Topology Optimization based Graph Convolutional Network

Liang Yang[1,2,3] , **Zesheng Kang**[1] , **Xiaochun Cao**[2] , **Di Jin**[4] , **Bo Yang**[5,3] and **Yuanfang Guo**[6,*]
[1]School of Artificial Intelligence, Hebei University of Technology, China
[2]State Key Laboratory of Information Security, Institute of Information Engineering, CAS, China
[3]Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, China
[4]College of Intelligence and Computing, Tianjin University, China
[5]College of Computer Science and Technology, Jilin University, China
[6]School of Computer Science and Engineering, Beihang University, China
yangliang@vip.qq.com, andyguo@buaa.edu.cn

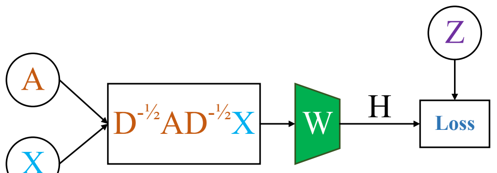
Abstract

In the past few years, semi-supervised node classification in attributed network has been developed rapidly. Inspired by the success of deep learning, researchers adopt the convolutional neural network to develop the Graph Convolutional Networks (GCN), and they have achieved surprising classification accuracy by considering the topological information and employing the fully connected network (FCN). However, the given network topology may also induce a performance degradation if it is directly employed in classification, because it may possess high sparsity and certain noises. Besides, the lack of learnable filters in GCN also limits the performance. In this paper, we propose a novel Topology Optimization based Graph Convolutional Networks (TO-GCN) to fully utilize the potential information by jointly refining the network topology and learning the parameters of the FCN. According to our derivations, TO-GCN is more flexible than GCN, in which the filters are fixed and only the classifier can be updated during the learning process. Extensive experiments on real attributed networks demonstrate the superiority of the proposed TO-GCN against the state-of-the-art approaches.

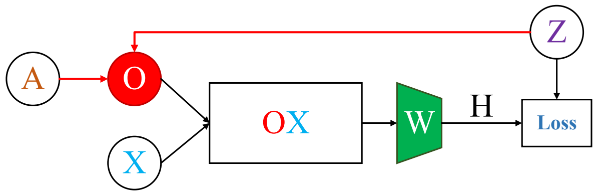
1 Introduction

Objects in real world, which is usually modeled by nodes in typical graphs/networks, are likely to be connected in various approaches. For example, people are connected through social network and online social networks. Computers are connected via intranets and Internet. Proteins are connected by the existence of electrostatic forces. To explore the characteristics and mechanisms of these networks, network analysis is introduced [Newman, 2003]. In network analysis, node classification, which is also named as network partition, community detection and graph clustering, is one of the most widely studied topic [Fortunato, 2010; Fortunato and Hric, 2016; Tao *et al.*, 2017].

[*]Corresponding author.



(A) Graph Convolutional Networks



(B) Topology Optimization based Graph Convolutional Networks

Figure 1: The difference between Graph Convolutional Networks (GCN) and our proposed TO-GCN. Learnable variables are shown with a background color. GCN takes the given network topology A and node feature X as input and only learns the parameters of FCN W with label Z. TO-GCN jointly learns the network topology O (red line and disk) and the parameters of the FCN W to fully explore the labels information Z.

Classical node classification is developed based on the utilization of the network topology. Although they contribute significantly in understanding the characteristics and mechanisms of the networks, their performances are typically unsatisfactory due to the sparsity properties of the networks and the noises in the networks. To improve the performances, numerous side information is adopted. Among them, label and content information are the most commonly utilized [Tu *et al.*, 2016; Yang *et al.*, 2013]. By effectively exploiting the labels and network topology, label propagation [Zhu *et al.*, 2003] achieves a great performance boost. Many algorithms are later proposed to utilize the pairwise constraint information, which is another kind of supervised information in the form of must-link and cannot-link [Lu and Peng, 2013]. By exploring the correlations between node feature and network topology structure, node classification task in attributed networks has achieved great success and performance improvement [Wang *et al.*, 2016].

Recently, many efforts have been made to resolve the task of semi-supervised node classification in attributed networks, which predicts the labels by exploiting the network topology, node features and labels. Motivated by the success of deep learning [Goodfellow *et al.*, 2016], especially the convolutional neural networks (CNNs) [Krizhevsky *et al.*, 2012], researchers extend CNNs to process the irregular graph data, e.g., graphs and manifolds. Among these extensions, Graph Convolutional Networks (GCN), which simplifies ChebNet [Defferrard *et al.*, 2016], has attracted a large amount of attentions due to its simplicity and high performance [Kipf and Welling, 2017]. As shown in Figure 1(A), GCN is equivalent to smoothing the node features in the neighbourhoods and process them with a fully connected network (FCN). The success of GCN is yielded by the network topology, which is shown in [Li *et al.*, 2018], and the label information, which is only employed to train the parameters in FCN.

