EXPLORING STRUCTURE-ADAPTIVE GRAPH LEARNING FOR ROBUST SEMI-SUPERVISED CLASSIFICATION

Xiang Gao, Wei Hu, and Zongming Guo*

{gyshgx868, forhuwei, guozongming}@pku.edu.cn
Wangxuan Institute of Computer Technology, Peking University, Beijing

ABSTRACT

Graph Convolutional Neural Networks (GCNNs) are generalizations of CNNs to graph-structured data, in which convolution is guided by the graph topology. In many cases where graphs are unavailable, existing methods manually construct graphs or learn task-driven adaptive graphs. In this paper, we propose Graph Learning Neural Networks (GLNNs), which exploit the optimization of graphs (the adjacency matrix in particular) from both data and tasks. Leveraging on spectral graph theory, we propose the objective of graph learning from a sparsity constraint, properties of a valid adjacency matrix as well as a graph Laplacian regularizer via maximum a posteriori estimation. The optimization objective is then integrated into the loss function of the GCNN, which adapts the graph topology to not only labels of a specific task but also the input data. Experimental results show that our proposed GLNN significantly outperforms state-of-the-art approaches over widely adopted social network datasets and citation network datasets for semi-supervised classification.

Index Terms— Graph learning, graph convolutional neural networks, structure-adaptive, robustness

1. INTRODUCTION

Graphs are natural and efficient representation for non-Euclidean data, such as social networks, citation networks, brain neural networks and 3D geometric data. Recently, Graph Convolutional Neural Networks (GCNNs) have been proposed to generalize CNNs to non-Euclidean domain [1], which has shown its efficiency in various applications such as semi-supervised classification [2, 3, 4, 5, 1, 6]. Graphs are fed into the network for basic operations such as graph convolution, and thus play a vital role in feature extraction. In cases where graphs are not readily available, many previous works manually construct graphs from the input data, which tend to be sub-optimal.

Existing graph learning methods can be divided into three main categories: statistical methods, graph spectral methods, and deep learning methods. Statistical methods estimate the inverse covariance or precision matrix given sufficient empirical data, assuming certain prior topological information (*e.g.*, sparsity) [7, 8]. Graph spectral learning methods often optimize the Laplacian matrix [9] from observed graph signals, which are assumed to reside in a low-dimensional subspace spanned by low frequency components [10, 11, 12].

Recently, there have been few attempts at graph learning in GCNNs. Li et al. [13] learn an adaptive graph via distance metric learning for graph classification, which is *task-driven* during the model optimization. Jiang et al. [14] propose a graph learning convolutional network, which learns a non-negative edge weight function that represents pairwise similarities within graph data for semi-supervised learning. However, only edge weights are learned while edge connectivities resort to the structure of the given ground truth graph, which may not capture potential connections. Li et al. [15] propose a spatio-temporal graph learning scheme for skeleton-based action recognition, which adaptively learns intrinsic high-order connectivities for skeleton joints. Shi et al. [16] come up with a two-stream adaptive graph convolutional network for skeleton-based action recognition, where a shared graph for all instances and an individual graph for each instance are learned in an end-to-end manner. Nevertheless, the optimization of the above network models either are task-driven or focus on edge weight learning based on an available graph structure.

To this end, we propose a structure-adaptive Graph Learning Neural Network (GLNN) for *robust* graph representation learning especially at low label rates. The key idea is to pose graph learning as an optimization problem of an adjacency matrix[1] from *both data and labels* based on a structure-adaptive regularization—*Graph Laplacian Regularizer* (GLR) [17]. GLR measures the smoothness of data with respect to the underlying graph represented by the graph Laplacian matrix[2], which essentially enforces the graph to capture pairwise similarities within the data. This structure-adaptive regularization enhances the robustness of the net-

* Corresponding author: Wei Hu (forhuwei@pku.edu.cn). This work was supported by National Natural Science Foundation of China [61972009] and Beijing Natural Science Foundation [4194080].

[1] The adjacency matrix encodes connectivities of a graph.

