AUT Journal of Modeling and Simulation

AUT J. Model. Simul., 56(1) (2024) 103-116
DOI: 10.22060/miscj.2024.23107.5358

# A Guassian Mixture Variational Graph Autoencoder for Node Classification

Mohadeseh Ghayekhlou, Ahmad Nickabadi *

Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran.

**ABSTRACT:** Graph embedding is the procedure of transforming a graph into a low-dimensional, informative representation. The majority of existing graph embedding techniques have given less consideration to the embedding distribution of the latent codes and more attention to the graph's structure. Recently, Variational Graph AutoEncoders (VGAEs) have demonstrated good performance by learning smooth representations from unlabeled training samples. On the other hand, in regular VGAEs, the prior distribution over latent variables is generally a single Gaussian distribution. However, complex data distributions cannot be well-modelled under the assumption of a single Gaussian distribution. This choice of prior distribution is important because each dimension of a multivariate Gaussian can learn a separate continuous latent feature, which can result more structured and disentangled representation. In this paper, we employ the Gaussian Mixture Model (GMM) as the prior distribution in a Variational Graph Autoencoder (GMM-VGAE) framework for node classification in graphs. In this framework, GMM effectively discovers the inherent complex data distribution, and graph convolutional networks (GCNs) exploit the structure of the nodes of a graph to learn more informative representations. The proposed model incorporates several Graph Convolutional Networks (GCNs): one to map the input feature vector to the latent representation utilized for classification, another to generate the parameters of the latent distribution for learning from unlabeled data, and finally, an additional GCN is employed for reconstructing the input and delivering the reconstruction loss. Through extensive experiments on well-known Citations, Co-authorship, and Social network graphs, GMM-VGAE's superiority over state-of-the-art methods is demonstrated.

**Review History:**

Received: Apr. 13, 2024
Revised: Jul. 17, 2024
Accepted: Jul. 30, 2024
Available Online: Sep. 05, 2024

**Keywords:**

Graph Convolutional Networks (GCNs)
Variational Graph Autoencoders (VGAEs)
Gaussian Mixture Model (GMM)

## 1- Introduction

A graph is a type of data structure made up of edges and nodes. Graph data is huge dimensional and has a complicated structure, making it difficult to process with most typical machine learning algorithms. As a general solution to these issues, Graph Representation Learning (GRL), is introduced to embed nodes or graphs into smaller vector spaces while preserving as many structural and attribute properties as possible. Consequently, the main focus of GRL techniques has shifted to deep graph embedding systems based on the Graph Neural Networks (GNN) paradigm. GNNs can create node representations based on the structure and feature information of the graph by using parameter sharing. The neglect of data distribution by (GNNs) can result in inferior representations and susceptibility to overfitting[1]. The integration of GNNs with deep generative models, however, has demonstrated a substantial enhancement in the learned data distribution, leading to significantly improved generated representations.

*Corresponding author's email: nickabadi@aut.ac.ir

The framework of the Variational Graph Auto-Encoder (VGAEs) [2] is extensively explored in this context. In the domain of graph representation learning, VGAEs aggregate neighbourhood messages through Graph Convolutional Network (GCN) encoders. This incorporation aims to generate more informative node embeddings within VGAE. However, current VGAEs encounter architectural issues. They employ an unimodal Gaussian distribution as the prior distribution for their latent space, a choice that poses challenges, especially in handling complex data where each sample may come from various distributions, whether known or unknown, collectively known as multimodal distributions. In essence, multimodal data encompass multiple regions with elevated probability levels.

In order to address this issue, a novel Variational Graph Autoencoder based on GMMs (GMM-VGAE) is proposed. We employ the Gaussian Mixture Models (GMMs) as the prior distribution of VGAE models. The fusion of these elements enhances the interpretability of the proposed model. Our contributions can be summarized as follows: We utilize the VGAE framework, incorporating the Gaussian Mixture

Model (GMM) to represent the prior distribution. We show the superiority of GMM-AVGAE over state-of-the-art techniques on Citation, Co-authorship, and Social network benchmarks through node classification tasks. It should be noted that it is the first time VGAE is employed in the node classification task.

In the remainder of the paper, Section 2 delves into related works, while Section 3 outlines the problem statement. The proposed GMM-VGAE model is detailed in Section 4. Experimental results and evaluations are presented in Section 5. The paper concludes with Section 6.