Unfortunately, GCN has not fully exploited the potential of the network topology and the flexibility of the FCN is also limited. Specifically, the given network topology is not optimal due to certain sparsity and noises. Some nodes in one class may not be close, while other nodes belonging to different classes may be directly connected. These phenomenons have not been taken into considerations by GCN. On the other hand, the flexibility of GCN is limited compared to CNN. Typically, CNN consists of two learnable components, convolutional layers (including pooling layers) and fully connected layers. The former one can be regarded as the feature extractor with learnable filters, while the latter one is equivalent to a learnable classifier. On the contrary, the filters A in GCN are fixed and only the classifier is learnable (Figure 1(A)).

To better utilize the network topology via refinement and improve the flexibility of the network, we propose a novel Topology Optimization based Graph Convolutional Networks (TO-GCN). As shown in Figure 1(B), the given labels are utilized to simultaneously and jointly learn the network topology and the parameters of the FCN, which provides more flexibility compared to GCN.

Specifically, with the given labels being the pairwise constraints (must-link and cannot-link), the network topology is refined to satisfy the pairwise constraints and predict the unknown labels according to the given labels. To measure the consistencies between the learned network topology and pairwise constraints, constraint propagation is formulated as a minimization of an objective function with the refined topology being the parameters. This model is optimized with respect to the refined topology and the parameters of FCN. Analysis of the joint optimization algorithm indicates that the learned topology is affected by both the given constraints and classification results.

Our main contributions are summarized as follows:

- We analyze the impact of network topology to the performance of semi-supervised node classification in attributed networks and demonstrate that the label information has not been fully explored in most of the existing methods.
- We propose a novel Topology Optimization based Graph Convolutional Networks (TO-GCN), which jointly learns the network topology and the parameters of the FCN with respect to the given labels.
- We provide an analysis of the joint optimization algorithm to demonstrate its superiority compared to the straightforward separate optimization approach.

2 Backgrounds and Motivations

2.1 Notations and Definitions

Define an attributed network as $G = (V, E, X)$ with nodes/vertices $V = \{v_1, v_2, ..., v_N\}$ and edges $E = \{e_1, e_2, ..., e_M\}$. The attributes of all the vertices are represented as an attribute matrix $X \in \mathbb{R}^{N\times F}$, where the n^{th} row of which, $x_n \in \mathbb{R}^{1\times F}$, corresponds to the attributes of vertex v_n in the form of a F-dimensional vector. The network topology is represented by an adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{N\times N}$. The elements in the adjacency matrix possess binary value, i.e., $a_{ij} = 1$, if an edge exists between the vertices v_i and v_j, and vice versa. $D = \text{diag}(d_1, d_2, ..., d_N)$, where $d_n = \sum_j a_{nj}$ is the degree of vertex v_n, is the degree matrix.

In general, semi-supervised node classification predicts the labels $Y \in \{0, 1\}^{N\times C}$ of the vertices in V, given the network $G = (V, E, X)$ and some labelled nodes in set V_l in the form of $Z = [Z_{nc}] \in \{0, 1\}^{N\times C}$, where C is the number of classes and $Z_{nc} = 1$ if and only if the vertex v_n belongs to c^{th} class.

2.2 Label Propagation

The semi-supervised node classification originates from graph-based semi-supervised learning, which makes full use of the limited number of labels by exploring the graph structure. The philosophy behind them is the assumption that nearby vertices on a graph tend to share the same label. Thus, they can be formulated as minimizing the objective function

$$\mathcal{L}(Y) = \frac{1}{2}\sum\nolimits_{i,j} w_{ij}(y_i - y_j)^2, \qquad (1)$$

where w_{ij} is the similarity between the vertices v_i and v_j.

Unfortunately, most of the graph-based semi-supervised learning algorithms ignore either network topology or node feature. When the node features are neglected, some methods like LPA [Raghavan *et al.*, 2007] only exploit the label propagation on the network (topology information), i.e., $w_{ij} = a_{ij}$, which is an element in the adjacency matrix. Other methods propagate the labels in similarity graph, which is constructed from the node features without taking the topological information into consideration [Zhu *et al.*, 2003]. In the similarity graph, $w_{ij} = \exp\left(||x_i - x_j||_2^2/\sigma^2\right)$ is the similarity calculated between the vertices v_i and v_j with reference to their features. The process of the label propagation in graph can be seen as refining the graph structure with the given labels.

