Neurocomputing xxx (xxxx) xxx

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A unified structure learning framework for graph attention networks

Jinliang Yuan [a,b], Meng Cao [a,b], Hao Cheng [a,b], Hualei Yu [a,b], Junyuan Xie [a,b], Chongjun Wang [a,b,*]

[a] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[b] Department of Computer Science and Technology, Nanjing University, Nanjing 210046, China

ARTICLE INFO

Article history:
Received 14 March 2021
Revised 6 November 2021
Accepted 17 January 2022
Available online xxxx

Keywords:
Graph attention networks
Graph structure learning
Semi-supervised classification

ABSTRACT

Graph Neural Networks (GNNs) have achieved state-of-the-art performance in many fields and attracted a lot of attention in the communit. Most Graph Neural Networks can be merely used when graph-structured data is available. However, many graph structures have noise, or data itself has no graph structures, so learning the dynamic and adaptive graph structures is necessary. In this paper, we propose a unified structure learning framework for Graph Attention Networks. Specifically, we first design a strategy to learn the graph structures. Then we develop a novel attention mechanism based on structure context information of graph and node representations. Further, we devise Structure Learning Graph Attention Networks (SLGAT) and Structure Learning Attention-based Graph Neural Networks (SLAGNN) by using the new attention mechanism on the new graph. Finally, we demonstrate that our approaches outperform competing methods on six standard datasets for the semi-supervised node classification task.

1. Introduction

Graph Neural Networks (GNNs) have shown significant performance progress in many graph-related tasks, such as node classification [1,2], graph classification [3,4], link prediction [5], recommender systems [6–8]. Graph Convolution Network (GCN) [1] is a popular and efficient Graph Neural Network model that aggregates the neighbors' features by a first-order spectral low-pass-type filter. However, GCN aggregates neighbor information equally and cannot distinguish the importance of different neighbors. Graph Attention Network (GAT) [2] adopts self-attention to resolve the issue and proposes to focus on the most relevant neighbors of the target node. Similarly, Attention-based Graph Neural Network (AGNN) [9] uses an attention mechanism over neighbors to capture the relevance of different neighbors and weighs their contributions accordingly. The majority of GNNs can be classified as Message Passing Neural Networks (MPNNs) [10], including GCN, GAT, and AGNN, which aggregate messages from one-hop neighbors at each layer. GAT and AGNN have shown performance improvements in semi-supervised node classification, and they compute the attention between two connected nodes and depend on the node representations

However, GAT and AGNN can only aggregate one-hop neighbors' information in a single layer. And they are shallow models because stacking many layers usually suffers from the over-smoothing problem [11–13]. This implies that receptive fields are limited and they cannot capture long-range interactions. Here, we first try to enlarge the receptive fields and calculate the attention between the target node and its high-order neighbors. The results of semi-supervised node classification on three citation networks are shown in Fig. 1. The y-axis is the classification accuracy. The x-axis represents the size of receptive fields. For example, $k = 2$ means that GAT and AGNN compute attention scores between the target node and its neighbors within two hops. Meanwhile, they aggregate messages from two–hop neighbors at each layer. As we can see, in most cases, the accuracy decreases when the receptive field increases. The results show that enlarging the receptive fields simply is useless for GAT and AGNN.

Firstly, the reason may be that the number of neighbors increases exponentially when the receptive field increases and the target node gathers lots of neighbors' information and loses its inherent representations. The models may suffer from the over-smoothing problem. Secondly, the models are susceptible to overfitting because the parameters for calculating attention increase exponentially with the increase of the receptive fields. Finally, the graph structures are noisy or incomplete due to the inevitable error-prone data measurement or collection. To enlarge the receptive fields and alleviate the over-smoothing and

* Corresponding author at:National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China.
E-mail address: chjwang@nju.edu.cn (C. Wang).

https://doi.org/10.1016/j.neucom.2022.01.064
0925-2312/

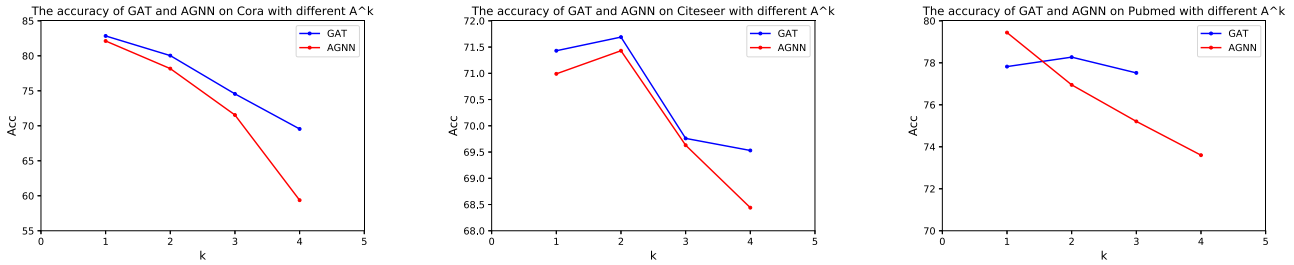

Fig. 1. The results of GAT and AGNN with semi-supervised node classification (GAT is out of memory on Pubmed with $k = 4$).

over-fitting problem, we propose an effective structure learning framework for Graph Attention Networks.[1]

In this paper, we first propose to learn the graph topology via structure learning. And the existing GNNs can aggregate high-order neighbors' representations on the new graph structure, which can help models enlarge the receptive fields. Secondly, we present a novel strategy for attention calculation based on graph structure context information and node representations, which is suitable for the existing Graph Attention Networks. Thirdly, we improve the GAT and AGNN via structure learning and attention calculation. And we propose the Structure Learning Graph Attention Network (SLGAT) and Structure Learning Attention-based Graph Neural Network (SLAGNN). Finally, we conduct a large number of experiments on six standard datasets with semi-supervised node classification to demonstrate that our approaches outperform the state-of-the-art methods.

Our methods have the following advantages. 1) The strategy of structure learning can help models capture long-range interaction between the target node and its high-order neighbors at each layer. And it is a universal framework for the most existing GNNs. 2) The novel method of attention calculation can exploit both topologies of the graph and content features of the nodes, which is a universal strategy for the existing attention mechanisms in GNNs. 3) The improved methods can discriminate dynamically and adaptively which nodes are relevant to the target node for downstream tasks.

The organization of the paper is as follows. Section 2 introduces the problem and related works. In Section 3, we present graph structure learning and a novel scheme of attention calculation. And we apply them to two existing models, GAT and AGNN. Section 4 shows the experiments and analysis. In Section 5, we propose conclusions and future works.

2. Problem and related works

2.1. Problem definition

The graph is defined as $G = (V, E)$, where V is the set of $|V| = n$ nodes and E is the set of edges. $A \in \{0,1\}^{n*\ n}$ represents the adjacency matrix of G, where $A_{ij} = 1$ if there exists an edge between node v_i and v_j, otherwise $A_{ij} = 0$. The features of nodes denote as a matrix $X \in R^{n*\ d}$, where n is the number of nodes and d is the dimension of node features. The labels of nodes represent as $Y \in \{0,1\}^{n*\ C}$ with C is the number of classes. For semi-supervised classification, m nodes ($0 < m \ll n$) have labels Y^L and the labels Y^U of the remaining $n - m$ nodes are missing. Based on a graph $G = (V, E)$ with the node feature matrix X and observed node labels Y^L, the problem of semi-supervised node classification is to learn a classifier $f : (G, X, Y^L) \rightarrow Y^U$ to infer the missing labels Y^U for unlabeled nodes.