[2] In spectral graph theory, a graph Laplacian matrix is an algebraic representation of connectivities and node degrees of the corresponding graph, which can be computed from the adjacency matrix and will be introduced in Section 2.

978-1-7281-1331-9/20/$31.00

work model especially in case of low label rates by enforcing smoothness over labels of similar samples. Also, we learn both graph connectivities and edge weights encoded in the adjacency matrix.

Specifically, we first represent features of network data (*e.g.*, social/citation networks) as signals on the graph, and pose a Maximum a Posteriori (MAP) estimation on the underlying adjacency matrix. In the MAP estimation, we propose a likelihood function based on GLR. Also, we propose a prior distribution from the sparsity constraint and properties of a valid adjacency matrix, considering a symmetric, normalized and loopless graph. The MAP estimation then leads to an optimization problem for the underlying adjacency matrix. Secondly, we integrate the optimization objective into the cross-entropy loss function of a GCNN, thus optimizing the network model in both **structure-adaptive** and **task-driven** fashion. Finally, we design a framework of the proposed GLNN, which mainly consists of a Graph Learning Layer and Graph Convolution Layers. We evaluate GLNN on semi-supervised classification tasks for social networks and citation networks, and set the new state-of-the-art. Further, we demonstrate the robustness of GLNN at low label rates, which is crucial to practical applications.

In summary, our main contributions are as follows:

- We propose graph learning regularized by GLR for robust semi-supervised node classification especially at low label rates, which jointly learns graph connectivities and edge weights in both structure-adaptive and task-driven manners.
- We present a GLNN framework that integrates graph learning with graph convolution, which optimizes the network model by introducing GLR, properties of valid adjacency matrices and sparsity constraints into the loss function.
- Extensive experiments demonstrate the superiority and robustness of our method compared to state-of-the-art approaches on widely used citation network datasets and social network datasets.

2. BACKGROUND IN SPECTRAL GRAPH THEORY

2.1. Graph Laplacian

We consider an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$ composed of a node set $\mathcal{V}$ of cardinality $|\mathcal{V}| = N$, an edge set $\mathcal{E}$ connecting nodes, and a weighted *adjacency matrix* $\mathbf{A}$. $\mathbf{A}$ is a real symmetric $N \times N$ matrix, where $a_{i,j}$ is the weight assigned to the edge (i, j) connecting nodes i and j.

The *graph Laplacian* is defined from the adjacency matrix. Among different variants of Laplacian matrices, the commonly used *combinatorial graph Laplacian* [18] is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{1}$$

where $\mathbf{D}$ is the *degree matrix*—a diagonal matrix where $d_{i,i} = \sum_{j=1}^{N} a_{i,j}$.

To ensure the numerical stability in a deep neural network model, we employ the symmetric *normalized* Laplacian [18], which is defined as

$$\mathcal{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}, \tag{2}$$

where $\mathbf{I}$ is an identity matrix.

2.2. Graph Laplacian Regularizer

Graph signal refers to data that reside on the nodes of a graph. In our context, we treat each person in a social network or each paper in a citation network as a node in a graph, and the relationship between each pair of nodes as the edge. Then we define the graph signal as the social feature or paper information of each node.

A graph signal $\mathbf{x} \in \mathbb{R}^N$ defined on a graph $\mathcal{G}$ is *smooth* with respect to $\mathcal{G}$ [17] if

$$\mathbf{x}^\top \mathbf{L}\mathbf{x} = \sum_{i=1}^{N}\sum_{j=1}^{N} a_{i,j}(x_i - x_j)^2 < \epsilon, \tag{3}$$

where ϵ is a small positive scalar. To satisfy (3), disconnected or weakly connected node pair x_i and x_j with significantly different values must correspond to a small edge weight $a_{i,j}$; for strongly connected x_i and x_j with similar values, $a_{i,j}$ can be large. Hence, (3) forces $\mathcal{G}$ to capture pairwise similarities in $\mathbf{x}$, and is commonly called the *graph Laplacian Regularizer* (GLR) [17].

3. THE PROPOSED GRAPH LEARNING

In this section, we elaborate on the formulation of structure-adaptive graph learning, and then describe the corresponding network model optimization.

