SLGAT: Soft Labels Guided Graph Attention Networks

Yubin Wang[1,2], Zhenyu Zhang[1,2], Tingwen Liu[1,2(✉)], and Li Guo[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{wangyubin,zhangzhenyu1996,liutingwen,guoli}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. Graph convolutional neural networks have been widely studied for semi-supervised classification on graph-structured data in recent years. They usually learn node representations by transforming, propagating, aggregating node features and minimizing the prediction loss on labeled nodes. However, the pseudo labels generated on unlabeled nodes are usually overlooked during the learning process. In this paper, we propose a soft labels guided graph attention network (SLGAT) to improve the performance of node representation learning by leveraging generated pseudo labels. Unlike the prior graph attention networks, our SLGAT uses soft labels as guidance to learn different weights for neighboring nodes, which allows SLGAT to pay more attention to the features closely related to the central node labels during the feature aggregation process. We further propose a self-training based optimization method to train SLGAT on both labeled and pseudo labeled nodes. Specifically, we first pre-train SLGAT on labeled nodes and generate pseudo labels for unlabeled nodes. Next, for each iteration, we train SLGAT on the combination of labeled and pseudo labeled nodes, and then generate new pseudo labels for further training. Experimental results on semi-supervised node classification show that SLGAT achieves state-of-the-art performance.

Keywords: Graph neural networks · Attention mechanism · Self-training · Soft labels · Semi-supervised classification

1 Introduction

In recent years, graph convolutional neural networks (GCNs) [26], which can learn from graph-structured data, have attracted much attention. The general approach with GCNs is to learn node representations by passing, transforming, and aggregating node features across the graph. The generated node representations can then be used as input to a prediction layer for various downstream tasks, such as node classification [12], graph classification [30], link prediction [17] and social recommendation [19].

Graph attention networks (GAT) [23], which is one of the most representative GCNs, learns the weights for neighborhood aggregation via self-attention mechanism [22] and achieves promising performance on semi-supervised node

H. W. Lauw et al. (Eds.): PAKDD 2020, LNAI 12084, pp. 512–523, 2020.
https://doi.org/10.1007/978-3-030-47426-3_40

classification problem. The model is expected to learn to pay more attention to the important neighbors. It calculates important scores between connected nodes based solely on the node representations. However, the label information of nodes is usually overlooked. Besides, the cluster assumption [3] for semi-supervised learning states that the decision boundary should lie in regions of low density. It means aggregating the features from the nodes with different classes could reduce the generalization performance of the model. This motivates us to introduce label information to improve the performance of node classification in the following two aspects: (1) We introduce soft labels to guide the feature aggregation for generating discriminative node embeddings for classification. (2) We use SLGAT to predict pseudo labels for unlabeled nodes and further train SLGAT on the composition of labeled and pseudo labeled nodes. In this way, SLGAT can benefit from unlabeled data.

In this paper, we propose soft labels guided attention networks (SLGAT) for semi-supervised node representation learning. The learning process consists of two main steps. First, SLGAT aggregates the features of neighbors using convolutional networks and predicts soft labels for each node based on the learned embeddings. And then, it uses soft labels to guide the feature aggregation via attention mechanism. Unlike the prior graph attention networks, SLGAT allows paying more attention to the features closely related to the central node labels. The weights for neighborhood aggregation learned by a feedforward neural network based on both label information of central nodes and features of neighboring nodes, which can lead to learning more discriminative node representations for classification.

We further propose a self-training based optimization method to improve the generalization performance of SLGAT using unlabeled data. Specifically, we first pre-train SLGAT on labeled nodes using standard cross-entropy loss. Then we generate pseudo labels for unlabeled nodes using SLGAT. Next, for each iteration, we train SLGAT using a combined cross-entropy loss on both labeled nodes and pseudo labeled nodes, and then generate new pseudo labels for further training. In this way, SLGAT can benefit from unlabeled data by minimizing the entropy of predictions on unlabeled nodes.

We conduct extensive experiments on semi-supervised node classification to evaluate our proposed model. And experimental results on several datasets show that SLGAT achieves state-of-the-art performance. The source code of this paper can be obtained from `https://github.com/jadbin/SLGAT`.

