Collaborative Graph Convolutional Networks: Unsupervised Learning Meets Semi-Supervised Learning

Binyuan Hui, Pengfei Zhu,* Qinghua Hu
College of Intelligence and Computing
Tianjin University, Tianjin, China
{huibinyuan, zhupengfei, huqinghua}@tju.edu.cn

Abstract

Graph convolutional networks (GCN) have achieved promising performance in attributed graph clustering and semi-supervised node classification because it is capable of modeling complex graphical structure, and jointly learning both features and relations of nodes. Inspired by the success of unsupervised learning in the training of deep models, we wonder whether graph-based unsupervised learning can collaboratively boost the performance of semi-supervised learning. In this paper, we propose a multi-task graph learning model, called collaborative graph convolutional networks (CGCN). CGCN is composed of an attributed graph clustering network and a semi-supervised node classification network. As Gaussian mixture models can effectively discover the inherent complex data distributions, a new end to end attributed graph clustering network is designed by combining variational graph auto-encoder with Gaussian mixture models (GMM-VGAE) rather than the classic k-means. If the pseudo-label of an unlabeled sample assigned by GMM-VGAE is consistent with the prediction of the semi-supervised GCN, it is selected to further boost the performance of semi-supervised learning with the help of the pseudo-labels. Extensive experiments on benchmark graph datasets validate the superiority of our proposed GMM-VGAE compared with the state-of-the-art attributed graph clustering networks. The performance of node classification is greatly improved by our proposed CGCN, which verifies graph-based unsupervised learning can be well exploited to enhance the performance of semi-supervised learning.

1 Introduction

The explosive growth of graph data in real-world applications, such as knowledge graph (Hamaguchi et al. 2017), social networks (Tang et al. 2008), and computer vision (Wang, Ye, and Gupta 2018), badly needs graph learning models with strong representation and reasoning abilities. In recent years, graph neural networks (Bruna et al. 2014; Henaff, Bruna, and LeCun 2015; Defferrard, Bresson, and Vandergheynst 2016; Kipf and Welling 2017) have shown surprising performance. Especially, graph convolutional networks (GCN) have been applied to a large variety of tasks, *e.g.* , attributed graph clustering and semi-supervised node classification (Kipf and Welling 2017).

Despite the success of GCN in graph data, there are still great challenges for GCN in attributed graph clustering and semi-supervised node classification. Firstly, variational graph auto-encoder (VGAE) (Kipf and Welling 2016) based attributed graph clustering approaches impose a single Gaussian prior on the learned latent embedding. However, the complex data distributions cannot be well modeled under the assumption of a single Gaussian prior. Additionally, many works conduct graph embedding and get clustering assignments by *k-means* separately, which does not utilize the end to end learning manner of deep learning. Secondly, semi-supervised learning directly uses unlabeled samples to enhance models by graph convolutions, but the supervised information is still not increased. Although a pseudo-labeling strategy is designed to exploit unlabeled samples with the most confidence scores (Li, Han, and Wu 2018), the accumulative error of pseudo-labels will gradually increase and degrade the model. Thirdly, pseudo-labels can be assigned to unlabeled samples by clustering. The potential of the pseudo-labels is not well investigated for graph convolutional networks.

Unsupervised learning can effectively boost the performance of deep models in many approaches. (Caron et al. 2018) used pseudo-labels generated by deep clustering to guide learning of visual features, which significantly improves the discrimination ability of the learned features. (Xie et al. 2019) proposed an unsupervised data augmentation strategy, which leads to substantial improvements when the labeled set is extremely small. Motivated by the success of unsupervised learning in visual feature learning and data augmentation, we investigate to use graph-based unsupervised learning, *i.e.* , attributed graph clustering, to enhance semi-supervised node classification performance. Specifically, our contributions are summarized as follows:

- We propose a multi-task graph learning model, *i.e.* , collaborative graph convolutional networks (CGCN). CGCN uses the samples whose clustering assignments and semi-supervised predictions are consistent to further boost the learning performance of semi-supervised node classification. CGCN effectively alleviates the accumulative error

*Corresponding Author

of pseudo-labels via joint training of clustering and semi-supervised node classification networks.

- An attributed graph clustering model is given, *i.e.* , variational graph auto-encoder with Gaussian mixture model (GMM-VGAE). It uses Gaussian mixture model to discover complex data distributions, which can lead to better clustering results in an end to end manner.
- Extensive experiments on graph data verify the effectiveness of our proposed GMM-VGAE for attributed graph clustering and multi-task graph learning model, *i.e.* , CGCN, for semi-supervised node classification.

2 Related Work

2.1 Graph Convolutional Networks

Recently, there have been increasing interests in deep learning approaches for graph data. The neural networks on graph data can be categorized into spatial (Atwood and Towsley 2016; Monti et al. 2017; Velickovic et al. 2017) and spectral approaches (Bruna et al. 2014; Defferrard, Bresson, and Vandergheynst 2016; Kipf and Welling 2017). For spectral approaches, by the eigen-decomposition of the graph Laplacian, the convolution operation is defined, resulting in potentially intense computations and non-spatially localized filters in the Fourier domain (Bruna et al. 2014). (Defferrard, Bresson, and Vandergheynst 2016) utilized Chebyshev expansion of the graph Laplacian to approximate the filters, removing the need of computing the eigenvectors of the Laplacian and yielding spatially localized filters. On this basis, (Kipf and Welling 2017) presented a simplified method by restricting the filters to operate in a 1-order neighborhood around each node, called graph convolutional networks (GCN), which is the focus of this paper. However, GCN has the intrinsic limitation that it cannot effectively propagate the labels to the entire graph. To address this problem, (Li, Han, and Wu 2018) proposed self-training approaches to enlarge the training dataset. Further more, (Sun, Zhu, and Lin 2019) proposed multi-stage self-training processing, but it only apply deep clustering on embedding and rely on distance measure to align, without considering using more appropriate attributed graph clustering approach. When the number of available labeled samples is limited, it is worth leveraging the potential of massive unlabeled samples to strengthen the performance of GCNs.