2.3 Graph Convolutional Neural Networks

Recently, semi-supervised node classification algorithms tend to improve the performance by jointly considering the network topology and node attributes (graph data) due to their necessity. Inspired by the successful applications of deep learning to the regular grid data (e.g. images and videos), researchers consider to adopt the deep learning techniques to process the irregular graph data (e.g. graphs or manifolds) [Defferrard *et al.*, 2016; Duvenaud *et al.*, 2015; Niepert *et al.*, 2016; Hamilton *et al.*, 2017; Scarselli *et al.*, 2009; Xu *et al.*, 2019]. To alleviate this difficulty, spectral approaches apply the convolution operation directly to the spectrum of the graph (i.e., the singular values of graph Laplacian) by treating the node attributes as signals in graph according to the spectral graph theory.

$$g_\theta * x = U g_\theta(\Lambda) U^T x, \tag{2}$$

where U and Λ represent the singular vectors and singular values of the graph Laplacian $L = D^{-1/2}(D - A)D^{-1/2}$, i.e., $L = U\Lambda U^T$, respectively.

Unfortunately, the high computational complexity of singular value decomposition (SVD) prevents the spectral approaches from being applied to large graphs. To overcome this deficit, may simplifications have been proposed. ChebNet [Defferrard *et al.*, 2016] approximates the spectral filter with P^{th} order Chebyshev polynomials as $g_\theta * x = \sum_{p=0}^{P} \theta'_p T_p(L)x$, where T_p and θ_p are the Chebyshev polynomials and coefficients, respectively. GCN (Kipf and Welling 2017) further simplifies ChebNet as

$$g_\theta * x = \theta(I + D^{-1/2}AD^{-1/2})x, \tag{3}$$

by constraining the Chebyshev polynomials with 1^{st} order and the largest singular values with 2, where I denotes the identity matrix. By denoting $\tilde{A} = A + I$ and $\tilde{D}_{nn} = \sum_j \tilde{A}_{nj}$ and generalizing one input channel x and one spectral filters θ to F input channels X and C spectral filters $\Theta \in \mathbb{R}^{F\times C}$, GCN becomes

$$H = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\Theta), \tag{4}$$

where $\sigma(.)$ is the nonlinear activation function, such as softmax or ReLU, as shown in Figure 1(A). According to the experiments, stacking two GCNs, which is shown in Eq. (5), will provide the best performance.

$$Z = f(X, A) = \text{softmax}\left(\hat{A}\,\text{ReLU}(\hat{A}X\Theta^{(0)})\Theta^{(1)}\right), \tag{5}$$

where $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$. The parameters $\Theta^{(0)}$ and $\Theta^{(1)}$ can be calculated by minimizing the cross-entropy errors of the labeled nodes

$$\mathcal{L} = -\sum_{n\in V_l}\sum_{c=1}^{C} Y_{nc}\log(Z_{nc}), \tag{6}$$

where V_l denotes the set of labelled nodes. Recent literature [Li *et al.*, 2018] indicates that the graph convolution operation $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X$ in GCN is equivalent to a Laplacian smoothing operation applied to the local neighbourhood, i.e., the graph convolution operation is essentially averaging the node attributes in a local neighbourhood with weight $\frac{1}{\sqrt{d_i d_j}}$, where d_i is the degree of node v_i.

Graph attention network (GAT) [Veličković *et al.*, 2018] extends GCN by imposing the attention mechanism [Bahdanau *et al.*, 2015] to the neighbouring weight assignment and formulates the weight between vertices v_n and v_k as

$$O_{ij} = \frac{\exp\left(c(x_i^T\Theta, x_j^T\Theta)\right)}{\sum_{k\in N(j)}\exp\left(c(x_k^T\Theta, x_j^T\Theta)\right)}, \tag{7}$$

Dataset	FCN	GCN	GCN-GT
Citeseer	57.1%	72.0%	100%
Cora	56.2%	81.3%	100%
PubMed	70.7%	79.2%	100%

Table 1: Classification results with different topologies.

where $c(x, y)$ is the self-attention with a shared attentional mechanism $c : \mathbb{R}^C \times \mathbb{R}^C \to \mathbb{R}$. As can be observed, GAT is equivalent to estimating the edge weights according to the attribute similarities between the connected nodes.

Mixture model networks (MoNet) [Monti *et al.*, 2017] unifies some convolutional operations on non-Euclidean structured data (graph or manifold) as mixture CNNs.

2.4 Motivation

After the review of the existing schemes, the main drawbacks of these methods, label propagation (LP) and graph convolutional networks (GCN), are summarized that three kinds of information are not fully utilized by the existing approaches.

LP predicts the labels by combining either the network topology or node features with the given labelled nodes. Obviously, LP approaches have not fully explored all the available information, including the network topology, node features and given labels.

GCN in Eq. (4), which is represented in Figure 1(A), can be rewritten as

$$H = \sigma(\underbrace{(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}})}_{\text{First Term}}\ \underbrace{(X\Theta)}_{\text{Second Term}}\ ), \tag{8}$$

where the first term is the given graph Laplacian without the parameters, and the second term is the fully connected network (FCN) which directly employs the node features as inputs. [Li *et al.*, 2018] claims that the classification performance is significantly improved compared to GCN without the Laplacian smoothing term (first term). Therefore, the given network topology (for Laplacian smoothing) plays a very important roles in the classification task.