2 Related Work

Graph-Based Semi-supervised Learning. A large number of methods for semi-supervised learning using graph representations have been proposed in recent years, most of which can be divided into two categories: graph regularization-based methods and graph embedding-based methods. Different graph regularization-based approaches can have different variants of the regularization term. And graph Laplacian regularizer is most commonly used in

previous studies including label propagation [32], local and global consistency regularization [31], manifold regularization [1] and deep semi-supervised embedding [25]. Recently, graph embedding-based methods inspired by the skip-gram model [14] has attracted much attention. DeepWalk [16] samples node sequences via uniform random walks on the network, and then learns embeddings via the prediction of the local neighborhood of nodes. Afterward, a large number of works including LINE [21] and node2vec [8] extend DeepWalk with more sophisticated random walk schemes. For such embedding based methods, a two-step pipeline including embedding learning and semi-supervised training is required where each step has to be optimized separately. Planetoid [29] alleviates this by incorporating label information into the process of learning embeddings.

Graph Convolutional Neural Networks. Recently, graph convolutional neural networks (GCNs) [26] have been successfully applied in many applications. Existing GCNs are often categorized as spectral methods and non-spectral methods. Spectral methods define graph convolution based on the spectral graph theory. The early studies [2,10] developed convolution operation based graph Fourier transformation. Defferrard et al. [4] used polynomial spectral filters to reduce the computational cost. Kipf & Welling [12] then simplified the previous method by using a linear filter to operate one-hop neighboring nodes. Wu et al. [27] used graph wavelet to implement localized convolution. Xu et al. [27] used a heat kernel to enhance low-frequency filters and enforce smoothness in the signal variation on the graph. Along with spectral graph convolution, define the graph convolution in the spatial domain was also investigated by many researchers. GraphSAGE [9] performs various aggregators such as mean-pooling over a fixed-size neighborhood of each node. Monti et al. [15] provided a unified framework that generalized various GCNs. GraphsGAN [5] generates fake samples and trains generator-classifier networks in the adversarial learning setting. Instead of fixed weight for aggregation, graph attention networks (GAT) [23] adopts attention mechanisms to learn the relative weights between two connected nodes. Wang et al. [24] generalized GAT to learn representations of heterogeneous networks using meta-paths. Shortest Path Graph Attention Network (SPAGAN) to explore high-order path-based attentions.

Our method is based on spatial graph convolution. Unlike the existing graph attention networks, we introduce soft labels to guide the feature aggregation of neighboring nodes. And experiments show that this can further improve the semi-supervised classification performance.

3 Problem Definition

In this paper, we focus on the problem of semi-supervised node classification. Many other applications can be reformulated into this fundamental problem. Let $G = (V, E)$ be a graph, in which V is a set of nodes, E is a set of edges. Each node $u \in V$ has a attribute vector $\mathbf{x}_u$. Given a few labeled nodes $V_L \in V$, where each node $u \in V_L$ is associated with a label $\mathbf{y}_u \in Y$, the goal is to predict the labels for the remaining unlabeled nodes $V_U = V \setminus V_L$.

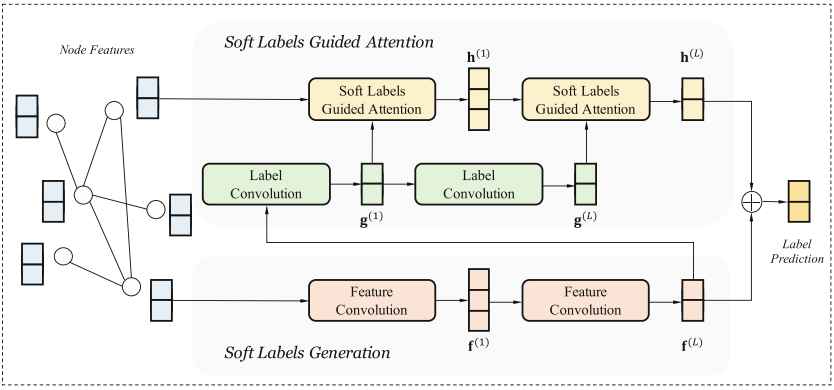

Fig. 1. The overall architecture of SLGAT.