2.2 Attributed Graph Clustering

Attributed graph clustering is to cluster nodes by using node features and topological information of graph data (Yang et al. 2009). Generative models (Chang and Blei 2009; He et al. 2017; Bojchevski and Günnemann 2018) aim to interact between the connectivity of graph and node features. In recent years, benefiting from the development of graph convolution networks, attributed graph clustering has progressed significantly. Many algorithms employ graph auto-encoder (GAE) and variational graph auto-encoder (VGAE) (Kipf and Welling 2016) to learn representation with two GCN layers and then reconstruct the adjacency matrix. To learn a robust embedding and enforce the latent codes to match a prior distribution, two variants of adversarial approaches, adversarially regularized graph autoencoder (ARGA) and adversarially regularized variational graph autoencoder (ARVGA) are developed (Pan et al. 2018). (Wang et al. 2019) proposed an attentional embedding algorithm (DAEGC), to jointly perform graph clustering and learn graph embedding in a unified framework. To capture global cluster structure and adaptively select the appropriate order of different graphs, (Zhang et al. 2019) proposed an adaptive graph convolution method (AGC). Most VGAE-based methods adopt a two-step strategy with single Gaussian prior, that is, first learn embedding and then get cluster assignment by *k-means*. Recently (Nalisnick, Hertel, and Smyth 2016; Shu 2016; Jiang et al. 2017) introduced the VAE with Gaussian mixture model for non-graph data clustering.

3 Preliminaries and Problem Definition

Basic Notations on Graphs. Given a non-directed graph $\boldsymbol{\mathcal{G}} = (\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{X})$, where $\boldsymbol{V} = \{v_1, v_2, \ldots, v_n\}$ consists of a set of nodes with $|\boldsymbol{V}| = n$, and $\boldsymbol{E} = \{e_{ij}\}$ is a set of edges between nodes. Adjacency matrix $\boldsymbol{A} = \{a_{ij}\} \in \mathbb{R}^{n\times n}$ denotes the topological structure of graph $\boldsymbol{\mathcal{G}}$, where $a_{i,j} = 1$ if $(v_i, v_j) \in \boldsymbol{E}$, otherwise $a_{i,j} = 0$. $\boldsymbol{X}$ is an attribute feature matrix of all the nodes, *i.e.* , $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \in \mathbb{R}^{n\times d}$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ is a real-value attribute vector associated with node v_i.

Graph Convolutional Layer. Given the adjacent matrix $\boldsymbol{A}$ and the input feature $\boldsymbol{X}$, a layer-wise transformation is conducted by a spectral convolution function:

$$f_\phi\left(\boldsymbol{Z}^{(l)}, \boldsymbol{A}|\boldsymbol{W}^{(l)}\right) = \phi\left(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{Z}^{(l)}\boldsymbol{W}^{(l)}\right), \quad (1)$$

where $\boldsymbol{Z}^{(l)}$ is the l^{th} hidden layer for convolution, and $\boldsymbol{Z}^{(l+1)}$ is the output after convolution. We have $\boldsymbol{Z}^{(0)} = \boldsymbol{X} \in \mathbb{R}^{n\times d}$ (n nodes and d features). $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N$, $\tilde{\boldsymbol{D}}_{ii} = \sum_j \tilde{\boldsymbol{A}}_{ij}$ and $\boldsymbol{I}_N$ is an identity matrix. $\boldsymbol{W}^{(l)}$ denotes the layer-specific trainable weight parameters, and ϕ is an activation function , *e.g.* , $Relu(\cdot) = \max(0, \cdot)$. For more details, please refer to (Kipf and Welling 2017).

Attributed Graph Clustering. Given the graph $\boldsymbol{\mathcal{G}}$, attributed graph clustering aims to partition the nodes in $\boldsymbol{\mathcal{G}}$ into k clusters $\boldsymbol{\mathcal{G}} = \{\boldsymbol{\mathcal{G}}_1, \boldsymbol{\mathcal{G}}_2, \cdots, \boldsymbol{\mathcal{G}}_k\}$ by unsupervised learning without any labeled node.

Semi-supervised Learning. The semi-supervised node classification aims to learn a classifier from a set of N training samples $\mathcal{D}$. These samples are split into an unlabeled set $\mathcal{D}_u = \{x_i\}_{i=1}^{N_u}$ and a labeled set $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^{N_l}$. $y_i \in \{0, 1\}^C$ is the one-hot encoding label for C classes corresponding to x_i and $N = N_l + N_u$. Pseudo-label is the prediction of a supervised learner or clustering assignments in unsupervised learning. As we seek to perform pseudo-labeling for the N_u unlabel samples, we assume that a pseudo-label $\tilde{y}$ is available for these samples. We can then reformulate the problem as training using $\tilde{\mathcal{D}} = \{(x_i, \tilde{y}_i)\}_{i=1}^{N}$, where $\tilde{y} = y$ for the N_l labeled samples.