## 2- Related Works

Lately, scientists have employed encoders that are specifically made to combine the local neighbourhood data of nodes and produce low-dimensional embeddings. For example, Autoencoders prove to be a valuable approach for acquiring low-dimensional representations from unlabeled data. The utilization of graph autoencoders (GAEs) [3] to construct deep latent representations of network topologies has been increasingly favoured in recent years. The GAE models integrate graph convolutional networks (GCNs) [3] into the autoencoder architectures to learn a latent representation for nodes. This is achieved by reconstructing the adjacency matrix or initial feature vectors from these latent representations. Additionally, GAE-based approaches such as MGAE [4], GDN [5], and GALA [6] strive to preserve informative node features in the latent representation through the use of learnable encoders and decoders. While GAE-based methods demonstrate effectiveness, they tend to neglect data distribution, resulting in suboptimal representations and susceptibility to overfitting [1]. The integration of the GAE framework with deep generative models serves to address these issues. Deep generative models can depict complicated relationships and interactions between input and output data by taking data distribution into consideration [7]. They have proven effective in applications such as unsupervised learning and the regularization of latent representations.

The primary categories of generative models encompass Variational Graph AutoEncoders (VGAEs) [2]. VGAE employs the Gaussian distribution as a prior and encourages the learned representations to closely align with this prior through the incorporation of a KL divergence penalty. Xie et al. [8] presented a representation learning technique for networked documents to model document contents and relations using a VGAE framework. The Higher-order Graph Attention Network (HGAT) [9] which analyses and shuffles neighbourhood information of each document in each order, is the foundation of their suggested approach. The network embedding method known as NetVAE, created by Jin et al. [10], addresses the orthogonality between network topology data and node attributes. Here, node characteristics and network structure compression use the same encoder. However, a dual decoder supported by GMMs is introduced to allow independent reconstruction of network topologies and node properties. In addition, the paper [11] enhances Graph Neural Networks (GNNs) by integrating deep generative models with the Variational Graph Auto-Encoder (VGAE) framework to address uncertainty in hidden variables. To improve upon traditional VGAE methods that struggle with data multimodality, the authors propose using a Gaussian Mixture Model (GMM) for the prior distribution. Additionally, adversarial regularization is incorporated to enhance the effectiveness of latent representations. The proposed approach is specifically applied to the link prediction task, demonstrating its practical utility

The majority of mentioned publications are based on the assumption that the prior uncertainty of the observed data would follow a Gaussian distribution which is not correct for complex multimodal data. In some publications, other distributions are used in place of the traditional Gaussian distribution. The von Mises-Fisher (vMF) unimodal density is taken into account by Davidson et. al. in their S-VGAE model [12] to model the prior distribution and produce a better representation. Zheng et al.[13] suggest a method, named DBGAN, for estimating the prior distribution of latent representation in a structure-aware way. Through prototype learning, the graph and feature space are implicitly connected rather than using the common Gaussian distribution assumption.

## 3- Problem Statement

The problem of node classification on graphs can be formally defined as follows. Given a graph $G=(V,A,X,Y)$, where the set of nodes is $V=\{v_1,...,v_N\}$; the adjacency matrix is $A\in R^{N\times N}$, and $a_{ij}$ denotes the presence or absence of an edge between $v_i$ and $v_j$. The feature matrix of $N$ nodes with $F$ features is $X\in R^{N\times F}$, where $X_{ij}$ Denotes the $j$-th characteristic of the $i$-th node. Also, in the Label set $Y$ where labels are partially observed, the goal is to predict the labels of the unlabeled nodes. Specifically, we aim to learn a function $f$ that maps each node $v_i$ to a label $y_i$ based on its features and the graph structure.

Formally, the node classification task involves learning node embeddings $Z$ where each $z_i$ is in $\mathbb{R}^d$ for all $v_i\in V$. Using these embeddings, each node $v_i$ is classified as one of the predefined classes $C=\{c_1,c_2,...,c_K\}$. The classification function $f$ can be trained using supervised learning on the labelled nodes and semi-supervised or unsupervised methods to exploit the structure and feature information of the unlabeled nodes.

The node classification problem can be represented as an optimization problem. Given the graph $G=(V,A,X,Y)$, the objective is to minimize the classification loss function $L$, which can be defined as:

$$L=\frac{1}{|V_L|}\sum_{v_i\in V_L} l\left(y_i,f(z_i)\right) \tag{1}$$

where $V_L\subset V$ is the set of labelled nodes, $l\left(y_i,f\left(z_i\right)\right)$ is the loss function measuring the discrepancy between the true label $y_i$ and the predicted label $f\left(z_i\right)$. By solving this optimization problem, we aim to accurately classify the