2.2. Related works

For semi-supervised node classification, many researchers have proposed graph Laplacian regularization methods based on the assumption that nearby nodes are more likely to have the same labels. Label Propagation [14] is one of the most popular methods. Then ManiReg [15] and ICA [16] are proposed. These early non-neural network methods are efficient but have limit performance. To improve the performance, many unsupervised node embedding approaches have been proposed to embed the nodes in latent Euclidean space. Then supervised learning is applied on node embeddings to train the models. There are many representative graph embedding methods: DeepWalk [17], node2vec [18], LINE [19], and so on. However, those methods do not use the node features and are not end-to-end models. They cannot meet the performance of the state-of-the-art models [9].

In recent years, Graph Neural Networks that use deep learning to process graph-structured data have achieved state-of-the-art performance in graph-related tasks, like semi-supervised node classification [1,20,21]. A representative work is GCN [1] that is an end-to-end model using approximate spectral graph convolution. Subsequently, many variants of GCN are devised, such as SGC [22], APPNP [23], DeepGCNs [24]. However, all the spectral methods learn filters based on the graph structure, which cannot generate node embeddings for previously unseen data. To solve the problem, many researchers have proposed plenty of spatial-based convolutional methods. GraphSAGE [25] presents to sample fixed-size local neighbors and aggregates features from the samples. MoNet [26] designs a unified spatial framework to generalize CNN to non-Euclidean domains.

However, the methods via graph convolutional network cannot capture the relevance between the target node and its different neighbors. Intuitively, neighbors may not be equally important. To distinguish the contribution of neighbors, GAT [2] first applies a self-attention mechanism to Graph Neural Networks and aggregates neighbors' representations based on the attention coefficients. Similarly, AGNN [9] utilizes another attention strategy for GNNs with only a single extra scaler parameter at each layer. But GAT and AGNN only compute the attention between the target node and its directed neighbors. To address the weakness, SPAGAN [27] calculates the attention scores between the center node and its high-order neighbors based on the shortest path. Recently, DAGN [28] introduces a direct multi-hop attention-based graph neural network that diffuses the attention scores from neighbors to high-order neighbors. SuperGAT [29] designs a self-supervised task to predict edges and computes the attention coefficients on the new graph. Compared with the two models, our methods are universal and simple. Besides, we can use the techniques in their models to improve our models, such as LayerNorm and Feed-Forward.

[1] In the paper, Graph Attention Networks represent that Graph Neural Networks aggregate message based on the attention mechanism, like Graph Attention Network (GAT), Attention-based Graph Neural Network (AGNN).

Most of the above Graph Neural Networks can only be used when graph-structured data is available. But the existing graph structures are often noisy or incomplete that cannot reflect the real graph topology, or many data have no graph structures, like natural language, image. Therefore, many researchers have proposed to learn the graph structures. AGCN [30] uses an adaptive graph convolutional neural network that combines a task-driven adaptive graph learning approach for each graph data with training. GLCN [31] presents to learn graph data representations by integrating both graph learning and graph convolution in a unified network architecture. IDGL [32] utilizes an iterative method to learn better graph structures and node embeddings. Pro-GNN [33] designs a general framework for learning graph topology and a robust GNN model for defending the adversarial attacks about the perturbation of graph structures. DAGG [34] introduces a data-adaptive graph generation to learn the graph structures among different traffic series. Recently, SimP-GCN [35] proposes a node similarity preserving aggregation method to balance information from graph structure and node features based on structure learning. However, all the graph structure learning methods are applied to Graph Convolutional Networks. To the best of our knowledge, there are no methods to unify the graph structure learning and attention-based Graph Neural Networks.

3. Models

In this section, we first review the original GAT and AGNN and then propose a unified framework for graph structure learning. Further, we introduce our Structure Learning Graph Attention Network (SLGAT) and Structure Learning Attention-based Graph Neural Network (SLAGNN).

3.1. GAT and AGNN

The original GAT designs a shared linear transformation for each node via a weight matrix W and then computes attention coefficients between the target node and its directed neighbors based on a shared attentional mechanism a.

$$e_{ij}^t = a(W^t H_i^t, W^t H_j^t), \tag{1}$$

where coefficient e_{ij}^t indicates the importance of node j's representation to node i in the t layer. The a is a single-layer feedforward neural network with *LeakyReLU* nonlinearity function. To make coefficients easily comparable across different nodes, GAT uses the *softmax* function to normalize the attention coefficients for each target node. The attention scores can be expressed as

$$\alpha_{ij}^t = \frac{exp(LeakyReLU(\bar{a}^T[W^t H_i^t || W^t H_j^t]))}{\sum_{k\in N_i\cup\{i\}} exp(LeakyReLU(\bar{a}^T[W^t H_i^t || W^t H_k^t])}, \tag{2}$$

where N_i represents the neighbors of node i. Further, GAT aggregates the representations of neighbors via the attention coefficients and applies a single neural network layer with a nonlinearity function to update the node representation

$$H_i^{t+1} = \sigma(\sum_{j\in N_i\cup\{i\}} \alpha_{ij}^t W^t H_j^t). \tag{3}$$

Finally, GAT performs multi-head attention on the prediction layer of the network to obtain the final node embeddings

$$H_i^{t+1} = \sigma(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i\cup\{i\}} \alpha_{ij}^{tk} W^{tk} H_j^t). \tag{4}$$

Unlike GAT, AGNN designs another way to calculate attention coefficients with only a single scalar parameter at each layer. The attention score from node j to node i is

$$P_{ij}^t = \frac{exp(\beta^t \cos(H_i^t, H_j^t))}{\sum_{k\in N(i)\cup\{i\}} exp(\beta^t \cos(H_i^t, H_k^t))}. \tag{5}$$

Then AGNN conducts message-passing based on the attention scores to update the node representations:

$$H_i^{t+1} = \sum_{j\in N_i\cup\{i\}} P_{ij}^t H_j^t. \tag{6}$$

It is worth noting that AGNN uses a single-layer feedforward neural network with *ReLU* for reducing the dimension of node features at the beginning of the model.

$$H^1 = ReLU(XW^0). \tag{7}$$

3.2. A unified framework of graph structure learning

Many data are generally in non-Euclidean domains, such as social networks, molecules, and traffic networks. These data are usually modeled as graphs, which can capture varying neighborhood vertex connectivity. The graph structure describes the relationships between nodes. For example, in the molecule graph, the relationship may be the chemical bond between atoms. However, the existing graph structure may be noisy or incomplete due to unavoidable errors in data measurement or acquisition. In many cases, the data itself does not have a graph structure, but the problem is appropriate to use the graph to handle, like point cloud segmentation. And the fixed graph structure may not be optimal for various downstream tasks. Therefore, it is significant to learn an adaptive graph structure for downstream tasks whose input data have or have not graph structures.

Most existing methods cast the graph structure learning problem as a metric learning problem based on distance or feature similarity. The methods of metric learning in graph structure learning generally include radial basis function kernel [30], attention mechanisms [31], and cosine similarity [32], which are supposed to be learnable and have shown promising performance. However, those metric learning methods usually contain many constraints about graph regularization based on some prior knowledge of graphs, such as smoothness, connectivity, sparsity, low-rank, and so on. Although these constraints are indeed beneficial for graph structure learning, the non-differentiable of some constraints and the addition of additional parameters make model optimization more difficult. Besides, the methods of graph structure learning usually only depend on the features or representations of nodes, which ignore the global structure similarity.