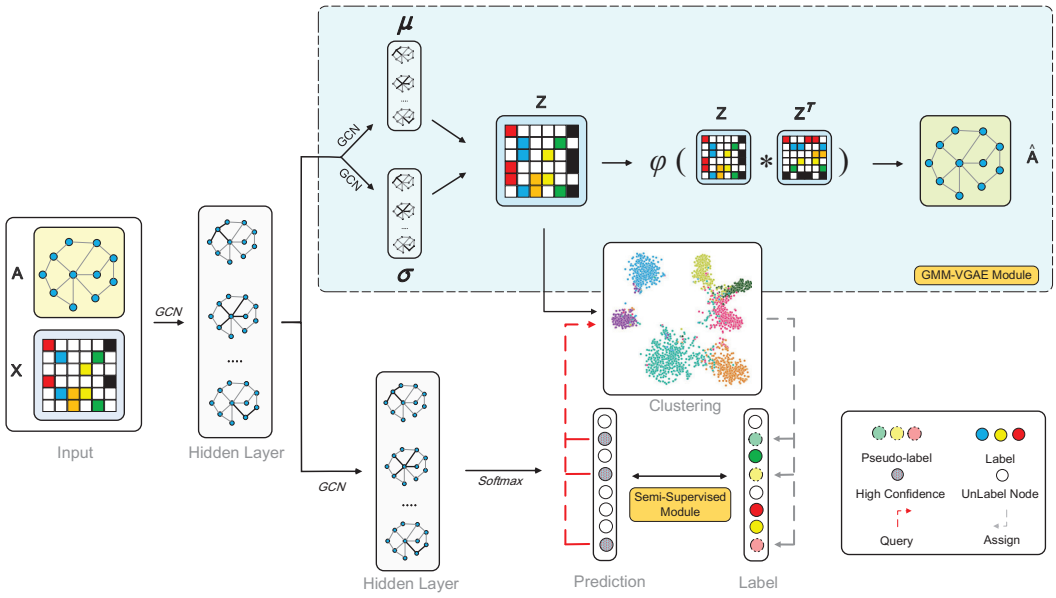

Figure 1: The architecture of collaborative graph convolutional networks (CGCN). CGCN is composed of GMM-VGAE module (attributed graph clustering) and semi-supervised module (node classification). They utilize the common shallow representation by sharing the first graph convolution layer. During the training, if the clustering assignment is consistent with the semi-supervised prediction for an unlabeled sample, then the unlabeled node with the pseudo-label is added to the label set, and the network will be retrained.

4 Proposed Method

In this section, we present the proposed multi-task graph learning model, *i.e.* , collaborative graph convolutional network (CGCN), which is shown in Fig. 1. It is pointed out that a vanilla GCN can not sufficiently propagate the label information to the entire graph with only a few labels, since repeatedly applying Laplacian smoothing may mix the features of vertices from different categories and make them indistinguishable during training (Li, Han, and Wu 2018) . Therefore GCN needs a considerable amount of labeled data for validation and model selection. An intuitive solution is to select the nodes with the most confidence scores for each class and then add them to the training set. However, the model prediction may gradually become unreliable because of the accumulative error with iterations induced by the uncertainty from pseudo-labels of the selected nodes. As shown in Fig. 2, the reliability of the selected nodes with high confidence scores will gradually decrease with iterations. Hence, these pseudo labels will probably lead the model to propagate incorrect (noise) information. Above all, the challenging issue for vanilla GCN is how to make good use of the nodes with pseudo-labels more accurately to boost the performance of GCN.

To this end, we employ unsupervised learning, *i.e.* , attributed graph clustering, to boost the performance of semi-supervised node classification. CGCN selects nodes with consistent pseudo-labels of both the unsupervised clustering network and semi-supervised node classification network. In this way, nodes can be selected more accurately and the propagation of possible incorrect information is alleviated as well. In addition, the multi-task learning model will be more beneficial to learn the general representation.

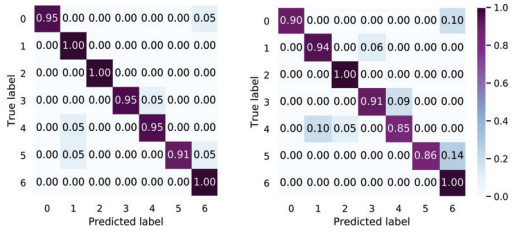

Figure 2: Confusion matrix for selected nodes with top confidence scores in different training iteration. The left subfigure shows the confuse matrix of the selected nodes in the first iteration and the right one is the selected nodes in the fifth iteration on *Cora* dataset.

4.1 GMM-VGAE

To discover the complex data distributions, we combine variational graph autoencoder with Gaussians mixture model (GMM-VGAE) and develop an end to end unsupervised model for attributed graph clustering. We first introduce the architecture of GMM-VGAE, and then present the optimization process. GMM-VGAE consists of encoder and decoder and the encoder can be constructed as:

$$\boldsymbol{Z}^{(1)} = f_{Relu}\left(\boldsymbol{X}, \boldsymbol{A}|\boldsymbol{W}^{(0)}\right), \tag{2}$$

$$\boldsymbol{Z}_{\mu}^{(2)} = f_{linear}\left(\boldsymbol{Z}^{(1)}, \boldsymbol{A}|\boldsymbol{W}_{\mu}^{(1)}\right), \tag{3}$$

$$\boldsymbol{Z}_{\sigma}^{(2)} = f_{linear}\left(\boldsymbol{Z}^{(1)}, \boldsymbol{A}|\boldsymbol{W}_{\sigma}^{(1)}\right). \tag{4}$$

Following VAGE (Kipf and Welling 2016), we can take a simple **inference model** parameterized by a two-layer GCN:

$$q(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{A}) = \prod_{i=1}^{n} q\left(z_i|\boldsymbol{X},\boldsymbol{A}\right), \tag{5}$$

$$q\left(z_i|\boldsymbol{X},\boldsymbol{A}\right) = \mathcal{N}\left(z_i|\boldsymbol{\mu}_i, diag\left(\boldsymbol{\sigma}_i^2\right)\right). \tag{6}$$

where $\boldsymbol{\mu} = \boldsymbol{Z}_{\boldsymbol{\mu}}^{(2)}$ and $\boldsymbol{\sigma}^2$ are the mean vectors and the variance. $\log \boldsymbol{\sigma}^2 = \boldsymbol{Z}_{\boldsymbol{\sigma}}^{(2)}$. The decoder is used to reconstruct the graph data, especially graph structure $\boldsymbol{A}$. More specifically, **generative model** is given by an inner product between latent variables:

$$p(\hat{\boldsymbol{A}}|\boldsymbol{Z}) = \prod_{i=1}^{n}\prod_{j=1}^{n} p\left(\hat{\boldsymbol{A}}_{ij}|z_i, z_j\right); \tag{7}$$

$$p\left(\hat{\boldsymbol{A}}_{ij}|z_i, z_j\right) = \phi\left(z_i^{\top}, z_j\right). \tag{8}$$

Here ϕ is the activation function (sigmoid), and $\hat{\boldsymbol{A}}_{ij}$ is the reconstruction graph structure from decoder.