3.1. Problem Formulation of Graph Learning

We propose to exploit the learning of the direct representation of the underlying graph, *i.e.*, the adjacency matrix. In particular, we pose a MAP estimation of the adjacency matrix, which leads to graph learning from the GLR, sparsity constraint and properties of a valid adjacency matrix.

3.1.1. MAP Estimation of Graph Adjacency Matrices

We first formulate a MAP estimation problem for the optimal graph adjacency matrix $\hat{\mathbf{A}}$: given the observed graph signal $\mathbf{x}$, find the most probable graph $\hat{\mathbf{A}}$,

$$\tilde{\mathbf{A}}_{\text{MAP}}(\mathbf{x}) = \arg\max_{\hat{\mathbf{A}}} \; f(\mathbf{x} \mid \hat{\mathbf{A}}) g(\hat{\mathbf{A}}), \tag{4}$$

where $f(\mathbf{x} \mid \hat{\mathbf{A}})$ is the likelihood function, and $g(\hat{\mathbf{A}})$ is the prior probability distribution of $\hat{\mathbf{A}}$. The likelihood function $f(\mathbf{x} \mid \hat{\mathbf{A}})$ is the probability of obtaining the observed graph signal $\mathbf{x}$ given the graph $\hat{\mathbf{A}}$, while the prior probability distribution $g(\hat{\mathbf{A}})$ provides the prior knowledge of $\hat{\mathbf{A}}$. The likelihood and prior functions are discussed in order as follows.

3.1.2. Proposed Prior Probability Distribution

The prior probability distribution $g(\hat{\mathbf{A}})$ provides the prior knowledge of $\hat{\mathbf{A}}$. We propose the prior knowledge of an adjacency matrix from two aspects: 1) the sparsity constraint

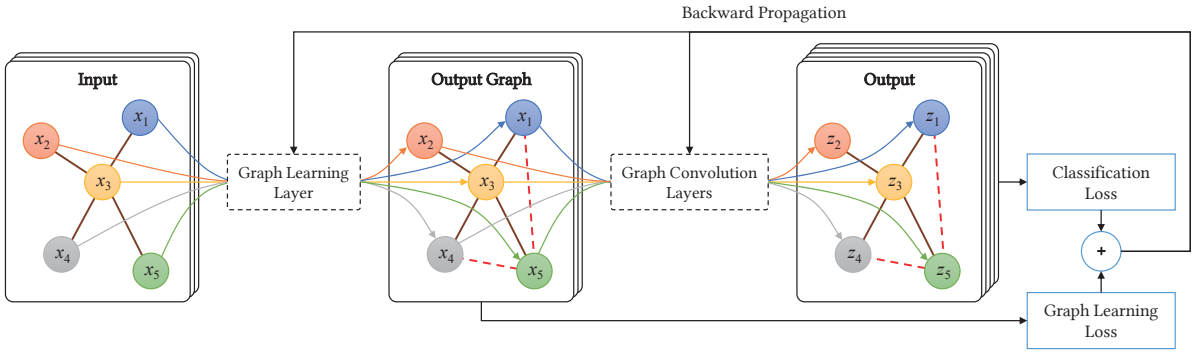


Fig. 1. The architecture of the proposed GLNN for semi-supervised node classification.

$g_s(\hat{\mathbf{A}})$; 2) the property constraint of a valid adjacency matrix $g_p(\hat{\mathbf{A}})$. Since the two prior constraints are independent, we have

$$g(\hat{\mathbf{A}}) = g_s(\hat{\mathbf{A}})g_p(\hat{\mathbf{A}}). \tag{5}$$

We elaborate on the sparsity constraint and property constraint separately as follows.

Sparsity Constraint. We pose a sparsity constraint on $\hat{\mathbf{A}}$. This is based on the observation that the similarity relationship in real-world data is often sparse. Also, this constraint avoids data overfitting by capturing prominent similarities. Mathematically, we define the sparsity prior as

$$g_s(\hat{\mathbf{A}}) = \exp\left(-\lambda_1 \|\hat{\mathbf{A}}\|_1\right), \tag{6}$$

where $\|\cdot\|_1$ denotes the l_1-norm of a matrix, and λ_1 is a weighting parameter.