Based on the above observation in [Li *et al.*, 2018], we further obtain that "A clearer network (community) structure will improve the performance of semi-supervised node classification". The presence of community (modular/cluster) structures, in which the nodes are densely connected within the communities and seldomly connected across the communities, is a common property of networks. For better illustration, we conduct experiments on three attributed networks, Cora, Citeseer and PubMed, with the same attribute matrix and labelled nodes yet three different network topologies.

- FCN only utilize the node features without the graph convolution operation;
- GCN takes the given network topology A as input;
- GCN-GT employs the ground truth membership matrix $G = [g_{ij}] \in \{0, 1\}^{N\times N}$, where $g_{ij} = 1$ if and only if the vertices v_i and v_j are in the same class.

It is obvious that the community structure of GCN-GT is clearer than that of GCN, and the community structure of GCN is clearer than that of FCN.

As can be observed in Table 1, the performance is improved when the network structure becomes clear. This certainly reveals that a refined network topology will benefit the classification performance. However, the network topology is fixed in GCN, which limits the flexibility of the network. Some previous work have shown that the network structure can be improved with the labels [Zhu *et al.*, 2003; Lu and Peng, 2013]. However, the given label information has not been properly integrated with the network topology in GCN. Therefore, GCN has not fully utilized the given label information.

3 Topology Optimization based Graph Convolutional Networks

The existing work reviewed in Section 2 motivates us to refine the network topology. Although there exists previous methods, which refine the network topology with the given labels, these label information has not been fully exploited. Therefore, in this paper, we consider to utilize the given labels to simultaneously and jointly refine the network topology and learn the parameters of the FCN. The flow chart of our proposed method is shown in Figure 1(B). In this section, our network topology refinement method is firstly introduced. Then, the proposed Topology Optimization based Graph Convolutional Networks (TO-GCN) is presented, followed by the optimization algorithm. Finally, the analysis of TO-GCN is given from the model and optimization perspectives.

3.1 Network Topology Refinement

To refine the network topology with the given labels and maintain the topology to be non-negative, the refinement is modeled as a label propagation process. With the assumption that the nearby vertices in a graph tend to share the same label, Eq. (1) can be reformed to

$$\min_{Y} \frac{1}{2}\sum_{i,j} a_{ij}||y_i - y_j||_2^2 = \min_{Y} \frac{1}{2}Tr(Y^T LY)$$
$$s.t.\, y_n = z_n \,\forall\, v_n \in V_l, \tag{9}$$

where V_l stands for the set of labelled vertices, $z_n \in \{0,1\}^{1\times K}$ and $y_n \in \{0,1\}^{1\times K}$ are the ground-truth and predicted labels of the vertex v_n. The n^{th} rows of Y and Z are y_n^T and z_n^T, respectively. $L = D - A$ is the graph Laplacian of network with the adjacency matrix A. By relaxing the elements in Y to be in $\mathbb{R}^{N\times K}$, the constrained integer optimization in Eq. (9) can be relaxed to

$$\min_{Y} \frac{1}{2}Tr(Y^T LY) + \frac{\lambda}{2}||Y - Z||_F^2. \tag{10}$$

Its optimal Y^* is also the optimal solution to the following problem according to [Wang *et al.*, 2012]

$$\min_{Y} \frac{1}{2}Tr(LYY^T) + \frac{\lambda}{2}||YY^T - ZZ^T||_F^2. \tag{11}$$

By denoting $Q = [Q_{ij}] = ZZ^T \in \{0,1\}^{N\times N}$, Q_{ij} equals to 1 (must-link constraint) if and only if the vertices v_i and v_j belong to the same class based on the given labels Z. Since the refinement is achieved with the given label information instead of directly obtaining the node label Y, we let $O = [O_{ij}] = YY^T \in \mathbb{R}^{N\times N}$ as the refined network topology and rewritten Eq. (11) as

$$\min_{O} \frac{1}{2}Tr(LO) + \frac{\lambda}{2}||O - Q||_F^2. \tag{12}$$

The must-link constraints (the pairs of nodes belong to the same class) are preserved during the transformation from Eq. (10) to Eq. (11). However, the cannot-link constraints (the pairs of nodes belong to different classes), which reduce the similarity (edge weight) between the nodes belonging to different classes, are ignored during the transformation. Therefore, Eq. (12) is revised to penalize the high similarities between the nodes from different classes as

$$\min_{O} \frac{1}{2}Tr(LO) + \frac{\lambda}{2}||O - Q||_F^2 + \frac{\alpha}{2}\sum_{i,j} O_{ij}||z_i - z_j||_2^2.$$

Let $G = [G_{ij}] \in \{0,1\}^{N\times N}$, where $G_{ij} = 1$ if and only if the vertices v_i and v_j belong to different classes. Then, the objective function of our network topology refinement method becomes

$$\min_{O} \frac{1}{2}Tr((L + \alpha G)O) + \lambda||O - Q||_F^2, \tag{13}$$

where Q and G are the pairwise constraint matrices and can be obtained from the given label matrix Z. By minimizing Eq. (13), the given topology A is refined to O with the help of the given labels.