To address these weaknesses, we design a novel unified framework of graph structure learning based on node representations and global structure information. The overview of graph structure learning is shown in Fig. 2. Firstly, we can use any existing metric learning methods that include cosine similarity, attention mechanisms, and radial basis function kernel to learn graph topology, especially when the data itself has no graph structure. The metric similarity between node i and node j is

$$\bar{S}_{ij} = M(H_i, H_j), \tag{8}$$

where H_i and H_j are the representations of node i and node j, respectively. The representations of nodes are the features of nodes at the beginning. M is a method of metric learning, such as attention mechanism

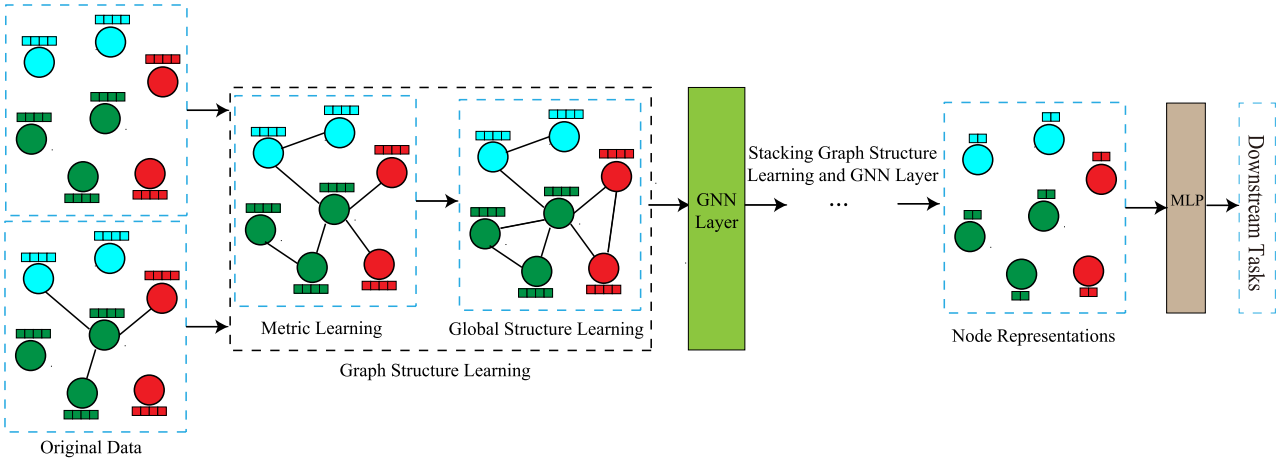

Fig. 2. The unified framework of graph structure learning for the GNNs.

$$\bar{S}_{ij} = \frac{exp(ReLU(\bar{a}^T|H_i - H_j|))}{\sum_{j=1}^{n} exp(ReLU(\bar{a}^T|H_i - H_j|))}, \tag{9}$$

or cosine similarity

$$\bar{S}_{ij} = cos(H_i, H_j) \tag{10}$$

Instead of using complex graph regularization constraints, we normalize the matrix and propose to learn a sparse graph structure via a threshold parameter k. The sparse matrix is denoted as

$$\bar{S}_{ij} = \begin{cases} \bar{S}_{ij}, & \text{if} \bar{S}_{ij} > k \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

The learning similarity matrix $\bar{S}$ can be regarded as the weight matrix of the graph, which may be dynamic at each layer and adaptive for downstream tasks. Further, we propose to modify the similarity matrix by using the global topology information. We first transform the weight matrix into an adjacency matrix, then calculate the transition matrix by graph diffusion, which reflects the global structure proximity. Two popular instantiations of the generalized graph diffusion are Personalized PageRank (PPR) [36] and heat kernel [37]. The closed-form solutions of PPR and heat kernel are expressed as follows:

$$S^{ppr} = \alpha(I_n - (1-\alpha)D^{1/2}AD^{1/2})^{-1}, \tag{12}$$

$$S^{heat} = exp(tAD^{-1} - t), \tag{13}$$

where α represents the teleport probability of random walk and t denotes diffusion time. The D is diagonal matrix of node degree, where $D_{ij} = \sum_j A_{ij}$. But the two methods are time-consuming. Therefore, we adopt the PPMI [20] matrix to represent the global structure proximity in this paper.

The PPMI has long been regarded as a state-of-the-art model to measure the similarity of words, which has been extensively investigated in natural language processing [38,39]. Next, we introduce the specific calculation method of PPMI matrix. Firstly, taking each node as the root node to conduct γ random walks, and the length of each walk is q. Then we can compute the number of times that two nodes appear on the same walk path, which is denoted as frequency matrix F. Based on F, we calculate the PPMI matrix P as

$$P_{ij} = \frac{F_{ij}}{\sum_{i,j} F_{ij}}; \tag{14}$$

$$P_{i,*} = \frac{\sum_j F_{ij}}{\sum_{i,j} F_{ij}}; \tag{15}$$

$$P_{*j} = \frac{\sum_i F_{ij}}{\sum_{i,j} F_{ij}}; \tag{16}$$

$$P_{ij} = \max\{\log(\frac{P_{ij}}{P_{i,*}P_{*j}}), 0\}. \tag{17}$$

In fact, the P_{ij} is the estimated probability that node i and j occur on the same random walk path, which reflects the global structure proximity of the two nodes. Similarly, we normalize the matrix and use a threshold parameter k to obtain a sparse PPMI matrix

$$P_{ij} = \begin{cases} P_{ij}, & \text{if} P_{ij} > k \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

Integrating the learning similarity matrix $\bar{S}$ and PPMI matrix P, which reflect node representation similarity and global structure proximity, we can obtain the graph structure S:

$$S = \bar{S} + P. \tag{19}$$

In some cases, the data have intrinsic graph structure A. After normalizing the adjacency matrix A, we combine the normalized adjacency matrix $\hat{A}$ and learned graph structure matrix to get the final graph structure $\bar{A}$:

$$\bar{A} = S + \hat{A}. \tag{20}$$

The $\bar{S}_{ij}$ represents the feature similarity of the two nodes, and P_{ij} reflects the global structure proximity of the two nodes. We normalize the two matrices $\bar{S}, P$ to learn a sparse graph structure via a threshold parameter k. The $S = \bar{S} + P$ is also a sparse graph, and S_{ij} produces a new edge when there is no edge between node i and node j on the original graph A, and $\bar{S}_{ij} > k$ or $P_{ij} > k$. Of course, we can transform the weight matrix into an adjacency matrix, and then the most off-the-shelf Graph Neural Networks can be used on the new graph. The learned graph structure can be dynamic at each layer and be adaptive for the downstream tasks. Based on the learned graph topology, the Graph Neural Networks can aggregate the representations of high-order neighbors at each layer, which enlarges the receptive fields.

3.3. SLGAT and SLAGNN

The 3.2 section describes a universal framework of graph structure learning that is suitable for many existing Graph Neural Networks. Here, we apply the strategy to Graph Attention Network (GAT) and Attention-based Graph Neural Network (AGNN) with a novel attention mechanism. The overview of proposed models, Structure Learning Graph Attention Network (SLGAT) and Structure Learning Attention-based Graph Neural Network (SLAGNN), is shown in Fig. 3. Firstly, we use the proposed graph structure learning scheme to learn a graph structure as the input graph. Then, we design a novel attention calculation method by using both the representations of nodes and the graph structure. Specifically, we can use any existing attention mechanism to compute the attention coefficient e_{ij}^t:

$$e_{ij}^t = Att(H_i^t, H_j^t), \tag{21}$$

where the coefficient e_{ij}^t mainly indicates the importance of node j's content features to node i in the t layer. In this paper, we choose the methods as shown in Eq. 2 (SLGAT) or Eq. 5 (SLAGNN).