Property Constraint. The properties of a valid normalized adjacency matrix include:

- Symmetry. We consider undirected graphs, corresponding to a symmetric adjacency matrix $\hat{\mathbf{A}}$, *i.e.*,
$$\hat{\mathbf{A}}^\top = \hat{\mathbf{A}}. \tag{7}$$
- Normalized. We enforce normalized adjacency matrices in order to avoid numerical instabilities and exploding or vanishing gradients when employed in a deep neural network architecture. The mathematical description is
$$\hat{\mathbf{A}}\mathbf{1} = \mathbf{1}, \tag{8}$$
where $\mathbf{1}$ denotes the vector with all elements equal to one.
- Loopless. Since it is often unnecessary to link a node with itself such as in a citation network or social network, we consider graphs without self loop, *i.e.*,
$$\text{tr}(\hat{\mathbf{A}}) = 0, \tag{9}$$
where $\text{tr}(\cdot)$ denotes the trace of a matrix.

Combining the above properties, we propose the following prior probabilistic distribution of $\hat{\mathbf{A}}$:

$$\begin{aligned} g_p(\hat{\mathbf{A}}) = \exp\big(&-\lambda_2 \|\hat{\mathbf{A}}^\top - \hat{\mathbf{A}}\|_F^2 \\ &-\lambda_3 \|\hat{\mathbf{A}}\mathbf{1} - \mathbf{1}\|_F^2 - \lambda_4 |\text{tr}(\hat{\mathbf{A}})|^2\big), \end{aligned} \tag{10}$$

where $\|\cdot\|_F$ is the Frobenius norm of a matrix, and λ_2, λ_3, and λ_4 are all weighting parameters.

3.1.3. *Proposed Likelihood Function*

Since GLR enforces the graph signal $\mathbf{x}$ to adapt to the topology of the graph described by $\hat{\mathbf{A}}$, we propose the likelihood function as

$$\begin{aligned} f(\mathbf{x} \mid \hat{\mathbf{A}}) &= \exp\left(-\lambda_0 \mathbf{x}^\top \mathbf{L}\mathbf{x}\right) \\ &= \exp\left(-\lambda_0 \mathbf{x}^\top (\mathbf{I} - \hat{\mathbf{A}})\mathbf{x}\right), \end{aligned} \tag{11}$$

where λ_0 is a parameter. Since we assume $\hat{\mathbf{A}}$ is normalized, the degree matrix is an identity matrix as in (11). This provides a structural assumption of data encoded in the adjacency matrix, which includes both graph connectivities and edge weights.

3.1.4. *Final MAP Estimation*

Having discussed the proposed likelihood function and prior, we arrive at our final MAP estimation of the graph adjacency matrix. Combining (4), (5), (6), (10) and (11), we have

$$\begin{aligned} \max_{\hat{\mathbf{A}}} \ & \exp\left(-\lambda_0 \mathbf{x}^\top (\mathbf{I} - \hat{\mathbf{A}})\mathbf{x}\right) \cdot \exp\left(-\lambda_1 \|\hat{\mathbf{A}}\|_1\right) \\ & \cdot \exp\big(-\lambda_2 \|\hat{\mathbf{A}}^\top - \hat{\mathbf{A}}\|_F^2 - \lambda_3 \|\hat{\mathbf{A}}\mathbf{1} - \mathbf{1}\|_F^2 \\ & -\lambda_4 |\text{tr}(\hat{\mathbf{A}})|^2\big). \end{aligned} \tag{12}$$