3.2 Topology Optimization based Graph Convolutional Networks

To exploit the network topology refinement in GCN, a straightforward approach, which is named P-GCN, is to simply perform the refinement in Eq. (13) at first, and then perform GCN on the refined network topology.

However, this approach is sub-optimal, because the pre-processed refined topology only facilitates the semi-supervised node classification by providing a high modular network structure while ignoring its direct impact to the classification results. Detailed analysis is provided at the end of this section.

Therefore, Topology Optimization based Graph Convolutional Networks (TO-GCN) is proposed to jointly learn the network topology and the parameters of a fully connected network (FCN) as shown in Figure 1(B). Similar to GCN, the cross-entropy loss in Eq. (6) is adopted to measure the classification errors, and the final objective function is

$$\begin{aligned}&\min_{W,O} \mathcal{L}_{classify} + \mathcal{L}_{refine}\\ &= \min_{W,O} -\sum_{n\in V_l}\sum_{c=1}^{C} Z_{nc}\log(H_{nc}) + Tr\left((L + \alpha G)O\right) +\\ &\frac{\lambda}{2}||O - Q||_F^2,\end{aligned} \tag{14}$$

where the first term is the classification loss and the second is the topology refinement loss. $H = [H_{nc}] \in \mathbb{R}^{N\times C}$ is the prediction matrix and

$$H = f(X, O) = \operatorname{softmax}\left(O \operatorname{ReLU}\left(OXW^{(0)}\right) W^{(1)}\right),$$

where $W = \{W^{(0)}, W^{(1)}\}$ is the parameters of FCN.

3.3 Optimization

To jointly learn the refined network topology O and FCN parameters W, we alternatively update one at a time while fix the other. When the refined network topology O is fixed, the updating of the FCN parameters W is identical to the updating procedures in GCN, and we thus omit the details of it. For convenience, the process of updating O with fixed W is described in details.

Although the second and third terms in Eq. (14) both possess analytical solutions, the gradient decent approach is still adopted to optimize Eq. (14), because the first term does not have a close-form solution. Since the Laplacian matrix L and the constraints matrices G and Q are symmetric, the gradient of $\mathcal{L}_{refine}$ with respect to O is

$$\frac{\partial \mathcal{L}_{refine}}{\partial O} = (L + \alpha G - \lambda Q) + \lambda O, \tag{15}$$

where $L + \alpha G - \lambda Q$ can be computed in advance, since it does not vary in each iteration.

Then, we compute the gradient of $\mathcal{L}_{classify}$ with respect to O. For simplicity, we consider the one-layer convolution operation i.e., $H = \operatorname{softmax}(OXW)$. Note that the prediction of the node v_n, $H_n = \operatorname{softmax}(O_n B)$, will only be affected by O_n, the n^{th} row of the topology matrix, where $B = XW$ is fixed when W is fixed. Thus, the gradient of $\mathcal{L}_{classify}$ with respect to O_{nk} is equivalent to that of $\mathcal{L}_{classify}(n) = \left(-\sum_{c=1}^{C} Z_{nc} \log(H_{nc})\right)$ with respect to O_{nk}, as shown in Eq. (14). According to the Chain Rule,

$$\frac{\partial \mathcal{L}_{refine}(n)}{\partial O_{nk}} = \sum_{p} \frac{\partial \mathcal{L}_{refine}(n)}{\partial T_{np}} \frac{\partial T_{np}}{\partial O_{nk}},$$

$$\frac{\partial \mathcal{L}_{refine}(n)}{\partial T_{np}} = -\sum_{c=1}^{C} Z_{nc} \frac{\partial \log(H_{nc})}{\partial T_{np}},$$

where $H_{nc} = \operatorname{softmax}(T_{nc})$ and $T_{nc} = \sum_k O_{nk} B_{kc}$. Since

$$\sum_{c=1}^{C} Z_{nc} = 1 \quad \text{and} \quad \frac{\partial \log(H_{nc})}{\partial T_{np}} = \delta_{cp} - H_{np},$$

where δ_{cp} is the Kronecker delta and $\delta_{cp} = 1$ if and only if $c = p$, then

$$\frac{\partial \mathcal{L}_{refine}(n)}{\partial T_{np}} = -\sum_{c=1}^{C} Z_{nc} (\delta_{cp} - H_{np}) = H_{np} - Z_{np}.$$