According to VGAE (Kipf and Welling 2016), it minimizes the reconstruction error of the graph data and optimizes the variational lower bound by a single Guassian prior. However, the complex data distributions cannot be well modeled under the assumption of a single Guassian prior. Guassian mixture model has been well applied to VaDE (Jiang et al. 2017), here we extend it for attributed graph clustering task. For VaDE, the input and output are both images, while the GMM-VGAE input is a feature of the node, and the output is an adjacency matrix representing the graph structure. We describe the process of clustering: given the hyper-parameter K for cluster numbers, we first choose a cluster c from the categorical distribution $Cat(\pi)$ which is parameterized by π, and $\pi \in \mathbb{R}_{+}^{K}$, $\sum_{k=1}^{K}\pi_k = 1$ is the prior probability for cluster. Then we obtain a latent vector z from the distribution $\mathcal{N}\left(\mu_c, \sigma_c^2 I\right)$. Given a certain cluster c, we can find the Gaussian distribution with mean μ_c and variance σ_c^2 and I is an identity matrix. Finally, the observed $\mu_a = \phi\left(z_i^{\top}, z_j\right)$ can be camputed and sample a from $Ber(\mu_a)$, where the $Ber(\cdot)$ denotes the multivariate Bernoulli distribution parameterized by the latent vector μ_a from reconstruction observed sample. Above all, we can factorize joint probability $p(a, z, c)$:

$$p(a, z, c) = p(a|z)p(z|c)p(c). \tag{9}$$

Variational Lower Bound In accordance to the VAE (Kingma and Welling 2013), GMM-VGAE can be trained together and optimized by maximizing the variational evidence lower bound, *i.e.* , *elbo*. Given the generative process, the *elbo* loss $\mathcal{L}_{elbo}$ of GMM-VGAE can be defined as:

$$\mathcal{L}_{elbo} = \mathbb{E}_{q(z,c|x,a)}\left[\log\frac{p(a,z,c)}{q(z,c|x,a)}\right], \tag{10}$$

similar with VaDE, we use variational posterior $q(z, c|x, a)$ which could be assumed as a mean-field distribution to approximate the true posterior $p(z, c|x, a)$.

$$q(z, c|x, a) = q(z|x, a)q(c|x, a). \tag{11}$$

By substituting the terms in Eq. 9, 10, 11, the variational evidence lower bound $\mathcal{L}$ can be rewritten as:

$$\begin{aligned}
&\mathcal{L}_{elbo} = \mathbb{E}_{q(z,c|x,a)}\left[\log\tfrac{p(a,z,c)}{q(z,c|x,a)}\right] \\
&= \mathbb{E}_{q(z,c|x,a)}[\log p(a, z, c) - \log q(z, c|x, a)] \\
&= \mathbb{E}_{q(z,c|x,a)}[\log p(a|z) + \log p(z|c) + \log p(c) \\
&- \log q(z|x, a) - \log q(c|x, a)],
\end{aligned} \tag{12}$$

where $q(z|x, a) = \mathcal{N}\left(z; \tilde{\mu}, \tilde{\sigma}^2 I\right)$ and $\tilde{\mu}$ and $\tilde{\sigma}^2$ are derived by $\boldsymbol{Z}_{\mu}^{(2)}$, $\boldsymbol{Z}_{\sigma}^{(2)}$ respectively. In order to straightforwardly optimize Eq. 12 using standard stochastic gradient methods, we need utilize the *SGVB estimator* and *reparameterization trick* proposed by VAE (Kingma and Welling 2013), the $\mathcal{L}_{elbo}$ could be divided into reconstruction term $\mathcal{L}_{rec}$ and KL-divergence term $\mathcal{L}_{kld}$:

$$\mathcal{L}_{elbo} = \mathcal{L}_{rec} + \mathcal{L}_{kld}, \tag{13}$$

$$\mathcal{L}_{rec} = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{D} a_i \log \mu_a^n|_i + (1 - a_i)\log\left(1 - \mu_a^n|_i\right). \tag{14}$$

The $\mathcal{L}_{rec}$ is defined as binary cross entropy to calculate reconstruction loss and D is the dimensions of μ_a^n. N is the number of *Monte Carlo sample* in the *SGVB estimator* and $*^n$ is the n^{th} sample.

$$\begin{aligned}
\mathcal{L}_{kld} = &-\frac{1}{2}\sum_{c=1}^{K}\gamma_c\sum_{j=1}^{J}\left(\log\sigma_c^2|_j + \frac{\tilde{\sigma}^2|_j}{\sigma_c|_j} + \frac{\left(\tilde{\mu}|_j - \mu_c|_j\right)^2}{\sigma_c^2|_j}\right) \\
&+ \sum_{c=1}^{K}\gamma_c\log\frac{\pi_c}{\gamma_c} + \frac{1}{2}\sum_{j=1}^{J}\left(1 + \log\tilde{\sigma}^2|_j\right),
\end{aligned} \tag{15}$$

where the K is the number of clusters, π_c is the prior probability of cluster c, γ_c denotes $q(c|x, a)$, and $*|_j$ denotes the j^{th} element of $*$. $\mu_a^{(n)}$ from the decoder is computed as:

$$\mu_a^n = \phi\left((z^n)^{\top}, z^n\right), \tag{16}$$

$$z^n = \widetilde{\mu} + \widetilde{\sigma} \circ \epsilon^n, \tag{17}$$

where $\circ$ is the element-wise operator, z^n can be obtained via *reparameterization trick* and $\epsilon^n \sim \mathcal{N}(0, \mathrm{I})$. Above all, we can rewritten $\mathcal{L}_{elbo}$ as:

$$\begin{aligned}
&\mathcal{L}_{elbo} = E_{q(z,c|x,a)}\left[\log\tfrac{p(a,z,c)}{q(z,c|x,a)}\right] \\
&= \textstyle\int_z \sum_c q(z|x, a)q(c|x, a)\cdot\left[\log\frac{p(a|z)p(z)}{q(z|x,a)} + \log\frac{p(c|z)}{q(c|x,a)}\right]dz \\
&= \textstyle\int_z q(z|x, a)\log\frac{p(a|z)p(z)}{q(z|x,a)}dz \\
&- \textstyle\int_z q(z|x, a)D_{KL}(q(c|x, a)\|p(c|z))dz.
\end{aligned} \tag{18}$$

