | 59.44 ± 1.24  |
|             | 65.11 ± 0.79  | 66.04 ± 0.58  |
| LF [22]     | 55.56 ± 0.62  | 66.29 ± 0.78  |
|             | 70.52 ± 0.72  | 52.66 ± 1.40  |
|             | 65.36 ± 1.05  |               |
| DeepWalk [23]| 54.54 ± 0.87  | 76.72 ± 0.83  |
|             | 77.35 ± 1.36  | 58.82 ± 0.96  |
|             | 61.62 ± 0.54  | 63.76 ± 0.82  |
| DGI [9]     | 72.05 ± 0.99  | 74.83 ± 0.46  |
|             | 75.77 ± 0.64  | 58.18 ± 0.57  |
|             | 61.66 ± 0.18  | 64.08 ± 0.39  |
| GraphSage [3]| 73.70 ± 0.92  | 77.22 ± 0.58  |
|             | 78.78 ± 0.31  | 61.85 ± 0.98  |
|             | 63.58 ± 0.68  | 65.21 ± 0.38  |
| GCN [7]     | 75.10 ± 0.80  | 76.45 ± 0.61  |
|             | 77.27 ± 0.57  | 60.36 ± 1.00  |
|             | 63.17 ± 0.61  | 63.87 ± 0.56  |
| GAT [4]     | 72.60 ± 0.81  | 75.13 ± 0.38  |
|             | 75.17 ± 0.59  | 69.42 ± 0.33  |
|             | 65.09 ± 0.54  | 62.19 ± 0.34  |

### Table II: Comparison results of ANGPN vs. NGPN on four benchmark datasets.

| Dataset     | method       | 10% 20% 30%   |
|-------------|--------------|---------------|
| SVHN        | NGPN         | 75.73 ± 0.67  |
|             |              | 77.72 ± 0.64  |
|             |              | 78.49 ± 0.59  |
| CIFAR       | NGPN         | 76.65 ± 0.83  |
|             |              | 78.69 ± 0.53  |
|             |              | 79.62 ± 0.55  |
| 20News      | NGPN         | 61.41 ± 0.77  |
|             |              | 64.51 ± 0.74  |
|             |              | 64.65 ± 0.58  |
| CoraML      | NGPN         | 62.42 ± 1.04  |
|             |              | 65.43 ± 0.66  |
|             |              | 65.90 ± 0.49  |
|             | ANGPN        | 90.01 ± 0.35  |
|             |              | 91.64 ± 0.54  |
|             |              | 91.97 ± 0.71  |
|             | ANGPN        | 90.80 ± 0.34  |
|             |              | 92.35 ± 0.37  |
|             |              | 92.83 ± 0.48  |
|             | ANGPN        | 63.42 ± 1.75  |
|             |              | 66.40 ± 0.71  |
|             |              | 67.88 ± 1.92  |
|             | ANGPN        | 65.53 ± 1.25  |
|             |              | 69.22 ± 1.30  |
|             |              | 69.55 ± 1.44  |

### References

[1] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” in *Advances in Neural Information Processing Systems*, 2016, pp. 3844–3852.

[2] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, “Geometric deep learning on graphs and manifolds using mixture model cnns,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5423–5434.

[3] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[4] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.

[5] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral networks and locally connected networks on graphs,” in *International Conference on Learning Representations*, 2014.

[6] M. Henaff, J. Bruna, and Y. LeCun, “Deep convolutional networks on graph-structured data,” *arXiv preprint arXiv:1506.05163*, 2015.

[7] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.

[8] J. Klicpera, A. Bojchevski, and S. Günnemann, “Predict then propagate: Graph neural networks meet personalized pagerank,” in *ICLR*, 2019.

[9] P. Veličković, W. Fedus, W. L. Hamilton, P. Lio, Y. Bengio, and R. D. Hjelm, “Deep Graph Infomax,” in *International Conference on Learning Representations*, 2019.

[10] B. Jiang, D. Lin, J. Tang, and B. Luo, “Data representation and learning with graph diffusion-embedding networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10414–10423.

[11] L. Rauoy, W. Sheng, Z. Feiyun, and J. Junzhou, “Adaptive graph convolutional neural networks,” in *AAAI Conference on Artificial Intelligence*, 2018, pp. 3546–3553.

[12] B. Jiang, Z. Zhang, J. Tang, and B. Luo, “Graph optimized convolutional networks,” *arXiv:1904.11883*, 2019.

[13] B. Jiang, B. Wang, J. Tang, and B. Luo, “Graph mask convolutional network,” *arXiv preprint arXiv:1910.01735v2*, 2019.

[14] F. Nie, X. Wang, and H. Huang, “Clustering and projected clustering with adaptive neighbors,” in *Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 2014.

[15] F. Nie, W. Zhu, and X. Li, “Unsupervised feature selection with structured graph optimization,” in *AAAI conference on artificial intelligence*, 2016.

[16] F. Wang and C. Zhang, “Label propagation through linear neighborhoods,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 55–67, 2008.

[17] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” in *NIPS workshop on deep learning and unsupervised feature learning*, 2011.

[18] K. Lang, “Newsweeder: Learning to filter netnews,” in *Proceedings of the Twelfth International Conference on Machine Learning*, 1995, pp. 331–339.

[19] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Citeseer, Tech. Rep., 2009.

[20] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, “Collective classification in network data,” *AI magazine*, vol. 29, no. 3, p. 93, 2008.

[21] M. Belkin, P. Niyogi, and V. Sindhwani, “Manifold regularization: A geometric framework for learning from labeled and unlabeled examples,” *Journal of machine learning research*, vol. 7, no. Nov, pp. 2399–2434, 2006.

[22] X. Zhu, Z. Ghahramani, and J. D. Lafferty, “Semi-supervised learning using gaussian fields and harmonic functions,” in *Proceedings of the 20th International conference on Machine learning* (ICML-03), 2003, pp. 912–919.

[23] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 701–710.

[24] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *International Conference on Learning Representations*, 2015.

[25] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *International conference on artificial intelligence and statistics*, 2010, pp. 249–256.