Besides, since $\frac{\partial T_{np}}{\partial O_{nk}} = B_{kp}$, then,

$$\frac{\partial \mathcal{L}_{refine}(n)}{\partial O_{nk}} = \sum_{p} (H_{np} - Z_{np}) B_{kp},$$

and it can be transformed to the matrix form

$$\frac{\partial \mathcal{L}_{refine}}{\partial O} = (H - Z)B^T = (H - Z)W^T X^T, \tag{16}$$

since $B = XW$. By combining Eqs. (15) and (16), the total gradient in the matrix form is

$$\frac{\partial \mathcal{L}}{\partial O} = (H - Z)W^T X^T + (L + \alpha G - \lambda Q) + \lambda O. \tag{17}$$

Tips for Implementation: To speedup the optimization, the refined network topology O is firstly initialized by minimizing Eq. (13) without considering the backward-propagation of the classification errors. Besides, the updating rule in Eq. (17) is only applied to update the weights of the existing edges and the must-link constraints, i.e., $A + Q$. Thus, the complexity of TO-GCN is $O(M)$ instead of $O(N^2)$, where N and M are the numbers of nodes and edges, respectively.

Remark: The extra cost of the learnable graph is the topology updating in Eq. (17). For efficiency, the topology is initialized via constrained propagation, as shown in Eq. (13), and the updating in Eq. (17) is only applied to update the weights of the existing edges and the must-link constraints. Therefore, the increased computations are linearly proportional to the number of edges. and the proposed TO-GCN can be applied to large-scale networks.

3.4 Insights

Here, we introduce some insights of the proposed method to further illustrate the superiority of TO-GCN and the advantage of jointly learning the refined network topology and parameters of the FCN.

From the Model Perspective

To elaborate the advantage of our proposed TO-GCN against GCN, they are both compared to the Convolution Neural Network (CNN). CNN consists of two learnable components: convolutional layers (including pooling layers) and fully connected layers. The former layers can be regarded as the feature extractor with learnable filters, while the latter layers are equivalent to a learnable classifier. In our proposed TO-GCN (Figure 1(B)), the learnable network topology O can be regarded as the learnable filters while FCN with parameters W is equivalent to the learnable classifier. Therefore, TO-GCN and CNN are similar. On the contrary, the filters A is fixed and only the classifier is learnable in GCN (Figure 1(A)). Therefore, TO-GCN is more flexible than GCN.

From the Optimization Perspective

The updating rule of O in Eq. (17) consists of three terms whose roles are elaborated in details as follows.

The first term is the main difference between the straightforward approach and the joint learning mechanism. $S = [S_{nk}] = (H - Z)W^T X^T$ can be regarded as the correlation between the prediction error $Z - H$ and the prediction from node attributes XW. S_{nk} can then be written as

$$S_{nk} = (h_n - z_n)(x_k W)^T, \tag{18}$$

where h_n and z_n represent the predicted and ground-truth label vectors of the vertex v_n, respectively.

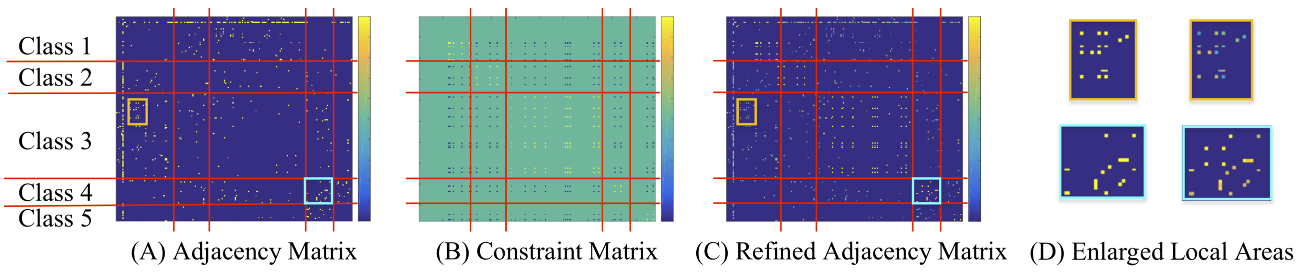


Figure 2: Visualizations of the learning process on the Cornell network. Three subfigures are the adjacency matrix, constraint matrix constructed from 10% labelled data, and learned adjacency matrix, respectively. The cyan boxes contain the intra-class edges while the orange boxes contain the inter-class edges. The color brightness indicate the weights of edges.