The $D_{KL}(q(\cdot)\|p(\cdot))$ defines the Kullback-Leibler (KL) (Kullback 1987) divergence between two distribution $q(\cdot)$ and $p(\cdot)$. Since the definition of KL divergence is non-negative and the first term has no dependence on c, when $D_{KL}[q(c|x, a)\|p(c|z)] \equiv 0$, the loss $\mathcal{L}_{elbo}$ is maximized. At last, we could compute $q(c|x, a)$ by:

$$q(c|x, a) = p(c|z) \equiv \frac{p(c)p(z|c)}{\sum_{c'=1}^{K} p\left(c'\right)p\left(z|c'\right)}. \tag{19}$$

4.2 Collaborative Graph Convolutional Networks

In this subsection, we will introduce how the attributed graph clustering network helps to boost the performance of semi-supervised learning in a collaborative learning manner. CGCN firstly pre-train two modules, *i.e.* , GMM-VGAE module and semi-supervised module.

Algorithm 1: CGCN

Input: Graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{X})$; The sample set $\mathcal{D}$ with all nodes. Unlabel set $\mathcal{D}_u$. Labeled set $\mathcal{D}_l$. Number of classes K. Number of query nodes q. Number of iteration T. Number of pretrain iteration T_p. Number of retrain iteration T_r.

Output: Final classification result.

Random initialize the GCN parameters $\boldsymbol{W}^{(0)}, \boldsymbol{W}_{\mu}^{(1)}, \boldsymbol{W}_{\sigma}^{(1)}, \boldsymbol{W}_{semi}^{(1)}$ in CGCN;
for $t = 1, 2, \cdots, T_p$ **do**
  Update CGCN by Eq. 18, 22; ▷ *Pretrain Networks*
end
for $t = 1, 2, \cdots, T$ **do**
  for $k = 1, 2, \cdots, K$ **do**
    Initialize pseudo set $\tilde{U}_k = \emptyset$;
    ▷ *For each class select the highest confidence* ***labeled*** *node according to softmax prediction.*
    if $v_l = top(\boldsymbol{Z}_{semi}^{(2)}, 1, k)$ && $v_l \in \mathcal{D}_l$; **then**
      Get v_l belone to which cluster c_k in GMM-VAGE module;
      Set all unlabeled nodes in c_k cluster with pseudo label k, append them to $\tilde{U}_k$;
    end
    ▷ *For each class select the top q confidence* ***unlabeled*** *nodes according to softmax prediction.*
    if $v_u = top(\boldsymbol{Z}_{semi}^{(2)}, q, k)$ && $v_u \in \mathcal{D}_u$ **then**
      if $v_u \in \tilde{U}_k$ **then**
        $\mathcal{D}_l$.add(v_u); ▷ *Add to labeled set*
        $\mathcal{D}_u$.remove(v_u); ▷ *Remove from unlabeld set*
      end
    end
  end
  Retrain networks with T_r iteration on new $\mathcal{D}_l, \mathcal{D}_u$;
end

For the semi-supervised module which aims to predict the ground truth of the unlabeled nodes, the input consists of labeled and unlabeled attribute nodes. The vanilla GCN is applied for semi-supervised node classification via a two-layer graph convolution with a *softmax* operation on the output features:

$$\boldsymbol{Z}_{semi}^{(1)} = f_{Relu}\left(\boldsymbol{X}, \boldsymbol{A} | \boldsymbol{W}_{semi}^{(0)}\right); \tag{20}$$

$$\boldsymbol{Z}_{semi}^{(2)} = f_{Softmax}\left(\boldsymbol{Z}_{semi}^{(1)}, \boldsymbol{A} | \boldsymbol{W}_{semi}^{(1)}\right); \tag{21}$$

The loss function is defined as the cross-entropy over limited labeled samples:

$$\mathcal{L}_{semi} = -\sum_{i \in \mathcal{D}_l} \sum_{f=1}^{F} \boldsymbol{Y}_{if} \ln \boldsymbol{Z}_{semi}^{(2)} \tag{22}$$

where $\mathcal{D}_l$ is the set of labeled nodes and F is the dimension of the output features, which is equal to the number of classes. $\boldsymbol{Y} \in \mathbb{R}^{|\mathcal{D}_l| \times F}$ denotes the label indicator matrix. $\boldsymbol{W}_{semi}^{(0)}$ and $\boldsymbol{W}_{semi}^{(1)}$ are learnable parameters which can be trained via gradient descent method. It is worth mentioning that $\boldsymbol{Z}_{semi}^{(2)}$ from softmax operation can be regarded as the confidence by GCN prediction. To make the two modules collaborate more effectively, we enforce them to share the parameters of the first hidden layer, *i.e.* , :

$$\boldsymbol{W}^{(0)} = \boldsymbol{W}_{semi}^{(0)} \Rightarrow \boldsymbol{Z}^{(0)} = \boldsymbol{Z}_{semi}^{(0)}, \tag{23}$$

For unlabeled samples, there are two types of pseudo-labels, including clustering assignments from GMM-VGAE and predictions from semi-supervised GCN. As shown in Fig. 1, the top q nodes with high prediction confidence scores are selected to query in the clustering results. To label all clusters learned by GMM-VGAE, we select the highest confidence (softmax prediction score) labeled sample of each class and pass them to the clustering network. Thus, all unlabeled samples will be labeled if the labeled sample belong to certain cluster. For example, if a sample $\{\mathbf{x}_i, \mathbf{y}_i\}$ belongs to the k^{th} cluster, then the pseudo-labels of all samples in the k^{th} cluster is $\mathbf{y}_i$. In this way, clustering assignments and semi-supervised predictions are connected. If the clustering assignment is consistent with the semi-supervised prediction, then the node together with the pseudo-label is added to the label set. When the selected nodes are added to the labeled set $\mathcal{D}_l$, the network will be retrained.