4 Proposed Model: SLGAT

In this section, we will give more details of SLGAT. The overall structure of SLGAT is shown in Fig. 1. The learning process of our method consists of two main steps. We first use a multi-layer graph convolution network to generate soft labels for each node based on nodes features. We then leverage the soft labels to guide the feature aggregation via attention mechanism to learn better representations of nodes. Furthermore, we develop a self-training based optimization method to train SLGAT on the combination of labeled nodes and pseudo labeled nodes. This enforces SLGAT can further benefit from the unlabeled data under the semi-supervised learning setting.

4.1 Soft Labels Generation

In the initial phase, we need to first predict the pseudo labels for each node based on node features $\mathbf{x}$. The pseudo labels can be soft (a continuous distribution) or hard (a one-hot distribution). In practice, we observe that soft labels are usually more stable than hard labels, especially when the model has low prediction accuracy. Since the labels predicted by the model are not absolutely correct, the error from hard labels may propagate to the inference on other labels and hurt the performance. While using soft labels can alleviate this problem.

We use a multi-layer graph convolutional network [12] to aggregate the features of neighboring nodes. The layer-wise propagation rule of feature convolution is as follows:

$$\mathbf{f}^{(l+1)} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}\mathbf{f}^{(l)}W_f^{(l)}\right) \tag{1}$$

Here, $\widetilde{A} = A + I$ is the adjacency matrix with added self-connections. I is the identity matrix, $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$ and $W_f^{(l)} \in \mathbb{R}^{d_f^{(l)} \times d_f^{(l+1)}}$ is a layer-specific trainable transformation matrix. $\sigma(\cdot)$ denotes an activation function such as $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$. $\mathbf{f}^{(l)} \in \mathbb{R}^{|V| \times d_f^{(l)}}$ denotes the hidden representations of nodes

in the l^{th} layer. The representations of nodes $\mathbf{f}^{(l+1)}$ are obtained by aggregating information from the features of their neighborhoods $\mathbf{f}^{(l)}$. Initially, $\mathbf{f}^{(0)} = \mathbf{x}$.

After going through L layers of feature convolution, we predict the soft labels for each node u based on the output embeddings of nodes:

$$\widehat{\mathbf{y}}_u = \text{softmax}\left(\mathbf{f}_u^{(L)}\right) \tag{2}$$

4.2 Soft Labels Guided Attention

Now we will present how to leverage the previous generated soft labels for each node to guide the feature aggregation via attention mechanism. The attention network consists of several stacked layers. In each layer, we first aggregate the label information of neighboring nodes. Then we learn the weights for neighborhood aggregation based on both aggregated label information of central nodes and feature embeddings of neighboring nodes.

We use a label convolution unit to aggregate the label information of neighboring nodes, and the layer-wise propagation rule is as follows:

$$\mathbf{g}^{(l+1)} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}\mathbf{g}^{(l)}W_g^{(l)}\right) \tag{3}$$

where $W_g^{(l)} \in \mathbb{R}^{d_g^{(l)} \times d_g^{(l+1)}}$ is a layer-specific trainable transformation matrix, and $\mathbf{g}^{(l)} \in \mathbb{R}^{|V| \times d_g^{(l)}}$ denotes the hidden representations the label information of nodes. The label information $\mathbf{g}^{(l+1)}$ are obtained by aggregating from the label information $\mathbf{g}^{(l)}$ of neighboring nodes. Initially, $\mathbf{g}^{(0)} = \text{softmax}\left(\mathbf{f}^{(L)}\right)$ according to Eq. 2.

Then we use the aggregated label information to guide the feature aggregation via attention mechanism. Unlike the prior graph attention networks [23,28], we use label information as guidance to learn the weights of neighboring nodes for feature aggregation. We enforce the model to pay more attention to the features closely related to the labels of the central nodes.