However, these attention mechanisms are computed mainly based on node content features, which ignores the graph structure. Therefore, we propose to integrate the attention coefficient e_{ij}^t and the learned structure proximity S_{ij}^t to obtain the final attention coefficient α_{ij}^t:

$$\alpha_{ij}^t = e_{ij}^t + \lambda S_{ij}^t. \tag{22}$$

where λ is a scalar parameter that adjusts the importance of node content features and learned graph structure. The new attention coefficients can reflect the relevance of the target node and its different neighbors in terms of both content feature proximity and structural proximity. Finally, we aggregate the representations of neighbors and update the representations of the target node

$$H_i^t = \sigma\Big(\sum_{j\in N_i\cup\{i\}} \alpha_{ij}^t W^t H_j^{t-1}\Big). \tag{23}$$

The SLGAT uses multi-head attention on the final layer:

$$Z = \sigma\Big(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i\cup\{i\}} \alpha_{ij}^k W^k H_j\Big). \tag{24}$$

And the SLAGNN uses a single layer MLP as the final layer:

$$Z = softmax(WH). \tag{25}$$

For semi-supervised node classification, the optimization objective is the cross-entropy loss function:

$$\mathcal{L} = -\sum_{l\in Y^L}\sum_{c=1}^{C} Y_c^l \ln Z_c^l, \tag{26}$$

where Y^L is the set of labeled nodes and C is the number of classes.

3.4. Discussions

In this paper, we modify the original graph structure from two aspects. One is to calculate the feature similarity of the two nodes, $\bar{S}$. And another is to estimated probability that node i and j occur on the same random walk path, PPMI, which reflects the global structure proximity of the two nodes. The two matrices represent the feature proximity and global structure proximity, respectively, which are the complementary for the original graph structure. The previous graph structure learning methods do not consider the global structure similarity.

The attention mechanism of GNNs can be regarded as a special graph structure learning, which learns the weight matrix from an adjacency matrix. The attention mechanism also can be used to calculate the similarity between nodes for graph structure learning, such as Eq. 9. Therefore, it is reasonable to add the learned similarity matrix into the attention coefficients. Based on the learned graph structure, the improved two models can compute the attention scores between the target node and its high-order neighbors at each layer, which helps models capture long-range interaction between nodes and improves the performance of models. Meanwhile, the new attention coefficients combine the proximity of content features and topology of nodes, which can represent the interactions between nodes more accurately and distinguish the importance of the neighbors to the target node. However, the original Graph Attention Networks only calculate the attention scores for direct neighbors based on node features or representations.

The SLGAT and SLAGNN contain two layers of message aggregation like GAT and AGNN. The main difference between SLGAT and SLAGNN is the attention mechanism. The SLGAT calculates the attention like Eq. 2 and SLAGNN uses Eq. 5 to get the attention coefficients. And SLAGNN only uses a single parameter at each layer to calculate attention and doesn't use the multi-head attention mechanism. However, SLGAT needs many parameters to calculate attention and uses the multi-head attention mechanism to aggregate information.

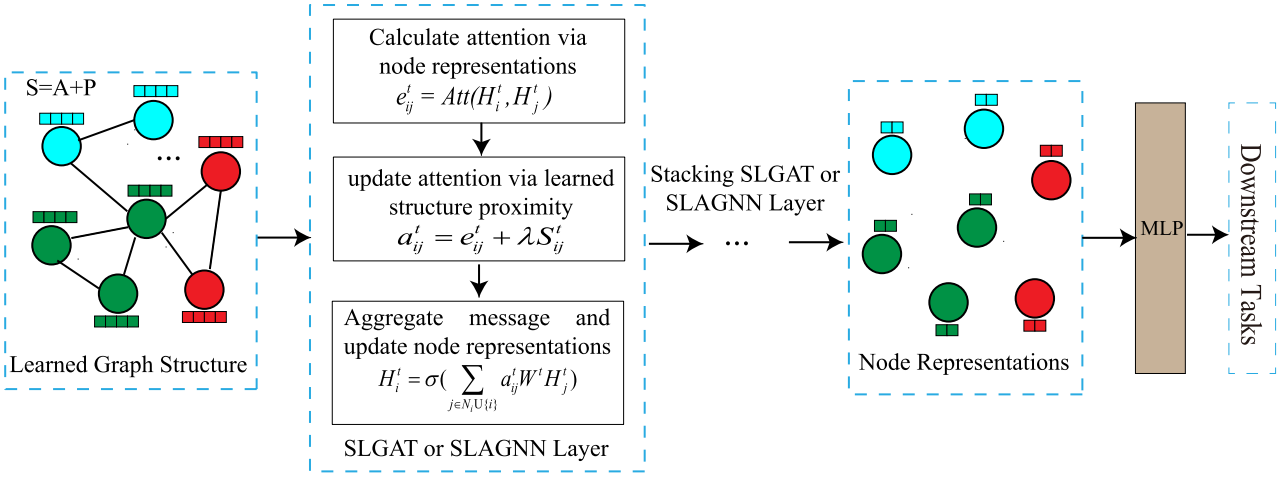

Fig. 3. The Overview of SLGAT or SLAGNN. The input data are learned graph structure and features of nodes, then we stack SLGAT or SLAGNN layers to learn node representations.

Finally, it should be noted that we do not learn the graph structure and node representations iteratively as shown in Fig. 2. To reduce the complexity of the model and prevent overfitting, we only learn the graph structure at the beginning as shown in Fig. 3. The similarity metric funtion like Eq. 10 used in this paper computes similarity scores for all pairs of nodes, which requires $O(n^2 * d_x)$ time complexity. The time complexity of PPMI is $O(n\gamma q^2)$. The PPMI and metric function could be calculated in parallel and only need to be calculated once in our models. And the time complexity of GAT and AGNN is $O(nd_xd_h + |E|d_h)$, where d_x and d_h are the dimensions of features and hidden layer, respectively. The time complexity of our models is $O(n^2d_x + n\gamma q^2 + nd_xd_h + |E|d_h)$.

4. Experiments

In this section, we present the results of semi-supervised node classification on six standard datasets for verifying the performance of SLGAT and SLAGNN. We first introduce the datasets, comparative baselines, and experimental setup. And we show the experimental results of our models and the state-of-the-art baselines under various experimental settings. Further, we give the ablation study and discuss the advantages and limitations of the proposed methods.

4.1. Datasets and baselines

Datasets. We utilize six standard datasets, including three citation networks, a co-authorship network, and two co-purchase networks. The citation networks include Cora, Citeseer, Pubmed [40], where the nodes represent documents and edges are their citation links. The features of nodes are the representations of bag-of-words for documents. And the labels of nodes denote what field the corresponding document belongs to. The co-authorship network is Coauthor CS [41], in which nodes are authors and edges represent the authors co-authored a paper. Node features are the keywords for authors' papers and each node has a label denoting the most active research field. The Amazon Computers and Amazon Photo [41] are co-purchase networks where nodes are the goods and edges represent the two goods frequently bought together. The node features are the bag-of-words for product reviews. And the label of a node is the category of product. To make a fair comparison, we closely follow the experimental setup in [1,41,42], which is the community convention. The detailed statistics of datasets are shown in Table 1.

Baselines. We compare our methods with the following strong baselines and state-of-the-art methods.

MLP uses the features of nodes as the representations of nodes, which does not leverage the graph structure.

Four Graph Neural Networks based on graph convolution: Cheb [43] designs a fast localized convolutional filter by using the Chebyshev polynomial. **GCN** [1] further simplifies the Cheb and learns node embeddings via a localized first-order approximation of spectral graph convolutions. **SGC** [22] removes the nonlinearities of GCN for reducing the complexity of the model without compromise for the performance of the model. **APPNP** [23] learns node representations via a novel propagation scheme, which is based on personalized PageRank.

Four Attention-based Graph Neural Networks: GAT [2] aggregates the messages of neighbors based on a self-attention mechanism, which can specify different weights to different nodes in a neighborhood. **AGNN** [9] learns a dynamic and adaptive local summary of the neighborhood based on a novel attention mechanism that has only a single scalar parameter at each layer. **SPAGAN** [27] calculates the attention scores based on the shortest path between nodes. **DAGN** [28] is the Direct multi-hop Attention-based Graph neural Network, which can calculate attention between the target node and its high-order neighbors by diffusing attention scores.