Taking the logarithm of (12) and multiplying by -1, we have

$$\begin{aligned} \min_{\hat{\mathbf{A}}} \ & \lambda_0 \mathbf{x}^\top (\mathbf{I} - \hat{\mathbf{A}})\mathbf{x} + \lambda_1 \|\hat{\mathbf{A}}\|_1 + \lambda_2 \|\hat{\mathbf{A}}^\top - \hat{\mathbf{A}}\|_F^2 \\ & + \lambda_3 \|\hat{\mathbf{A}}\mathbf{1} - \mathbf{1}\|_F^2 + \lambda_4 |\text{tr}(\hat{\mathbf{A}})|^2, \end{aligned} \tag{13}$$

where the first term is GLR, the second term is the sparsity constraint of the adjacency matrix, and the rest are the property constraints of a valid adjacency matrix. Hence, *the proposed formulation is to learn such a valid and sparse adjacency matrix $\hat{\mathbf{A}}$ that the graph signal $\mathbf{x}$ is adaptive to the learned graph.*

3.2. Network Model Optimization

We propose to integrate the optimization objective in (13) into the loss function of the GCN, which seamlessly optimizes the network model with graph learning. Accordingly, the proposed overall loss in graph learning $\mathcal{L}_{\text{GL}}$ includes three components: the loss in GLR $\mathcal{L}_{\text{GLR}}$, the loss in sparsity $\mathcal{L}_{\text{sparsity}}$,

and the loss in property constraints $\mathcal{L}_{\text{properties}}$:

$$\mathcal{L}_{\text{GL}} = \mathcal{L}_{\text{GLR}} + \mathcal{L}_{\text{sparsity}} + \mathcal{L}_{\text{properties}}. \tag{14}$$

In accordance with (13), $\mathcal{L}_{\text{GLR}}$ is defined as

$$\mathcal{L}_{\text{GLR}} = \lambda_0 \|\mathbf{x}^\top (\mathbf{I} - \hat{\mathbf{A}}_{\text{out}})\mathbf{x}\|_2^2. \tag{15}$$

where $\hat{\mathbf{A}}_{\text{out}}$ is the final output adjacency matrix at each epoch. $\mathcal{L}_{\text{GLR}}$ is in a quadratic form, which is differentiable.

$\mathcal{L}_{\text{sparsity}}$ is accordingly defined as

$$\mathcal{L}_{\text{sparsity}} = \lambda_1 \|\hat{\mathbf{A}}_{\text{out}}\|_1. \tag{16}$$

The l_1-norm is non-differentiable since it is not continuous everywhere. In the backward propagation algorithm of machine learning, the gradient of the l_1 regularization is solved using the sub-gradient.

Finally, we define $\mathcal{L}_{\text{properties}}$ as

$$\begin{aligned}\mathcal{L}_{\text{properties}} =& \lambda_2 \|\hat{\mathbf{A}}_{\text{out}}^\top - \hat{\mathbf{A}}_{\text{out}}\|_2^2 \\ &+ \lambda_3 \|\hat{\mathbf{A}}_{\text{out}}\mathbf{1} - \mathbf{1}\|_2^2 + \lambda_4 |\text{tr}(\hat{\mathbf{A}}_{\text{out}})|^2.\end{aligned} \tag{17}$$

This function is differentiable because all the three terms are l_2 norms.

4. THE PROPOSED GLNN

Having discussed the proposed graph learning, we elaborate on its integration with GCN, which leads to the proposed GLNN architecture. As shown in Fig. 1, given graph-structured networks as the input, GLNN consists of two major procedures: graph learning and graph convolution. The graph learning layer aims to produce an optimal adjacency matrix for the subsequent graph convolution. The graph convolution has two layers, with the same parameter settings as in [2].

4.1. Graph Learning Layer

The graph learning layer aims to construct an optimal graph structure from the input graph signals. In this layer, we first randomly initialize the adjacency matrix before training the network, and then train the graph using gradient descent from the backward propagation of the proposed loss function introduced in (14).

In order to further simplify the implementation, we perform the following symmetrization on the adjacency matrix $\hat{\mathbf{A}}$ before the output of the graph learning layer, which aims to ensure the symmetry property:

$$\hat{\mathbf{A}}_{\text{out}} = \frac{1}{2}\left(\hat{\mathbf{A}}^\top + \hat{\mathbf{A}}\right), \tag{18}$$

where $\hat{\mathbf{A}}_{\text{out}} \in \mathbb{R}^{N\times N}$ is the final output adjacency matrix of the graph learning layer at each epoch. It is a real symmetric matrix by definition. Thus, we can remove the symmetric term in (17).