A single-layer feedforward neural network is applied to calculate the attention scores between connected nodes based on the central node label information $\mathbf{g}^{(l+1)}$ and the neighboring node features $\mathbf{h}^{(l)}$:

$$a_{ij} = \tanh\left(\mathbf{a}^{(l)^\top}\left[W_t^{(l)}\mathbf{g}_i^{(l+1)} \| W_h^{(l)}\mathbf{h}_j^{(l)}\right]\right) \tag{4}$$

where $\mathbf{a}^{(l)} \in \mathbb{R}^{2d_h^{(l+1)}}$ is a layer-specific attention vector, $W_t^{(l)} \in \mathbb{R}^{d_h^{(l+1)} \times d_g^{(l+1)}}$ and $W_h^{(l)} \in \mathbb{R}^{d_h^{(l+1)} \times d_h^{(l)}}$ are layer-specific trainable transformation matrices, $\mathbf{h}^{(l)} \in \mathbb{R}^{|V| \times d_h^{(l)}}$ denotes the hidden representations of node features. $\cdot^\top$ represents transposition and $||$ is the concatenation operation. Then we obtain the attention weights by normalizing the attention scores with the softmax function:

$$\alpha_{ij} = \frac{\exp\left(a_{ij}\right)}{\sum_{k \in N_i} \exp\left(a_{ik}\right)} \tag{5}$$

where N_i is the neighborhood of node i in the graph. Then, the embedding of node i can be aggregated by the projected features of neighbors with the corresponding coefficients as follows:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j \in N_i} \alpha_{ij} W_h^{(l)} \mathbf{h}_j^{(l)}\right) \tag{6}$$

Finally, we can achieve better predictions for the labels of each node u by replacing the Eq. 2 as follows:

$$\widehat{\mathbf{y}}_u = \text{softmax}\left(\mathbf{f}_u^{(L)} \oplus \mathbf{h}_u^{(L)}\right) \tag{7}$$

where $\oplus$ is the mean-pooling aggregator.

4.3 Self-training Based Optimization

Grandvalet & Bengio [7] argued that adding an extra loss to minimize the entropy of predictions on unlabeled data can further improve the generalization performance for semi-supervised learning. Thus we estimate pseudo labels for unlabeled nodes based on the learned node representations, and develop a self-training based optimization method to train SLGAT on both labeled and pseudo labeled nodes. Int this way, SLGAT can further benefit from the unlabeled data.

For semi-supervised node classification, we can minimize the cross-entropy loss over all labeled nodes between the ground-truth and the prediction:

$$\mathcal{L}_{sup} = -\frac{1}{|V_L|} \sum_{i \in V_L} \sum_{j=1}^{C} \mathbf{y}_{ij} \cdot \log \widehat{\mathbf{y}}_{ij} \tag{8}$$

where C is the number of classes.

To achieve training on the composition of labeled and unlabeled nodes, we first estimate the labels of unlabeled nodes using the learned node embeddings as follows:

$$\widetilde{\mathbf{y}}_u = \text{softmax}\left(\frac{\mathbf{f}_u^{(L)} \oplus \mathbf{h}_u^{(L)}}{\tau}\right) \tag{9}$$

where τ is an annealing parameter. We can set τ to a small value (e.g. 0.1) to further reduce the entropy of pseudo labels. Then the loss for minimizing the entropy of predictions on unlabeled data can be defined as:

$$\mathcal{L}_{unsup} = -\frac{1}{|V_U|} \sum_{i \in V_U} \sum_{j=1}^{C} \widetilde{\mathbf{y}}_{ij} \cdot \log \widehat{\mathbf{y}}_{ij} \tag{10}$$

The joint objective function is defined as a weighted linear combination of the loss on labeled nodes and unlabeled nodes:

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda \mathcal{L}_{unsup} \tag{11}$$

where λ is a weight balance factor.