Graph structure learning method: GCN-k uses GCN on the new graph structure (Eq. 20). **IDGL** [32] proposes an end-to-end framework for jointly and iteratively learning graph structure and node embeddings. **SimP-GCN** [35] proposes a framework that can preserve node similarity and exploit graph structure.

We use the source codes of comparative models including Cheb, GCN, SGC, APPNP, GAT, AGNN, which are from the website[2]. And they are based on the PyTorch Geometric. For comparison, we also implement the SLGAT and SLAGNN by using the PyTorch Geometric, which will be public when the paper publishes. The source code of IDGL is from the website[3]. And the code of SimP-GCN is from the website[4]. We conduct 10 runs for these models following the setup of original works. The results of SPAGAN and DAGN are from the original works [27,28]. For the proposed models, we initialize the parameters using initialization in [44] and train them by using Adam optimizer [45]. We set the hyper-parameters including: weight decay=$5e-4$, length of random walk $q = 5$, the number of random walk for each node $\gamma = 40$, dropout = 0.8. We tune the following hyper-parameters: (1) learning rate $lr \in \{0.005, 0.01, 0.015, 0.02\}$. (2) dimensions of hidden layer {16, 64}. (3) $\lambda \in \{0.1, 0.2, 1\}$. (4) epochs {200, 500, 2000}. (5) threshold $k \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ Unless otherwise specified, the parameters of SLGAT and SLAGNN are the same as GAT and AGNN, respectively.

4.2. Results of semi-supervised classification

We first report the accuracies of semi-supervised node classification on the benchmarks that are usually used to evaluate the performance of models in the community. Results are summarized in Table 2 and Table 3. Our models achieve the best performance on five datasets. And IDGL achieves the best accuracy on Pubmed, which shows the advantages of graph structure learning. Our SLGAT and SLAGNN are always better than GAT and AGNN on all datasets, which proves the effectiveness of our methods. For example, Our SLAGT is 1.24%, 2.58%, 2.63%, 0.47%, 5.48%, 2.14% relative improvement over GAT on Cora, Citeseer, Pubmed, CS, Computers, and Photo, respectively. It should be noted that the results of DAGN are from the original work without LayerNorm for fair comparison with our models because this method is orthogonal to our methods and can be used to improve our models, which is left to future research. The results of DAGN, SPAGAN, IDGL are missing in the Table 3 is since the results are not given in the original papers.

From the tables, we can also find that the Graph Neural Networks are better than MLP, proving the advantages of GNNs that learn node representations by integrating both graph structure and node features. Secondly, in most cases, the attention-based methods outperform the graph convolution methods, which proves that it is effective to assign different weights to different neighbors during the message passing. Finally, the model with graph structure learning is generally superior to the models based on graph convolution and graph attention, which shows that it is useful to learn graph structure. Our models combine the advantages of graph attention mechanism and graph structure learning, which outperform other structure learning methods in most datasets. The structure learning updates the graph topology and can enlarge the receptive field for Graph Attention Networks, and the new attention mechanism integrates the node content feature

[2] https://github.com/rusty1s/pytorch_geometric
[3] https://github.com/hugochan/IDGL
[4] https://github.com/ChandlerBang/SimP-GCN

Table 1
Statistics of the six datasets.

Datasets	Nodes	Edges	Classes	Feature	Training	Validation	Test
Cora	2708	5429	7	1433	20 per class	500	1000
Citeseer	3327	4732	6	3707	20 per class	500	1000
Pubmed	19717	44338	3	500	20 per class	500	1000
CS	18333	81894	15	6808	20 per class	30 per class	Rest nodes
Computers	13381	245778	10	767	20 per class	30 per class	Rest nodes
Photo	7487	119043	8	745	20 per class	30 per class	Rest nodes

Table 2
Results of semi-supervised node classification in terms of accuracy.

Methods/Datasets	Cora	Citeseer	Pubmed
MLP	59.20 ± 0.72	56.98 ± 0.99	72.86 ± 0.85
Cheb	79.34 ± 0.64	67.75 ± 0.86	78.16 ± 0.74
GCN	81.78 ± 0.82	70.93 ± 1.17	79.01 ± 0.55
SGC	79.28 ± 0.47	70.67 ± 0.08	76.88 ± 0.06
APPNP	83.14 ± 0.79	71.21 ± 0.84	80.03 ± 0.24
DAGN	83.80 ± 0.60	71.10 ± 0.50	79.80 ± 0.20
SPAGAN	83.60 ± 0.50	73.00 ± 0.40	79.60 ± 0.40
GCN-k	82.35 ± 0.92	71.59 ± 0.46	78.70 ± 0.14
SimP-GCN	82.56 ± 0.74	72.48 ± 0.63	80.94 ± 0.22
IDGL	84.30 ± 0.30	71.50 ± 0.20	**82.90 ± 0.20**
GAT	82.84 ± 0.59	71.43 ± 1.25	77.82 ± 0.48
SLGAT	83.87 ± 0.49	**73.27 ± 0.56**	79.87 ± 0.51
Improve	1.24%	2.58%	2.63%
AGNN	82.12 ± 0.60	70.99 ± 0.94	79.44 ± 0.41
SLAGNN	**84.86 ± 0.59**	72.19 ± 0.59	79.52 ± 0.21
Improve	3.34%	1.69%	0.10%

Table 3
Results of semi-supervised node classification in terms of accuracy.

Methods/Datasets	CS	Computers	Photo
MLP	86.96 ± 0.60	53.87 ± 2.90	70.12 ± 2.58
Cheb	90.85 ± 0.43	72.73 ± 1.68	86.66 ± 0.40
GCN	89.45 ± 0.47	76.35 ± 1.32	89.37 ± 0.85
SGC	90.08 ± 0.52	59.34 ± 1.06	71.56 ± 0.56
APPNP	91.44 ± 0.25	73.10 ± 0.24	85.48 ± 0.20
GCN-k	90.40 ± 0.44	76.94 ± 0.29	89.46 ± 0.18
SimP-GCN	91.11 ± 0.56	78.24 ± 2.35	89.42 ± 0.57
GAT	90.98 ± 0.20	75.51 ± 0.56	88.38 ± 1.08
SLGAT	91.41 ± 0.05	**79.65 ± 1.53**	**90.27 ± 0.31**
Improve	0.47%	5.48%	2.14%
AGNN	89.01 ± 0.97	76.64 ± 3.14	88.13 ± 1.58
SLAGNN	**91.74 ± 0.39**	77.42 ± 0.37	89.27 ± 0.24
Improve	3.07%	1.02%	1.29%

proximity and structure proximity, which can obtain more accurate attention coefficients.

In addition to measuring the performance of models from a quantitative perspective using accuracy, we further introduce a data visualization technique t-SNE [46] to evaluate the effectiveness of the models. The t-SNE is a technique that combines the dimension reduction and visualization, which can reflect the distinguishability of node presentations in the graph. The visualization results on Cora are shown in Fig. 4. Compared to the seven baselines, our models have relatively better or matching discrimination boundaries, which further demonstrates the advantages of SLAGT and SLAGNN.

4.3. Training set sizes

In the real world, the labeled data is scarce. Here, we try to explore the performance of models with few training data. We select nine comparative methods and report the mean classification accuracy after ten runs. We set four groups of experiments with different numbers of training nodes. For example, the training set size is 5, which means that we randomly select five nodes from each class as the training set. The validation set and test set keep the same with Table 1. We set the number of layers of SGC and APPNP as ten because they can explore a large receptive field. The classification results of comparative evaluation experiments on Cora are summarized in Table 4. We can see that our SLGAT and SLAGNN are superior to all the models under different training set sizes. The APPNP and SGC perform better than Cheb and GCN, suggesting that enlarging the receptive fields is useful. The reason may be that long-range interaction can provide weakly supervised information when the training set is small. The advantages of our models over the comparison models are more obvious when the training set is small. Specifically, SLGAT and SLAGNN can improve the GAT and AGNN by a margin of 29.74% and 35.48% on Cora, respectively, when the training set size is 1, suggesting that the proposed graph structure learning can help the models aggregate information from high-order neighbors without stacking many layers.