Through repeated iterations, the number of reliable samples in the training set increases continuously, which will improve the performance of semi-supervised learning. The parameters of the first hidden layer are shared between the clustering module and semi-supervised node classification module. The feature representation ability of the first hidden layer is gradually improved by the multi-task learning strategy. Thus, CGCN can boost the performance of both semi-supervised node classification and clustering, which will be validated in Section 5. Overall, we describe the pipeline of CGCN in detail in Algorithm 1.

Table 1: Data statistics.

Datasets	# Nodes	# Edges	# Features	# Classes
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6
Pubmed	19717	44338	500	3

5 Experiment

In this section, experiments are conducted on three benchmark graph datasets and two tasks, including attributed graph clustering and semi-supervised node classification.

Table 2: Clustering performance on Cora, CiteSeer, and Pubmed.

Method	Input	Cora			Citeseer			Pubmed		
		Acc (↑)	NMI (↑)	F1 (↑)	Acc (↑)	NMI (↑)	F1 (↑)	Acc (↑)	NMI (↑)	F1 (↑)
K-means	Feature	34.65	16.73	25.42	38.49	17.02	30.47	57.32	29.12	57.35
Spectral	Graph	34.19	19.49	30.17	25.91	11.84	29.48	39.74	3.46	51.97
DeepWalk	Graph	46.74	31.75	38.06	36.15	9.66	26.70	61.86	16.71	47.06
DNGR	Graph	41.90	37.29	34.01	32.59	18.02	44.19	45.35	15.38	17.90
GAE	Both	53.25	40.96	41.97	41.26	18.34	29.13	64.08	22.97	49.26
VGAE	Both	55.95	38.45	41.50	44.38	22.71	31.88	65.48	25.09	50.95
MGAE	Both	63.43	45.57	38.01	63.56	39.75	39.49	43.88	8.16	41.98
ARGE	Both	64.00	44.90	61.90	57.30	35.00	54.60	59.12	23.17	58.41
ARVGE	Both	63.80	45.00	62.70	54.40	26.10	52.90	58.22	20.62	23.04
DAEGC	Both	70.40	52.80	**68.20**	67.20	39.70	63.60	67.10	26.60	65.90
AGC	Both	68.92	53.68	65.61	67.00	41.13	62.48	69.78	**31.59**	68.72
GMM-VGAE	Both	**71.50**	**54.43**	67.76	**67.44**	**42.30**	**63.22**	**71.03**	30.28	**69.74**

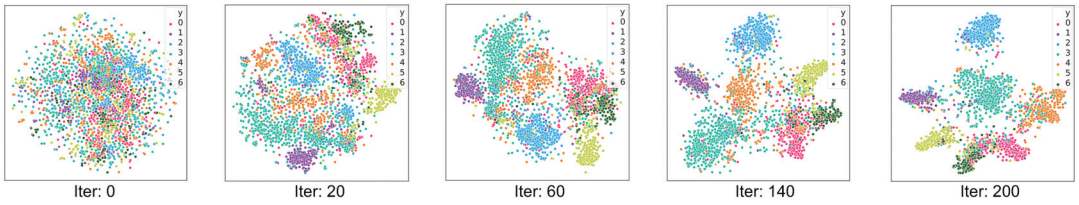

Figure 3: 2D visualization of the GMM-VGAE algorithm on Cora during training.

5.1 Benchmark Datasets

We evaluate the performance of the proposed model on three benchmark datasets. The detailed statistics of the three datasets are summarized in Table 1. *Cora*, *Citeseer* and *Pubmed* (Sen et al. 2008) are citation networks where the number of nodes varies from 2708 to 19717 and the number of feature varies from 500 to 3703.

5.2 Attributed Garph Clustering

In this section, we evaluate the performance of GMM-VGAE on attributed graph clustering independently.

Parameter Settings. We train GMM-VGAE module with Adam learning algorithm (the learning rate is set as 0.01) for all datasets. We construct encoder using a two-layer GCN with 32 and 16 filters respectively, and initialize encoder weights as described in (Glorot and Bengio 2010). During training, we firstly pretrain the graph autoencoder.

Baselines and Evaluation Metrics. Following (Xia et al. 2014), we use three widely used metrics to evaluate the clustering results, *i.e.* , accuracy (ACC), normalized mutual information (NMI) and macro F1 score (F1). According to the type of inputs, there are generally three kinds of attributed graph clustering methods: a) methods only using node features, including *k-means* clustering with node features. b) methods only using graph structures, including spectral clustering that uses the eigenvalues to perform dimensionality reduction before clustering, DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) which is a structure-only representation learning method, and DNGR (Cao, Lu, and Xu 2016) which uses stacked denoising auto-encoders and encodes each vertex into a low-dimensional vector representation. c) methods using both node features and graph structures, including graph autoencoder (GAE), variational graph autoencoder (VGAE) (Kipf and Welling 2016), marginalized graph autoencoder (MGAE) (Wang et al. 2017), adversarially regularized graph autoencoder (ARGE), variational graph autoencoder (ARVGE) (Pan et al. 2018), unsupervised deep attentional embedded graph clustering (DAEGC) (Wang et al. 2019), and adaptive graph convolution (AGC) (Zhang et al. 2019).