Algorithm 1: Optimization Algorithm

Input: A graph G, the features of each node $\{\mathbf{x}_u : u \in V\}$ and the labels $\{\mathbf{y}_u : u \in V_L\}$ of some nodes
Output: Labels $\{\widehat{\mathbf{y}}_u : u \in V_U\}$ for unlabeld nodes
Pre-train the model with $\{\mathbf{x}_u : u \in V\}$ and $\{\mathbf{y}_u : u \in V_L\}$ according to Eq. 8.
while not converge **do**
  Generate pseudo labels $\{\widetilde{\mathbf{y}}_u : u \in V_U\}$ on unlabeled nodes based on Eq. 9.
  Predict $\{\widehat{\mathbf{y}}_u : u \in V\}$ based on Eq. 7
  Update parameters with $\{\mathbf{y}_u : u \in V_L\}$, $\{\widetilde{\mathbf{y}}_u : u \in V_U\}$ and $\{\widehat{\mathbf{y}}_u : u \in V\}$ based on Eq. 8, Eq. 10 and Eq. 11.
end
Predict $\{\widehat{\mathbf{y}}_u : u \in V_U\}$ based on Eq. 7

We give a self-training based method to train SLGAT which is listed in Algorithm. 1. The inputs to the algorithm are both labeled and unlabeled nodes. We first use labeled nodes to pre-train the model using cross-entropy loss. Then we use the model to generate pseudo labels on unlabeled nodes. Afterward, we train the model by minimizing the combined cross-entropy loss on both labeled and unlabeled nodes. Finally, we iteratively generate new pseudo labels and further train the model.

5 Experiments

In this section, we evaluate our proposed SLGAT on semi-supervised node classification task using several standard benchmarks. We also conduct an ablation study on SLGAT to investigate the contribution of various components to performance improvements.

5.1 Datasets

We follow existing studies [12,23,29] and use three standard citation network benchmark datasets for evaluation, including Cora, Citeseer and Pubmed. In all these datasets, the nodes represent documents and edges are citation links. Node features correspond to elements of a bag-of-words representation of a document. Class labels correspond to research areas and each node has a class label. In each dataset, 20 nodes from each class are treated as labeled data. The statistics of datasets are summarized in Table 1.

5.2 Baselines

We compare against several traditional graph-based semi-supervised classification methods, including manifold regularization (ManiReg) [1], semi-supervised embedding (SemiEmb) [25], label propagation (LP) [32], graph embeddings (DeepWalk) [16], iterative classification algorithm (ICA) [13] and Planetoid [29].

Table 1. The Statistics of Datasets.

Dataset	Nodes	Edges	Features	Classes	Training	Validation	Test
Cora	2,708	5,429	1,433	7	140	500	1,000
Citeseer	3,327	4,732	3,703	6	120	500	1,000
Pubmed	19,717	44,338	500	3	60	500	1,000

Furthermore, since graph neural networks are proved to be effective for semi-supervised classification, we also compare with several state-of-arts graph neural networks including ChebyNet [4], MoNet [15], graph convolutional networks (GCN) [12], graph attention networks (GAT) [23], graph wavelet neural network (GWNN) [27], shortest path graph attention network (SPAGAN) [28] and graph convolutional networks using heat kernel (GraphHeat) [27].

5.3 Experimental Settings

We train a two-layer SLGAT model for semi-supervised node classification and evaluate the performance using prediction accuracy. The partition of datasets is the same as the previous studies [12,23,29] with an additional validation set of 500 labeled samples to determine hyper-parameters.

Weights are initialized following Glorot and Bengio [6]. We adopt the Adam optimizer [11] for parameter optimization with initial learning rate as 0.05 and weight decay as 0.0005. We set the hidden layer size of features as 32 for Cora and Citeseer and 16 for Pubmed. We set the hidden layer size of soft labels as 16 for Cora and Citeseer and 8 for Pubmed. We apply dropout [20] with $p = 0.5$ to both layers inputs, as well as to the normalized attention coefficients. The proper setting of λ in Eq. 11 affects the semi-supervised classification performance. If λ is too large, it disturbs training for labeled nodes. Whereas if λ is too small, we cannot benefit from unlabeled data. In our experiments, we set $\lambda = 1$. We anticipate the results can be further improved by using sophisticated scheduling strategies such as deterministic annealing [7], and we leave it as future work. Furthermore, inspired by dropout [20], we ignore the loss in Eq. 10 with $p = 0.5$ during training to prevent overfitting on pseudo labeled nodes.