4.4. Ablation study

In this section, we first conduct the ablation study to verify the effect of designed graph structure learning and attention mechanism. Then we further analyze the influence of parameters on the models. Firstly, we show the results of variations of models on semi-supervised node classification in Table 5. The '-M' means the models only use metric learning to update graph structure, '-S' means the models only leverage global structure learning, and '-PPR', '-Heat' means that we use PPR, Heat instead of PPMI to learn global structure. We can find that the two structure learning methods can both improve the performance of models in most cases. And integrating these two methods can further improve the performance of models. The PPR and heat also can improve the performance of models. However, the PPR and Heat needs to calculate the inverse of the matrix, and the time complexity is higher. Some approximate calculation methods [47,48] can be used in the future. The results show that metric learning and global structure learning are both necessary for structure learning.

Then we report the results of SLGAT and SLAGNN with different threshold k (In Eq. 11 and Eq. 18). As shown in Fig. 5, the SLGAT and SLAGNN achieve the best performance when the $k = 0.1$ on Cora and $k = 0.2$ on Citeseer. It should be noted that each element of the matrix in Eq. 11 and Eq. 18 is less than or equal to 1 after normalization. And we do not use the graph structure learning when $k = 1$. The accuracy of models with $k = 1$ is less than the models with $k \in \{0.1, 0.2\}$, which demonstrates that graph structure learning enables to improve the performance of the models. We also find that the learned graph structure is a dense matrix when $k = 0$, but the real-world graph structure is the sparse matrix, in which case the performance of the models decreases. Therefore, we need to select a reasonable threshold for learning a sparse graph structure.

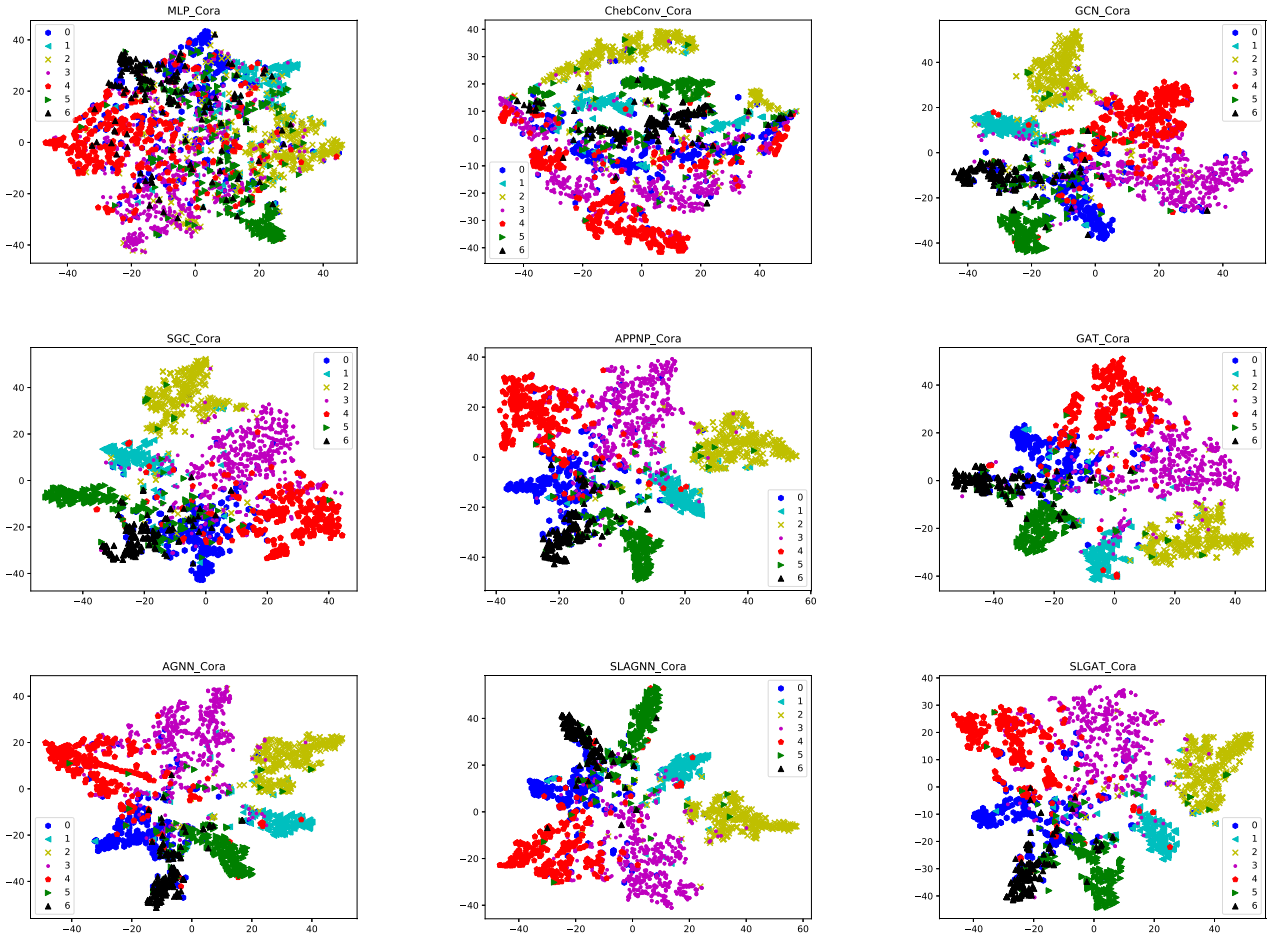

Fig. 4. The t-SNE visualization of node representations on Cora.

Table 4
Results of semi-supervised node classification with different training set sizes on Cora.

Methods/Training Nodes	1	5	10	15
MLP	27.64 ± 4.18	37.47 ± 1.96	48.22 ± 1.71	51.57 ± 1.56
Cheb	28.24 ± 4.73	53.69 ± 1.85	67.07 ± 2.50	70.17 ± 1.88
GCN	38.68 ± 3.91	64.07 ± 1.78	76.53 ± 0.65	77.02 ± 0.64
SGC	45.19 ± 2.20	70.10 ± 0.35	77.60 ± 0.00	79.02 ± 0.08
APPNP	60.04 ± 4.25	76.47 ± 0.83	80.58 ± 0.55	81.71 ± 0.88
GCN-k	56.71 ± 2.39	73.37 ± 0.52	76.92 ± 0.73	80.26 ± 0.63
SimP-GCN	39.73 ± 7.81	70.92 ± 2.83	77.07 ± 1.40	78.79 ± 1.10
GAT	48.61 ± 4.21	75.38 ± 1.10	80.05 ± 0.87	80.97 ± 0.81
SLGAT	63.07 ± 0.87	76.75 ± 0.60	**80.72 ± 1.28**	**82.64 ± 0.32**
Improve	29.74%	1.82%	0.84%	2.06%
AGNN	47.89 ± 4.21	72.46 ± 0.73	77.33 ± 1.06	79.61 ± 0.56
SLAGNN	**64.88 ± 1.79**	**77.17 ± 0.76**	80.71 ± 0.94	82.00 ± 0.68
Improve	35.48%	6.50%	4.37%	3.00%

Further, we explore the effect of structure proximity on attention calculation. The results are shown in Fig. 6. The $\lambda = 0$ in Eq. 22 means that we do not utilize the structure proximity for attention calculation. The accuracy of the models with $\lambda = 0$ is lower than the models with $\lambda = 0.1$, which verify that the proposed attention mechanism integrating feature proximity and structure proximity can improve the accuracy of models. We also find that SLGAT is sensitive to different λ. But the λ has little influence on the performance of SLAGNN. We suspect that the attention coefficient calculated in SLAGNN is more stable with fewer parameters comparing with SLGAT. Finally, we report the results of SLGAT and SLAGNN with different learning rates in Fig. 7. We can find that the different models achieve the best accuracy under different learning rates. For example, SLGAT and SLAGNN have the best performance when the learning rate are 0.015 and 0.02 on Cora, respectively. The results show that our models are sensitive to the learning rate. We need to choose a reasonable learning rate on different datasets, which is the limit of our models. Besides, our models require additional time for graph structure learning.