Then, we analyze the role of S_{nk}. If the ground truth and the predicted label of node v_n are identical, then $S_{nk} = 0$. Otherwise, if the ground truth and the predicted label of node v_n are i and j, respectively, then

$$h_n - z_n = (0, ..., 0, \underbrace{1}_{j}, 0, ..., 0, \underbrace{-1}_{i}, 0, ..., 0). \tag{19}$$

S_{nk} can be rewritten as $S_{nk} = -x_k(w_{.i} - w_{.j})$, where $w_{.j}$ is the j^{th} column of W. Since $x_k w_{.j}$ is the confidence of node v_k being classified into class j, $x_k(w_{.j} - w_{.i})$ is the confidence difference of node v_k being classified into classes i and j based on the node attributes x_k. Therefore, updating O with $S = (H - Z)W^T X^T$ reduces the weights O_{nk} if the classification result of v_k makes v_n being misclassified.

The second term is the combination of the graph Laplacian and constraint matrix, which reveals the impact of the constraints to the refined network. The third term is to guarantee the updating process to be performed regularly.

Overall, by jointly learning the refined network topology and the parameters of FCN, the network topology is refined by both the given constraints and classification results. Therefore, TO-GCN is better than the straightforward P-GCN.

4 Evaluations

In this section, we evaluate the proposed TO-GCN with extensive experiments on real attributed networks and provide some visualizations for better illustrations.

4.1 Datasets and Setup

As in previous work, three citation networks, in which the vertices and edges are documents and undirected citations respectively, are utilized for performance evaluation. Node attributes are the bag-of-words representations of documents. Each node is assigned a label based on its discipline. The statistics of the networks, Cora, CiteSeer and PubMed, are summarized in Table 2. Following the setup in [Yang *et al.*, 2016] and [Kipf and Welling, 2017], the network topology, the features of all the nodes and the labels of 20 nodes per class are available for training. The performance of all the baseline methods are evaluated on 1,000 test nodes with 500 additional nodes for validation. Note that the balancing parameter between $L_{classify}$ and L_{refine} are not employed, which is equivalent to set the balancing parameter to be 1. α and λ, which are the hyperparameters for cannot-link and must-link constraints, are set according to the results of the validation set.

Dataset	Nodes	Edges	Classes	Features
Citeseer	3,327	4,732	6	3,703
Cora	2,708	5,429	7	1,433
PubMed	19,717	44,338	3	500

Table 2: Datasets.

4.2 Baselines

Our proposed model are compared to 10 state-of-the-art approaches, including multilayer perceptron (MLP), label propagation (LP) [Zhu *et al.*, 2003], semi-supervised embedding (SemiEmb) [Weston *et al.*, 2012], manifold regularization (ManiReg) [Belkin *et al.*, 2006], graph embedding (DeepWalk) [Perozzi *et al.*, 2014], iterative classification algorithm (ICA) [Lu and Getoor, 2003], graph-based semi-supervised learning framework (Planetoid) [Yang *et al.*, 2016], graph convolution with Chebyshev filters [Defferrard *et al.*, 2016], graph convolutional networks (GCN) [Kipf and Welling, 2017], and mixture model networks (MoNet) [Monti *et al.*, 2017]. All the results of the baseline approaches are either from their original papers or obtained with their original codes and default settings. To show the superiority of the joint learning strategy, TO-GCN is also compared to the straightforward version, P-GCN, where the refined topology matrix is pre-computed and fed into GCN.

4.3 Results Analysis

As shown in Table 3, the results of semi-supervised node classification indicate that the proposed TO-GCN consistently outperforms other state-of-the-art methods and the straightforward P-GCN. The superior performance of TO-GCN demonstrates that both the proposed topology refinement and the jointly learning mechanism are vital to the performance improvements. As can be observed, the performance improvement for the Pubmed network is limited, because the network topology has not been appropriately refined due to the messy structure of Pubmed and the small dimensions of the features. The former aspect causes the constraint propagation to be inefficient, while the latter one affects the learnable weights obtained from the backward propagated classification error. To demonstrate the robustness of

Methods	Cora	Citeseer	Pubmed
MLP	55.1%	46.5%	71.4%
ManiReg	59.5%	60.1%	70.7%
SemiEmb	59.0%	59.6%	71.7%
LP	68.0%	45.3%	63.0%
DeepWalk	67.2%	43.2%	65.3%
ICA	75.1%	69.1%	73.9%
Planetoid	75.7%	64.7%	77.2%
Chebyshev	81.2%	69.8%	74.4%
GCN	81.5%	70.3%	79.0%
MoNet	81.7%	69.9%	78.8%
P-GCN	82.4%	70.9%	79.3%
TO-GCN	**83.1%**	**72.7%**	**79.5%**

Table 3: Node classification results.

the proposed TO-GCN compared to GCN, an experiment is conducted on Cora by adding 3% noisy edges are conducted. The accuracy degradations of GCN and TO-GCN are 11.2% and 8.7%, respectively, which also verifies the superiority of TO-GCN.