**Table 1. The concept of the notations used in the paper.**

| Symbol | Meaning |
|---|---|
| $A$ | The adjacency matrix |
| $N$ | The number of nodes in the graph |
| $K'$ | The number of classes in the graph |
| $X$ | The feature matrix |
| $F$ | The number of features in X |
| $X,W,C$ | The latent variables of the proposed method |
| $z_g$ | The latent variable (representation) used for reconstruction of X |
| $z_f$ | The latent variable (representation) used for classification |
| $\phi_f$ | The parameters of the $q_{\phi_f}(z_f \mid X, A)$ module |
| $\phi_g$ | The parameters of the $q_{\phi_g}(G \mid z_f)$ module |
| $\phi_w$ | The parameters of the $q_{\phi_w}(W \mid z_f)$ module |
| $\theta$ | The parameters of the $p_\theta(X \mid G)$ module |
| $\beta$ | The parameters of $p_\beta(G \mid W)$ module |
| $K$ | The number of components in GMM |
| $\pi$ | Mixing probability of GMM |
| $D_{KL}(q\ (\|)p)$ | KL-divergence between q and p |

nodes in the graph, leveraging both the labelled data and the inherent structure of the graph.

In addition, we use Gaussian Mixture Models (GMM) where GMM belongs to the parametric probability density function family. It is a weighted sum of $K$ components. Each of these components follows a multivariate Gaussian distribution. The mixing weight, also known as the mixture probability or coefficient, is denoted by $\pi_i$ for the $i$-th component, ensuring that $\sum_{i=1}^{K}\pi_i = 1$.

The notations used in this paper are listed in Table 1.

## 4- The Proposed Method

In this section, we introduce the Variational Graph Autoencoder based on the Gaussian Mixture Model (GMM-VGAE) designed for node classification in graphs. Initially, we outline the overarching structure of the proposed model, followed by a comprehensive examination of its key components. Lastly, we delve into the description of the loss functions and expound on the training process of the model.

### 4- 1- Overview

Following the idea of Guasian Mixture Models for Variational Auto Encoder (GMM-VAEs) )[14], instead of mapping the input vector $x$ to a fixed latent vector $z$, a multimodal distribution over the latent space for each input data is learned. Due to the sampling step of GMM-VAEs, these models produce different representations for fixed input data and hence different labels may be assigned to the same input [15].

The GMM-VGAE framework is composed of two primary components within the general architecture of a Variational Autoencoder (VAE): the inference module and the generative module. The overall structure of the proposed model is depicted in 1. The inference module is responsible for encoding the input data into latent representations and consists of a series of Graph Convolutional Networks (GCNs) that process the input feature matrix and adjacency matrix to generate the latent variables. The generative module reconstructs the input data from the latent representations, using the latent variables generated by the inference module to reconstruct the feature matrix and generate the Gaussian Mixture Model (GMM) parameters.

The model takes two inputs: the adjacency matrix $A$, capturing the structural information of the graph $G$ with $N$ nodes, and the feature matrix $X$, which corresponds to the feature matrix of $N$ nodes with $F$ features. Both the feature matrix $X$ and the adjacency matrix $A$ serve as inputs to the initial Graph Convolutional Network (GCN), denoted as (E1), which generates the initial latent representation $z_f$ for each node's feature vector. Subsequently, this representation undergoes further processing in a second GCN, labelled as (E2), resulting in the derivation of $z_g$. Moreover, $z_f$ is