Table 3: Semi-supervised classification accuracy on Cora

Rate	0.5%	1%	2%	3%	4%	5%
LP	56.4	62.3	65.4	67.5	69.0	70.2
GCN-V	38.0	52.0	62.4	70.8	74.1	77.6
GCN+V	50.9	62.3	72.2	76.5	78.4	79.7
Co-training	56.6	66.4	73.5	75.9	78.9	80.8
Self-training	53.7	66.1	73.8	77.2	79.4	80.0
Union	58.5	69.9	75.9	78.5	80.4	81.7
Intersection	49.7	65.0	72.9	77.1	79.4	80.2
Two-stage	57.9	67.0	74.8	79.0	81.5	83.3
CGCN	**64.3**	**72.4**	**76.8**	**80.1**	**82.7**	**84.2**
*GMM-VGAE**	*71.5*	*71.9*	*72.5*	*73.3*	*73.6*	*73.8*

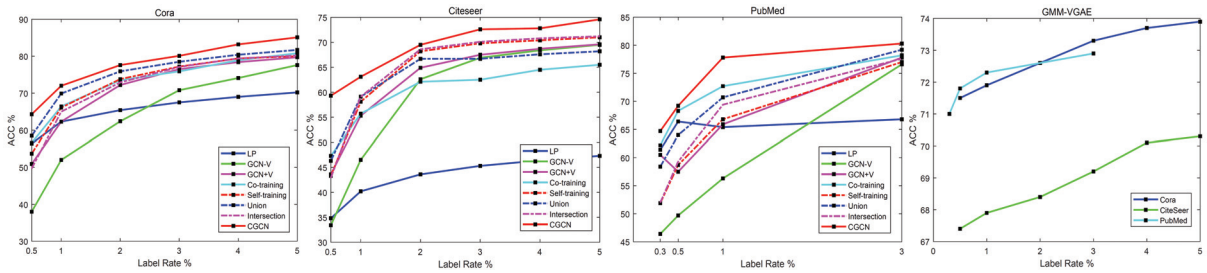

Figure 4: The first three columns represent the performance of CGCN semi-supervised module in different label rates, and the last column represents the performance of clustering module (GMM-VGAE) in different label rates.

Table 4: Semi-supervised accuracy on CiteSeer

Rate	0.5%	1%	2%	3%	4%	5%
LP	34.8	40.2	43.6	45.3	46.4	47.3
GCN-V	33.4	46.5	62.6	66.9	68.4	69.5
GCN+V	43.6	55.3	64.9	67.5	68.7	69.6
Co-training	47.3	55.7	62.1	62.5	64.5	65.5
Self-training	43.3	58.1	68.2	69.8	70.4	71.0
Union	46.3	59.1	66.7	66.7	67.6	68.2
Intersection	42.9	59.1	68.6	70.1	70.8	71.2
Two-stage	51.3	60.6	68.9	71.4	71.9	72.3
CGCN	**59.3**	**63.1**	**69.5**	**72.6**	**72.8**	**74.6**
*GMM-VGAE**	*67.4*	*67.9*	*68.5*	*69.3*	*69.6*	*70.2*

Table 5: Semi-supervised classification accuracy on PubMed

Rate	0.03%	0.05%	0.1%	0.3%
LP	61.4	66.4	65.4	66.8
GCN-V	46.4	49.7	56.3	76.6
GCN+V	60.5	57.5	65.9	77.8
Co-training	62.2	68.3	72.7	78.2
Self-training	51.9	58.7	66.8	77.0
Union	58.4	64.0	70.7	79.2
Intersection	52.0	59.3	69.4	77.6
Two-stage	60.7	64.1	72.2	78.2
CGCN	**64.7**	**69.2**	**77.8**	**80.3**
*GMM-VGAE**	*71.0*	*71.8*	*72.3*	*72.7*

Result Analysis We compare the proposed GMM-VGAE with the state-of-the-art attributed graph clustering algorithms and the clustering results are shown in Table 2. The results of the comparison methods are directly duplicated from the original paper. We can observe that our method outperforms all the baselines in terms of different evaluation metrics. It is worth mentioning that GMM-VGAE significantly outperforms GAE and VGAE, which verifies the effectiveness of GMM in discovering complex data distributions. In addition, we use the t-SNE algorithm (Maaten and Hinton 2008) to visualize the Cora dataset on the learned embedding during training. As shown in Figure 3, we can obtain a more meaningful distribution of the graph data.

5.3 Semi-Supervised Node Classification

In this section, we will evaluate the performance of proposed collaborative graph convolution networks (CGCN) on semi-supervised learning task. At the same time, we will observe the change of the clustering accuracy of GMM-VGAE in CGCN. GMM-VGAE* means the clustering performance under collaborative learning (in CGCN).

Parameter Settings We set the number of pretrain iterations T_p as 200, the number of retrain iterations T_r as 20 and the number of query high confidence nodes q as 20 for each pseudo-label assignment with $T = 5$ times. Following (Kipf and Welling 2017), we set the learning rate, dropout rate, regularization weight, and the size of second hidden layer as 0.01, 0.2, 0.5×10^{-4} and 16, respectively. For each run, we split the data into one small sample subset for training, and the test sample subset with 1000 samples. Similar to (Li, Han, and Wu 2018), in order to explore the effectiveness of collaborative learning, we conducted experiments with different label rates, *i.e.* , 0.1%, 1%, 2%, 3%, 4%, 5% on Cora and CiteSeer, and 0.03%, 0.05%, 0.1%, 0.3% on PubMed.

Baselines and Evaluation Metrics We compare our methods with GCN and it variants (Li, Han, and Wu 2018). 1) LP: Label propagation using ParWalks (Wu et al. 2012). 2) GCN+V: GCN with validation. 3) GCN-V: GCN without validation. 4) Co-Training: Method that trains GCN with a random walk model. 5) Self-Training: Simple self-training based on softmax prediction. 6) Union: Method that expands the label set with the most confident predictions found by the random walk and those found by the GCN itself, and then uses the expanded label set to continue to train the GCN. 7) Intersection: Method that adds the most confident predictions found by both the random walk and the GCN. 8) Two-stage: First use clustering network to pre-train and then train the semi-supervised network using the labelled data.