4.4 Visualizations

To elaborate the results with more intuitions, we intend to present more insights of the learned adjacency matrix. Since the network is sparse, it is difficult to effectively visualize the learned adjacency matrices of the above three networks. Therefore, we conduct these visualizations on medium-sized Cornell network from the WebKB dataset. It consists of 195 connected webpages, which are classified into 5 classes. Each webpage is annotated by 1703-dimensional binary-valued word attributes. Since the dimensions of node content are much larger than the number of nodes, the dimensions of node content are reduced to 50 via PCA to prevent the overfitting problem. 10% of the nodes are randomly selected for training. The results are introduced in Figure 2. The nodes are re-ordered according to the ground-truth labels, and the color brightness indicates the weights of the edges. The structure of the learned graph is the combination of network topology and the constraints constructed form the labels. During the training process, the weights of the edges are refined by the pairwise constraints and the back-propagation of the classifier. As can be observed, the weights of inter-class edges (orange boxes) are significantly reduced while those of intra-class edges (cyan boxes) remain at a high level (Figure 2(D)).

5 Conclusions

In this paper, we observe that the network topology (both the structure and weights) contributes significantly in semi-supervised node classification. A clear community structure (modularity) of network usually induces a high classification accuracy. Besides of this observation, existing approaches have not fully utilized the potentials of three types of information, including network topology, node features and given labels. Therefore, we propose a novel Topology Optimization based Graph Convolutional Networks (TO-GCN) to fully utilize these three types of information by jointly refining the network topology and learning the parameters of FCN. Extensive experiments on real attributed networks demonstrate the superior performance of the proposed TO-GCN against the state-of-the-arts.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (No.2017YFC0820106), in part by the National Natural Science Foundation of China under Grant 61503281, Grant 61802391, Grant 61772361, Grant U1636214, Grant U1605252, in part by the Foundation for Innovative Research Groups through the National Natural Science Foundation of China under Grant 61421003, in part by the Key Program of the Chinese Academy of Sciences (No. QYZDB-SSW-JSC003), and in part by the Fundamental Research Funds for the Central Universities.

References

[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[Belkin *et al.*, 2006] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *JMLR*, 7:2399–2434, 2006.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, pages 3837–3845, 2016.

[Duvenaud *et al.*, 2015] David K. Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, pages 2224–2232, 2015.

[Fortunato and Hric, 2016] Santo Fortunato and Darko Hric. Community detection in networks: A user guide. *Physics Reports*, 659:1 – 44, 2016.

[Fortunato, 2010] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75 – 174, 2010.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[Hamilton *et al.*, 2017] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, pages 1025–1035, 2017.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[Li *et al.*, 2018] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI*, pages 3538–3545, 2018.

[Lu and Getoor, 2003] Qing Lu and Lise Getoor. Link-based classification. In *ICML*, pages 496–503, 2003.

[Lu and Peng, 2013] Zhiwu Lu and Yuxin Peng. Exhaustive and efficient constraint propagation: A graph-based learning approach and its applications. *IJCV*, 103(3):306–325, 2013.

[Monti *et al.*, 2017] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M. Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *IEEE CVPR*, pages 5425–5434, 2017.

[Newman, 2003] Mark E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[Niepert *et al.*, 2016] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *ICML*, pages 2014–2023, 2016.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *ACM SIGKDD*, pages 701–710, 2014.

[Raghavan *et al.*, 2007] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *PRE*, 76 3 Pt 2:036106, 2007.

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

[Tao *et al.*, 2017] Zhiqiang Tao, Hongfu Liu, Sheng Li, Zhengming Ding, and Yun Fu. From ensemble clustering to multi-view clustering. In *IJCAI*, pages 2843–2849, 2017.

[Tu *et al.*, 2016] Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. Max-margin deepwalk: Discriminative learning of network representation. In *IJCAI*, pages 3889–3895, 2016.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[Wang *et al.*, 2012] Xiang Wang, Buyue Qian, and Ian Davidson. Labels vs. pairwise constraints: A unified view of label propagation and constrained spectral clustering. In *IEEE ICDM*, pages 1146–1151, 2012.

[Wang *et al.*, 2016] Xiao Wang, Di Jin, Xiaochun Cao, Liang Yang, and Weixiong Zhang. Semantic community identification in large attribute networks. In *AAAI*, pages 265–271, 2016.

[Weston *et al.*, 2012] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade - Second Edition*, pages 639–655. 2012.

[Xu *et al.*, 2019] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[Yang *et al.*, 2013] Jaewon Yang, Julian J. McAuley, and Jure Leskovec. Community detection in networks with node attributes. In *IEEE ICDM*, pages 1151–1156, 2013.

[Yang *et al.*, 2016] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *ICML*, pages 40–48, 2016.

[Zhu *et al.*, 2003] Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pages 912–919, 2003.