The graph obtained from the graph learning layer is then fed into the GCN architecture for the subsequent semi-supervised node classification task.

4.2. Graph Convolution and Feature Transfer

We consider a two-layer GCN with the learned adjacency matrix $\hat{\mathbf{A}}_{\text{out}}$. The forward model takes the form [2]:

$$\begin{aligned}\mathbf{Z} &= f(\mathbf{X}, \hat{\mathbf{A}}_{\text{out}}) \\ &= \text{softmax}\left(\hat{\mathbf{A}}_{\text{out}}\, \text{ReLU}\left(\hat{\mathbf{A}}_{\text{out}}\mathbf{X}\mathbf{W}^{(0)}\right)\mathbf{W}^{(1)}\right),\end{aligned} \tag{19}$$

where $\mathbf{W}^{(0)} \in \mathbb{R}^{C\times H}$ is an input-to-hidden weight matrix for a hidden graph convolutional layer with H output feature channels, and $\mathbf{W}^{(1)} \in \mathbb{R}^{H\times F}$ is a hidden-to-output weight matrix for an output layer with F classes. The softmax$(\cdot)$ and ReLU$(\cdot)$ are two activation functions.

4.3. The Overall Loss Function

In semi-supervised node classification tasks, we need to evaluate the cross-entropy loss (the Classification Loss in Fig. 1) over the labeled data:

$$\mathcal{L}_{\text{GCN}} = -\sum_{l\in\mathcal{Y}_L}\sum_{f=1}^{F} \mathbf{Y}_{lf} \ln \mathbf{Z}_{lf}, \tag{20}$$

where $\mathcal{Y}_L$ is a set of labeled nodes, $\mathbf{Y}_{l\cdot}$ denotes the label for the l^{th} labeled node, and $\mathbf{Z}_{l\cdot}$ is the predicted label for the l^{th} node.

Further, in cases where the ground truth graph of training samples is available, we add the following loss function to ensure the learned graph is closer to the ground truth:

$$\mathcal{L}_{\text{gt}} = \|\hat{\mathbf{A}}_{\text{out}} - \mathbf{A}_{\text{gt}}\|_2^2, \tag{21}$$

where $\mathbf{A}_{\text{gt}}$ is a ground truth adjacency matrix.

Hence, the overall loss function of our proposed GLNN framework is

$$\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{GCN}} + \mathcal{L}_{\text{GL}} + \alpha\mathcal{L}_{\text{gt}}. \tag{22}$$

where α is a weighting parameter.

5. EXPERIMENTAL RESULTS

5.1. Datasets

We consider two social networks: TerroristRel and Terror Attack [19], and three citation networks: Citeseer, Cora, and Pubmed [20]. The five datasets are available at https://linqs.soe.ucsc.edu/data. We represent the relationship between terrorists and citation links as undirected edges, and construct a binary and symmetric matrix as introduced in [2] as the ground truth adjacency matrix.

We employ 160 nodes as labeled data to train our model and leave 150 nodes for testing in the TerroristRel dataset, while in the Terror Attack dataset we deploy 120 nodes for training and leave 200 nodes for testing. In the Pubmed dataset, we sample 3,000 nodes to evaluate our model, among which we select 60 nodes as labeled data to train each model and test the performance on 1,000 nodes. For the Citeseer and Cora dataset, we follow the experimental setup in [21].

5.2. Implementation Details

We implemented the proposed model with the TensorFlow framework, and empolyed Adam to train the entire network

Table 1. Results of node classification in terms of accuracy (%). The asterisk indicates a sampled dataset.