Table 5
Results of semi-supervised node classification in terms of accuracy.

Methods/Datasets	Cora	Citeseer	Pubmed
GAT	82.84 ± 0.59	71.43 ± 1.25	77.82 ± 0.48
SLGAT-M	83.32 ± 0.75	72.24 ± 1.05	78.81 ± 0.37
SLGAT-S	83.51 ± 0.57	72.29 ± 0.60	79.14 ± 0.17
SLGAT-PPR	83.40 ± 0.78	72.24 ± 1.05	78.81 ± 0.37
SLGAT-Heat	83.32 ± 0.75	72.21 ± 0.95	78.89 ± 0.35
SLGAT	83.87 ± 0.49	**73.27 ± 0.56**	**79.87 ± 0.51**
AGNN	82.12 ± 0.60	70.99 ± 0.94	79.44 ± 0.41
SLAGNN-M	84.01 ± 0.51	71.36 ± 0.59	79.42 ± 0.71
SLAGNN-S	84.41 ± 0.62	71.62 ± 0.67	79.49 ± 0.45
SLAGNNT-PPR	83.66 ± 0.88	72.07 ± 0.42	79.59 ± 0.45
SLAGNN–Heat	84.06 ± 0.52	72.43 ± 0.79	79.46 ± 0.16
SLAGNN	**84.86 ± 0.59**	72.19 ± 0.59	79.52 ± 0.21

5. Conclusions

We have proposed the Structure Learning Graph Attention Network (SLGAT) and Structure Learning Attention-based Graph Neural Network (SLAGNN), two novel attention-based GNNs that integrate the advantages of graph structure learning and attention mechanism. The proposed graph structure learning is a unified framework for most existing GNNs. Meanwhile, the framework is suitable for graph-structured data or data without graph structure. The novel attention mechanism computes attention scores by using the node representations and structure proximity, which enables to fully exploit both topologies of the graph and content features of the nodes. We conduct a lot of experiments for semi-supervised node classification on six standard datasets, which demonstrates that our models can achieve better or matching

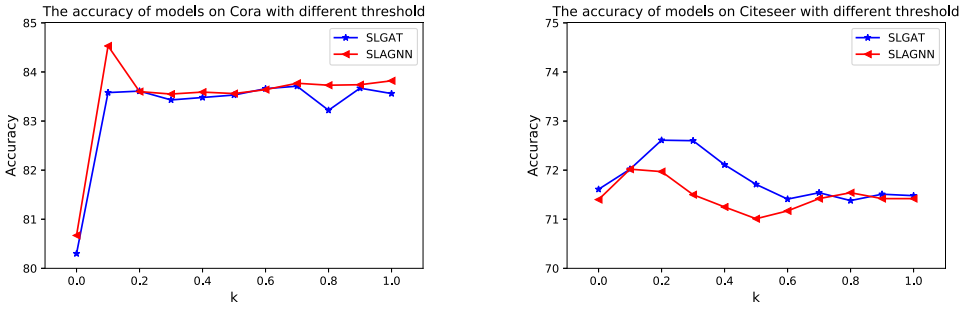

Fig. 5. The accuracy of SLGAT and SLAGNN with different threshold k.

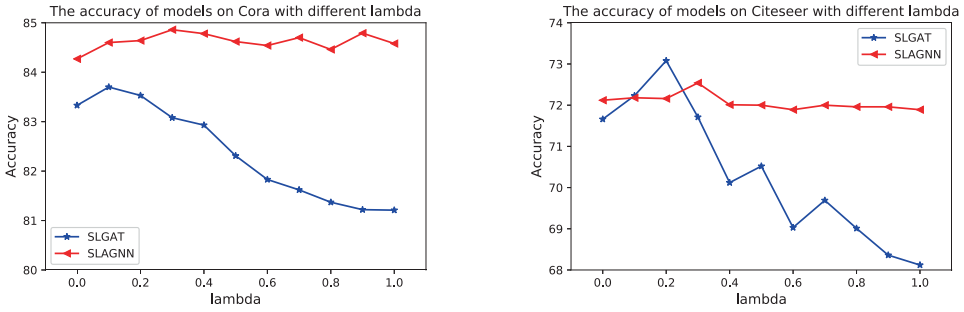

Fig. 6. The accuracy of SLGAT and SLAGNN with different λ.

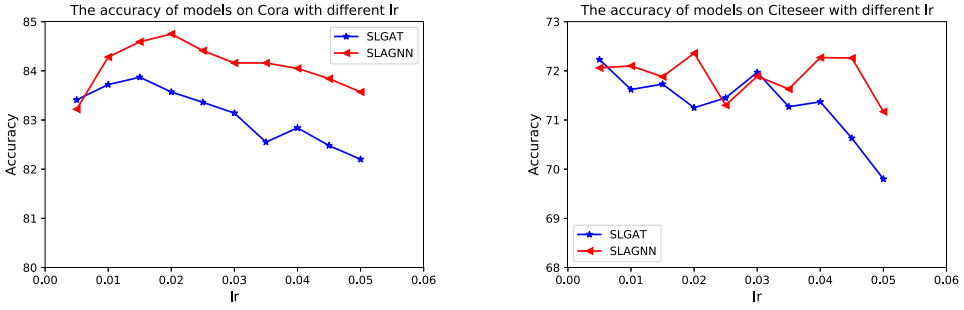

Fig. 7. The accuracy of SLGAT and SLAGNN with different learning rate.

performance compared with state-of-the-art baselines. The advantages of our models are more obvious when the training set is smaller, which shows the potential of our models under the few training set. In the future, we will apply our models to the data without graph structure and design novel attention mechanisms for GNNs. And we will explore the robustness of our models to structural attacks.

CRediT authorship contribution statement

Jinliang Yuan: Conceptualization, Methodology, Software, Writing - original draft. **Meng Cao:** Data curation, Writing - original draft. **Hao Cheng:** Visualization, Validation. **Hualei Yu:** Data curation. **Junyuan Xie:** Writing - review & editing. **Chongjun Wang:** Writing - review & editing, Resources.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This paper is supported by the National Key Research and Development Program of China (Grant No. 2018YFB1403400), the National Natural Science Foundation of China (Grant No. 61876080), the Key Research and Development Program of Jiangsu(Grant No. BE2019105), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

References

[1] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017. OpenReview.net, 2017..

[2] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018. OpenReview.net, 2018..

[3] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In 32nd International Conference on Neural Information Processing Systems, NeurIPS 2018, December 3–8, 2018, Montréal, Canada, pages 4805–4815, 2018..

[4] Hongyang Gao and Shuiwang Ji. Graph u-nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9–15 June 2019, Long Beach, California, USA, volume 97, pages 2083–2092. PMLR, 2019..

[5] Meng Qu, Yoshua Bengio, and Jian Tang. GMNN: graph markov neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9–15 June, 2019, Long Beach, California, USA, volume 97, pages 5241–5250. PMLR, 2019..

[6] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, Jure Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 974–983.

[7] Yu Wenhui, Zheng Qin, Graph convolutional network for recommendation with low-pass collaborative filters, in: In International Conference on Machine Learning PMLR, 2020, pp. 10936–10945.