5.4 Semi-supervised Node Classification

We now validate the effectiveness of SLGAT on semi-supervised node classification task. Following the previous studies [12,23,29], we use the classification accuracy metric for quantitative evaluation. Experimental results are summarized in Table 2. We present the mean classification accuracy (with standard deviation) of our method over 100 runs. And we reuse the results already reported in [5,12,23,27,28] for baselines.

We can observe that our SLGAT achieves consistently better performance than all baselines. When directly compared to GAT, SLGAT gains 1.0%, 2.3%

Table 2. Semi-supervised node classification accuracies (%).

Method	Cora	Citeseer	Pubmed
MLP	55.1	46.5	71.4
ManiReg [1]	59.5	60.1	70.7
SemiEmb [25]	59.0	59.6	71.7
LP [32]	68.0	45.3	63.0
DeepWalk [16]	67.2	43.2	65.3
ICA [13]	75.1	69.1	73.9
Planetoid [29]	75.7	64.7	77.2
ChebyNet [4]	81.2	69.8	74.4
GCN [12]	81.5	70.3	79.0
MoNet [15]	81.7 ± 0.5	–	78.8 ± 0.3
GAT [23]	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3
SPAGAN [28]	83.6 ± 0.5	73.0 ± 0.4	79.6 ± 0.4
GraphHeat [27]	83.7	72.5	80.5
SLGAT (ours)	$\mathbf{84.0 \pm 0.6}$	$\mathbf{74.8 \pm 0.6}$	$\mathbf{82.2 \pm 0.5}$

and 3.2% improvements for Cora, Citeseer and Pubmed respectively. The performance gain is from two folds. First, SLGAT uses soft labels to guide the feature aggregation of neighboring nodes. This indeed leads to more discriminative node representations. Second, SLGAT is trained on both labeled and pseudo labeled nodes using our proposed self-training based optimization method. SLGAT benefits from unlabeled data by minimizing the entropy of predictions on unlabeled nodes.

5.5 Classification Results on Random Data Splits

Following Shchur et al. [18], we also further validate the effectiveness and robustness of SLGAT on random data splits. We created 10 random splits of the Cora, Citeseer, Pubmed with the same size of training, validation, test sets as the standard split from Yang et al. [29]. We compare SLGAT with other most related competitive baselines including GCN [12] and GAT [23] on those random data splits.[1] We run each method with 10 random seeds on each data split and report the overall mean accuracy in Table 3. We can observe that SLGAT consistently outperforms GCN and GAT on all datasets. This proves the effectiveness and robustness of SLGAT.

[1] Note that we do not report results of SPAGAN and GraphHeat in this experiment, because we cannot reproduce these two methods without official implementation.

Table 3. Classification results on random data splits (%).

Method	Cora	Citeseer	Pubmed
GCN [12]	79.5	69.1	80.0
GAT [23]	79.7	68.8	79.2
SLGAT (ours)	**82.9**	**72.5**	**80.6**

Table 4. Ablation study results of node classification (%).

Method	Cora	Citeseer	Pubmed
SLGAT	**84.0**	**74.8**	**82.2**
SLGAT without soft labels guided attention	83.7	74.1	**82.2**
SLGAT without self-training	83.6	72.9	81.1
SLGAT without attention & Self-training	82.3	71.7	80.5

5.6 Ablation Study

In this section, we conduct an ablation study to investigate the effectiveness of our proposed soft label guided attention mechanism and the self-training based optimization method for SLGAT. We compare several variants of SLGAT on node classification, and the results are reported in Table 4.

We observe that SLGAT has better performance than the methods without soft labels guided attention in most cases. This demonstrates that using soft labels to guide the neighboring nodes aggregation is effective for generating better node embeddings. Note that attention mechanism seems has little contribution to performance on Pubmed when using self-training. The reason behind such phenomenon is still under investigation, we presume that it is due to the label sparsity of Pubmed.[2] The similar phenomenon is reported in [23] that GAT has little improvement on Pubmed compared to GCN.

We also observe that SLGAT significantly outperforms all the methods without self-training. This indicates that our proposed self-training based optimization method is much effective to improve the generalization performance of the model for semi-supervised classification.