Result Analysis As shown in Tables 3, 4, 5, our methods outperform the comparison methods using much fewer labels. CGCN has strong competitiveness in the case of very low label rates. For example, CGCN only needs 3% of the labeled data to get better results than GCN using 5% of the data on Cora and CiteSeer. Additionally, we report the accuracy of GMM-VGAE under different label rates in collaborative learning. The clustering performance is improved

compared with the completely unsupervised case. That is because the clustering module and semi-supervised module share the first hidden layer. With the improvement of semi-supervised module, the feature representation ability of the first hidden layer will also be enhanced, which can improve the clustering performance.

6 Conclusion

In this paper, we proposed a multi-task graph learning model, *i.e.* , collaborative graph convolutional networks (CGCN). CGCN effectively exploits attributed graph clustering, to collaboratively boost the performance of semi-supervised learning. To model complex data distributions in the graph embedding space, we presented an end to end attributed graph clustering network by combining variational graph auto-encoder with Gaussian mixture model. Experiments on both attributed graph clustering and semi-supervised node classification tasks validate the superiority of the proposed models.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 61876127, Natural Science Foundation of Tianjin Under Grants 17JCZDJC30800, 18YFZCGX00390, 18YFZCGX00680, and Young Elite Scientists Sponsorship Program by Tianjin.

References

Atwood, J., and Towsley, D. 2016. Diffusion-convolutional neural networks. In *NeurIPS*, 1993–2001.

Bojchevski, A., and Günnemann, S. 2018. Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure. In *AAAI*, 2738–2745.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2014. Spectral networks and locally connected networks on graphs. In *ICLR*.

Cao, S.; Lu, W.; and Xu, Q. 2016. Deep neural networks for learning graph representations. In *AAAI*, 1145–1152.

Caron, M.; Bojanowski, P.; Joulin, A.; and Douze, M. 2018. Deep clustering for unsupervised learning of visual features. In *ECCV*, 132–149.

Chang, J., and Blei, D. M. 2009. Relational topic models for document networks. In *AISTATS*, 81–88.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 3837–3845.

Glorot, X., and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 249–256.

Hamaguchi, T.; Oiwa, H.; Shimbo, M.; and Matsumoto, Y. 2017. Knowledge transfer for out-of-knowledge-base entities : A graph neural network approach. In *IJCAI*, 1802–1808.

He, D.; Feng, Z.; Jin, D.; Wang, X.; and Zhang, W. 2017. Joint identification of network communities and semantics via integrative modeling of network topologies and node contents. In *AAAI*, 116–124.

Henaff, M.; Bruna, J.; and LeCun, Y. 2015. Deep convolutional networks on graph-structured data. *CoRR* abs/1506.05163.

Jiang, Z.; Zheng, Y.; Tan, H.; Tang, B.; and Zhou, H. 2017. Variational deep embedding: An unsupervised and generative approach to clustering. In *IJCAI*, 1965–1972.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. In *ICLR*.

Kipf, T. N., and Welling, M. 2016. Variational graph auto-encoders. In *NeurIPS Workshop on Bayesian Deep Learning*.

Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Kullback, S. 1987. Letter to the editor: The kullback-leibler distance. *AMERICAN STATISTICIAN*.

Li, Q.; Han, Z.; and Wu, X. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI*, 3538–3545.

Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *Journal of machine learning research* 9(Nov):2579–2605.

Monti, F.; Boscaini, D.; Masci, J.; Rodolà, E.; Svoboda, J.; and Bronstein, M. M. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In *CVPR*, 5425–5434.

Nalisnick, E.; Hertel, L.; and Smyth, P. 2016. Approximate inference for deep latent gaussian mixtures. In *NeurIPS Workshop on Bayesian Deep Learning*, volume 2.

Pan, S.; Hu, R.; Long, G.; Jiang, J.; Yao, L.; and Zhang, C. 2018. Adversarially regularized graph autoencoder for graph embedding. In *IJCAI*, 2609–2615.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: online learning of social representations. In *SIGKDD*, 701–710.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine* 29(3):93–93.

Shu, R. 2016. Stochastic video prediction with conditional density estimation. In *ECCV Workshop on Action and Anticipation for Visual Learning*.

Sun, K.; Zhu, Z.; and Lin, Z. 2019. Multi-stage self-supervised learning for graph convolutional networks. *CoRR* abs/1902.11038.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. Arnetminer: extraction and mining of academic social networks. In *SIGKDD*, 990–998.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2017. Graph attention networks. In *ICLR*.

Wang, C.; Pan, S.; Long, G.; Zhu, X.; and Jiang, J. 2017. MGAE: marginalized graph autoencoder for graph clustering. In *CIKM*, 889–898.

Wang, C.; Pan, S.; Hu, R.; Long, G.; Jiang, J.; and Zhang, C. 2019. Attributed graph clustering: A deep attentional embedding approach. In *IJCAI*, 3670–3676.

Wang, X.; Ye, Y.; and Gupta, A. 2018. Zero-shot recognition via semantic embeddings and knowledge graphs. In *CVPR*, 6857–6866.

Wu, X.; Li, Z.; So, A. M.; Wright, J.; and Chang, S. 2012. Learning with partially absorbing random walks. In *NeurIPS*, 3086–3094.

Xia, R.; Pan, Y.; Du, L.; and Yin, J. 2014. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*, 2149–2155.

Xie, Q.; Dai, Z.; Hovy, E. H.; Luong, M.; and Le, Q. V. 2019. Unsupervised data augmentation. *CoRR* abs/1904.12848.

Yang, T.; Jin, R.; Chi, Y.; and Zhu, S. 2009. Combining link and content for community detection: a discriminative approach. In *SIGKDD*, 927–936.

Zhang, X.; Liu, H.; Li, Q.; and Wu, X. 2019. Attributed graph clustering via adaptive graph convolution. In *IJCAI*, 4327–4333.