[8] Jianing Sun, Wei Guo, Dengcheng Zhang, Yingxue Zhang, Florence Regol, Yaochen Hu, Huifeng Guo, Ruiming Tang, Han Yuan, Xiuqiang He, and Mark Coates. A framework for recommending accurate and diverse items using bayesian graph convolutional neural networks. In KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020, pages 2030–2039. ACM, 2020..

[9] Kiran K Thekumparampil, Chong Wang, Sewoong Oh, and Li-Jia Li. Attention-based graph neural network for semi-supervised learning. arXiv preprint arXiv:1803.03735, 2018..

[10] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017, volume 70, pages 1263–1272. PMLR, 2017..

[11] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018, pages 3538–3545. AAAI Press, 2018..

[12] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In The Thirty-Fourth AAAI Conference on Artificial Intelligence, New York, NY, USA, February 7-12, 2020, pages 3438–3445. AAAI Press, 2020..

[13] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020. OpenReview.net, 2020..

[14] Xiaojin Zhu. Learning from labeled and unlabeled data with label propagation. 2002..

[15] Mikhail Belkin, Partha Niyogi, Vikas Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, J. Mach. Learn. Res. 7 (11) (2006).

[16] Qing Lu and Lise Getoor. Link-based classification. In Tom Fawcett and Nina Mishra, editors, Proceedings of the Twentieth International Conference on Machine Learning, ICML 2003, August 21–24, 2003, Washington, DC, USA, pages 496–503. AAAI Press, 2003..

[17] Bryan Perozzi, Rami Al-Rfou, Steven Skiena, Deepwalk: online learning of social representations, in: Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, Rayid Ghani (Eds.), The 20th International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA, 2014, pp. 701–710.

[18] Aditya Grover, Jure Leskovec, node2vec: Scalable feature learning for networks, in: Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, Rajeev Rastogi (Eds.), Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 2016, pp. 855–864.

[19] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: large-scale information network embedding. In Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015, pages 1067–1077. ACM, 2015..

[20] Chenyi Zhuang and Qiang Ma. Dual graph convolutional networks for graph-based semi-supervised classification. In Pierre-Antoine Champin, Fabien L. Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23–27, 2018, pages 499–508. ACM, 2018..

[21] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. Graph random neural networks for semi-supervised learning on graphs. In Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual, 2020..

[22] Wu. Felix, Amauri H. Souza Jr, Tianyi Zhang, Christopher Fifty, Yu Tao, Kilian Q. Weinberger, Simplifying graph convolutional networks In Proceedings of the 36th International Conference on Machine Learning, vol. 97, Long Beach, California, USA, 2019, pp. 6861–6871.

[23] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019. OpenReview.net, 2019..

[24] Guohao Li, Matthias Müller, Ali K. Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 – November 2, 2019, pages 9266–9275. IEEE, 2019..

[25] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Annual Conference on Neural Information Processing Systems, NeurIPS 2017, December 4–9, 2017, Long Beach, CA, USA, pages 1024–1034, 2017..

[26] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, Michael M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model cnns, in: In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, IEEE Computer Society, 2017, pp. 5425–5434.

[27] Yiding Yang, Xinchao Wang, Mingli Song, Junsong Yuan, Dacheng Tao, SPAGAN: shortest path graph attention network, in: In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019 ijcai.org, August 10–16, 2019, Macao, China, 2019, pp. 4099–4105.

[28] Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. Direct multi-hop attention based graph neural network. arXiv preprint arXiv:2009.14332, 2020..

[29] Dongkwan Kim, Oh. Alice, How to find your friendly neighborhood: Graph attention design with self-supervision, in: In International Conference on Learning Representations, ICLR, 2021.

[30] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2–7, 2018, pages 3546–3553. AAAI Press, 2018..

[31] 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, Long Beach, CA, USA, June 16–20, 2019, pages 11313–11320. Computer Vision Foundation/ IEEE, 2019..
[32] Yu Chen, Lingfei Wu, and Mohammed J. Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual, 2020..
[33] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, Jiliang Tang, Graph structure learning for robust graph neural networks, in: In KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event ACM, August 23–27, 2020, CA, USA, 2020, pp. 66–74.
[34] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. In Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6–12, 2020, virtual, 2020..
[35] Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. Node similarity preserving graph convolutional networks. In Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pages 148–156, 2021..
[36] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, The pagerank citation ranking: Bringing order to the web, Stanford InfoLab, 1999, Technical report.
[37] Risi Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In Proceedings of the Nineteenth International Conference on Machine Learning, ICML 2002, University of New South Wales, Sydney, Australia, July 8–12, 2002, pages 315–322. Morgan Kaufmann, 2002..
[38] Peter D. Turney, Patrick Pantel, From frequency to meaning: Vector space models of semantics, J. Artif. Intell. Res. 37 (2010) 141–188.
[39] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In Annual Conference on Neural Information Processing Systems, NeurIPS 2014, December 8–13 2014, Montreal, Quebec, Canada, pages 2177–2185, 2014..
[40] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, Tina Eliassi-Rad, Collective classification in network data, AI Magazine 29 (3) (2008) 93.
[41] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. arXiv preprint arXiv:1811.05868, 2018..
[42] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19–24, 2016, volume 48, pages 40–48. JMLR.org, 2016..
[43] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In Annual Conference on Neural Information Processing Systems, NeurIPS 2016, December 5–10, 2016, Barcelona, Spain, pages 3837–3845, 2016..
[44] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 249–256. JMLR Workshop and Conference Proceedings, 2010..
[45] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings, 2015..
[46] Laurens Van der Maaten, Geoffrey Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (11) (2008).
[47] Zhewei Wei, Xiaodong He, Xiaokui Xiao, Sibo Wang, Shuo Shang, and Ji-Rong Wen. Topppr: top-k personalized pagerank queries with precision guarantees on large graphs. In Proceedings of the 2018 International Conference on Management of Data, pages 441–456, 2018..
[48] Aleksandar Bojchevski, Johannes Klicpera, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. Scaling graph neural networks with approximate pagerank. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2464–2473, 2020..

Jinliang Yuan received the B.S degree and master's degree from the School of Information Science and Engineering, Lanzhou University, in 2016 and 2019, where he is currently pursuing the Doctor's degree with the Department of Computer Science and Technology, Nanjing University. His research interests include graph neural networks, data mining and complex networks.

Meng Cao received the B.S. and Master's degrees from the School of Electronic Science and Engineering, Nanjing University, in 2014 and 2017. She is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Nanjing University. Her research interests include social network analysis, data mining, and graph neural networks.

Hao Cheng received the B.S. degree from the School of Chemistry and Chemical Engineering, Nanjing University in 2017. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Nanjing University. His research interests include multi-agent system, game theory and social network.

Hualei Yu received the Master's degrees from the School of Software Engineering, Xi'an Jiao tong University, in 2019. She is currently a Ph.D. candidate at the Department of Computer Science and Technology, Nanjing University. Her research interests include data mining, and graph neural networks.

Junyuan Xie is currently a Professor with the Department of Computer Science and Technology, Nanjing University. He has vast research interests in social network analysis, and multi-agent systems. Until now, he has published more than 100 papers in conferences and journals. Now, his research is sponsored by the National Key Research and Development Program of China, the National Natural Science Foundation of China, the Key Research and Development Program of Jiangsu, and the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

Chongjun Wang received the Ph.D. degree from the Department of Computer Science and Technology, Nanjing University. He is currently a Professor with the Department of Computer Science and Technology, Nanjing University. He has vast research interests in machine learning, data mining, social network analysis, and multi-agent systems. Until now, he has published more than 100 papers in conferences and journals. Now, his research is sponsored by the National Key Research and Development Program of China, the National Natural Science Foundation of China, the Key Research and Development Program of Jiangsu, and the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.