Method	Terrorists Rel	Terror Attack	Citeseer	Cora	Pubmed*
Planetoid [21]	-	-	64.7	75.7	74.5
ChebNet [3]	60.1	66.5	69.6	81.2	71.6
MoNet [22]	59.2	62.4	-	81.7	73.5
LoNGAE [23]	63.5	66.0	71.8	79.0	74.7
GAT [24]	63.5	65.5	71.0	82.3	72.3
DGI [25]	52.7	61.5	71.8	82.3	74.6
GWNN [26]	65.5	65.0	71.7	82.8	72.4
GCN [2]	62.2	68.0	70.3	81.5	73.8
GLNN	**70.9**	**76.5**	**72.4**	**83.4**	**76.7**

with the learning rate 0.01. Before the first epoch of the training step in the graph learning layer, we generate a random matrix that follows a uniform distribution, which serves as the initial graph representation. This matrix is then optimized in the subsequent training epochs. In the graph learning loss function, we set $\lambda_1 = 0.1$, $\lambda_3 = 0.1$, and $\lambda_4 = 0.001$. Note that λ_2 is set to 0 with the symmetrization operation introduced in (18). In the overall loss function (22), we set $\alpha = 10$. The settings of λ_0 are adaptive to each dataset, as will be discussed in Section 5.5. In the graph convolutional layers, we follow the same experimental settings in [2], *i.e.*, two graph convolutional layers with 16 hidden units.

5.3. Node Classification Results

Table 1 lists the comparison results on the five datasets. Reported numbers denote the classification accuracy in percentage. Note that the underlying graphs of TerroristsRel and Terror Attack datasets are disconnected, so the Planetoid [21] cannot be executed properly.

We outperform all the competing methods significantly. In particular, our GLNN outperforms the baseline method GCN [2] on all the datasets, thus validating the effectiveness of the proposed graph learning for semi-supervised classification. Further, GLNN outperforms the recently proposed network GWNN [26], which indicates the benefit of GLNN on graph representation learning.

Table 2. Results of node classification in terms of accuracy with different λ_0.

Dataset	Values of λ_0					
	1.0	10^{-1}	10^{-2}	10^{-3}	10^{-4}	0.0
TerroristsRel	52.0	**70.9**	**70.9**	63.5	63.5	62.8
Terror Attack	**76.5**	**76.5**	67.5	51.5	57.5	57.0
Citeseer	**72.4**	71.2	72.2	70.4	70.4	69.8
Cora	76.7	80.0	**83.4**	82.0	82.1	82.1
Pubmed*	**76.7**	74.8	73.0	73.0	72.9	72.9

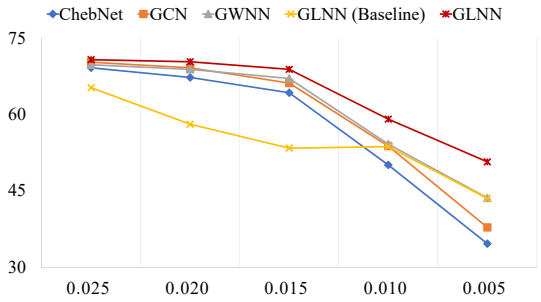
5.4. Robustness Test

We test the robustness of our model on citation networks at low label rates. We adopt five different label rates in $\{0.005, 0.010, 0.015, 0.020, 0.025\}$ to train our model and compare with three representative methods: ChebNet [3], GCN [2] and GWNN [26]. Fig. 2 presents the classification accuracy at the five different label rates on the Citeseer and Cora datasets respectively. While the performance of the other methods drops quickly with decreasing label rate, the proposed GLNN is much more robust, with the classification accuracy of 51.8% and 58.0% on the Citeseer and Cora datasets even when the label rate is lower than 0.01.

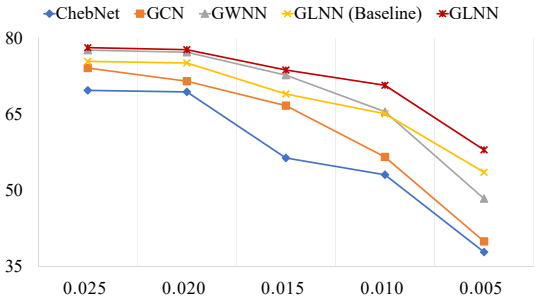
Further, to validate the necessity and effectiveness of our proposed GLR loss, we also compare with a **Baseline** scheme of our model, where the GLR loss term $\mathcal{L}_{\text{GLR}}$ is removed from (14). As shown in Fig. 2, we outperform the **Baseline** scheme by a large margin, which validates the superiority of our proposed structure-adaptive regularization—GLR loss function.