6 Conclusion

In this work, we propose SLGAT for semi-supervised node representation learning. SLGAT uses soft labels to guide the feature aggregation of neighboring nodes for generating discriminative node representations. A self-training based optimization method is proposed to train SLGAT on both labeled data and pseudo labeled data, which is effective to improve the generalization performance of

[2] The label rate of Cora, Citeseer and Pubmed are 0.052, 0.036 and 0.003 respectively.

SLGAT. Experimental results demonstrate that our SLGAT achieves state-of-the-art performance on several semi-supervised node classification benchmarks. One direction of the future work is to make SLGAT going deeper to capture the features of long-range neighbors. This perhaps helps to improve performance on the dataset with sparse labels.

Acknowledgment. This work is supported by the National Key Research and Development Program of China (grant No. 2016YFB0801003) and the Strategic Priority Research Program of Chinese Academy of Sciences (grant No. XDC02040400).

References

1. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J. Mach. Learn. Res. **7**, 2399–2434 (2006)
2. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. In: International Conference on Learning Representations (ICLR) (2013)
3. Chapelle, O., Weston, J., Schölkopf, B.: Cluster kernels for semi-supervised learning. In: Advances in Neural Information Processing Systems, pp. 601–608 (2003)
4. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, pp. 3844–3852 (2016)
5. Ding, M., Tang, J., Zhang, J.: Semi-supervised learning on graphs with generative adversarial nets. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 913–922 (2018)
6. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feed for leveraging graph wavelet transform to address the short-comings of previous spectral graphrd neural networks. In: AISTATS, pp. 249–256 (2010)
7. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: Advances in Neural Information Processing Systems, pp. 529–536 (2005)
8. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
9. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, pp. 1024–1034 (2017)
10. Henaff, M., Bruna, J., LeCun, Y.: Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163 (2015)
11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)
13. Lu, Q., Getoor, L.: Link-based classification. In: Proceedings of the 20th International Conference on Machine Learning, pp. 496–503 (2003)
14. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR Workshop (2013)

15. Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., Bronstein, M.M.: Geometric deep learning on graphs and manifolds using mixture model CNNS. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5425–5434 (2017)
16. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
17. Schütt, K., Kindermans, P.J., Felix, H.E.S., Chmiela, S., Tkatchenko, A., Müller, K.R.: Schnet: a continuous-filter convolutional neural network for modeling quantum interactions. In: Advances in Neural Information Processing Systems, pp. 991–1001 (2017)
18. Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S.: Pitfalls of graph neural network evaluation. arXiv preprint arXiv:1811.05868 (2018)
19. Shi, C., et al.: Deep collaborative filtering with multi-aspect information in heterogeneous networks. IEEE Trans. Knowl. Data Eng. 1 (2019)
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
21. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1067–1077 (2015)
22. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
23. Velikovi, P., Cucurull, G., Casanova, A., Romero, A., Lió, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (ICLR) (2018)
24. Wang, X., et al.: Heterogeneous graph attention network. In: The World Wide Web Conference, pp. 2022–2032 (2019)
25. Weston, J., Ratle, F., Mobahi, H., Collobert, R.: Deep learning via semi-supervised embedding. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 7700, pp. 639–655. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35289-8_34
26. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S.: A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596 (2019)
27. Xu, B., Shen, H., Cao, Q., Qiu, Y., Cheng, X.: Graph wavelet neural network. In: International Conference on Learning Representations (ICLR) (2019)
28. Yang, Y., Wang, X., Song, M., Yuan, J., Tao, D.: SPAGAN: shortest path graph attention network. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 4099–4105 (2019)
29. Yang, Z., Cohen, W.W., Salakhutdinov, R.: Revisiting semi-supervised learning with graph embeddings. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning, pp. 40–48 (2016)
30. Zhang, M., Cui, Z., Neumann, M., Chen, Y.: An end-to-end deep learning architecture for graph classification. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
31. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Advances in Neural Information Processing Systems, pp. 321–328 (2004)
32. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using Gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on International Conference on Machine Learning, pp. 912–919 (2003)