5.5. GLR Analysis

We evaluate the contribution of GLR via different assignments of the GLR weighting parameter λ_0. Table 2 lists the classification results of five datasets with $\lambda_0 \in \{1.0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 0.0\}$. We see that when λ_0 is set to a small value, GLNN can produce reasonable results. The performance keeps improving with increasing value of λ_0 in general, finally achieving state-of-the-art performance. This shows that the GLR loss term with higher weights makes significant contribution to classification results, which demonstrates the superiority of GLR-based graph learning.



(a) Citeseer dataset.



(b) Cora dataset.

Fig. 2. Comparison of classification accuracy from competing methods at different label rates.

6. CONCLUSION

In this paper, we propose a novel Graph Learning Neural Network (GLNN), aiming to learn an optimal graph in both structure-adaptive and task-driven manners for robust semi-supervised classification. Our GLNN model exploits structure-adaptive graph learning based on graph Laplacian

regularizer (GLR), which enforces the graph to characterize pairwise similarities within data. We also pose the sparsity constraint and properties of a valid adjacency matrix, which formulates the graph learning loss function along with GLR in a GCNN. Experimental results on widely adopted citation network datasets and social network datasets validate the superiority and robustness of the proposed GLNN especially at low label rates.

7. REFERENCES

[1] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun, "Spectral networks and locally connected networks on graphs," *Computer Science*, 2013.

[2] Thomas N Kipf and Max Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[3] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 3844–3852.

[4] Marco Gori, Gabriele Monfardini, and Franco Scarselli, "A new model for learning in graph domains," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2005, vol. 2, pp. 729–734.

[5] 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 (NIPS)*, 2015, pp. 2224–2232.

[6] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov, "Learning convolutional neural networks for graphs," in *International Conference on Machine Learning (ICML)*, 2016, pp. 2014–2023.

[7] Arthur P Dempster, "Covariance selection," *Biometrics*, pp. 157–175, 1972.

[8] Jerome Friedman, Trevor Hastie, and Robert Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[9] F. K. Chung, "Spectral graph theory," *American Mathematical Society*, 1997.

[10] Xiaowen Dong, Dorina Thanou, Pascal Frossard, and Pierre Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing (TSP)*, vol. 64, no. 23, pp. 6160–6173, 2016.

[11] Hilmi E Egilmez, Eduardo Pavez, and Antonio Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, 2017.

[12] Xiang Gao, Wei Hu, Jiaxiang Tang, Jiaying Liu, and Zongming Guo, "Optimized skeleton-based action recognition via sparsified graph regression," in *ACM International Conference on Multimedia (ACM MM)*, 2019, pp. 601–610.

[13] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang, "Adaptive graph convolutional neural networks," in *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[14] Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo, "Semi-supervised learning with graph learning-convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11313–11320.

[15] Bin Li, Xi Li, Zhongfei Zhang, and Fei Wu, "Spatio-temporal graph routing for skeleton-based action recognition," in *Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[16] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12026–12035.

[17] David I Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, pp. 83–98, 2013.

[18] Fan RK Chung, "Spectral graph theory," in *Conference Board of the Mathematical Sciences*. 1997, number 92, American Mathematical Society.

[19] Bin Zhao, Prithviraj Sen, and Lise Getoor, "Entity and relationship labeling in affiliation networks," in *International Conference on Machine Learning Workshop on Statistical Network Analysis*, 2006.

[20] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.

[21] Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," *arXiv preprint arXiv:1603.08861*, 2016.

[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 (CVPR)*, 2017, pp. 5115–5124.

[23] Phi Vu Tran, "Learning to make predictions on graphs with autoencoders," in *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2018, pp. 237–245.

[24] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio, "Graph attention networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[25] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm, "Deep graph infomax," in *International Conference on Learning Representations (ICLR)*, 2019.

[26] Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng, "Graph wavelet neural network," in *International Conference on Learning Representations (ICLR)*, 2019.