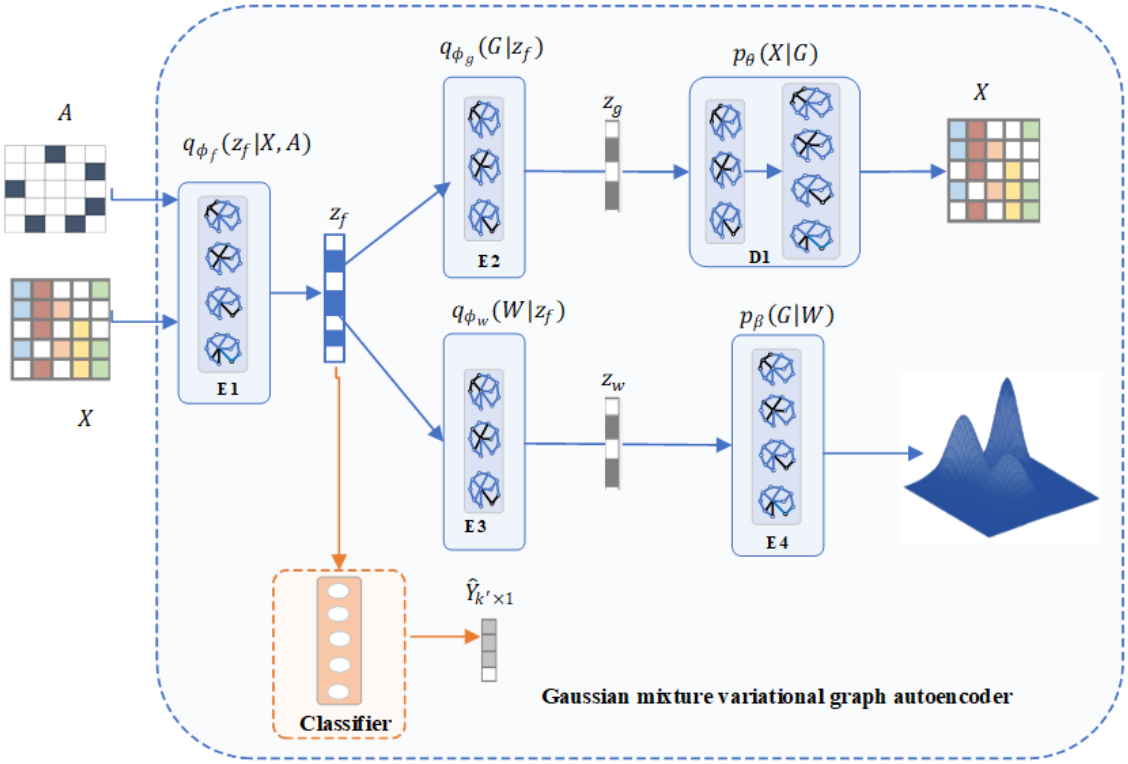


**Fig. 1. The outline of the proposed Variational Graph Autoencoder based on the Guassian Mixture Model (GMM-VGAE).**

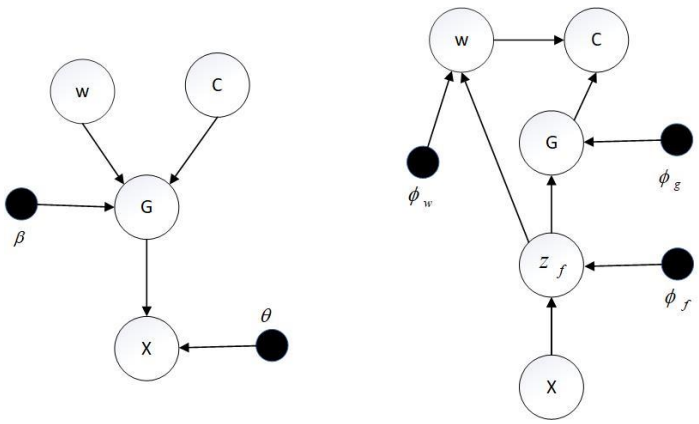


**Fig. 2. The graphical models for GMM-VGAE show the generative model (left) and the inference model (right).**

subjected to additional transformation within a third GCN, denoted as (E3), ultimately producing $z_w$. In parallel, $z_w$ serves as input for a fourth GCN, labelled as (E4), which generates $K$ latent representations in the form of $K$ mean vectors ($\mu_k$) and variance vectors ($\sigma_k$). For the purpose of reconstructing the input feature vectors, a two-layer GCN network (D1) is employed, utilizing the latent representation $z_g$ as input.

The inference module includes the following steps: E1 encodes the input feature matrix $X$ and adjacency matrix $A$ to produce $z_f$; E2 further encodes $z_f$ to derive $z_g$; E3 transforms $z_f$ to generate $z_w$; and E4 encodes $z_w$ to produce $K$ mean vectors ($\mu_k$) and variance vectors ($\sigma_k^2$). The generative module includes the decoder (D1), which uses $z_g$ to reconstruct the input feature matrix $X$.

Detailed explanations of each of these two components and their corresponding training procedures are presented in the subsequent subsections.

### 4- 2- Generative, Inference

GMM-VGAE consists of two main modules: a GCN-based generative module, GCN-based inference module. In general, the generative and inference models of our proposed method are depicted in Figure 2. In the following, each of these modules is explained in detail.

**GCN-based generative module:** As depicted in Figure 2, the generative model $p_{\beta,\theta}(X,G,W,C)=p(C)p(W)p_\beta(G|W,C)p_\theta(X|G)$ generates an observed sample $X$ from a set of latent variables $G$, $W$, and $C$, in contrast to VGAE, which employs a singular latent variable. This process can be described as:

$$\begin{aligned} W &\sim \mathcal{N}(0,I) \\ C &\sim Mult(\pi) \\ G|W,C &\sim \Pi_{k=1}^{K} N\left(\mu_{c_k}(W;\beta),\sigma_{c_k}^2(W;\beta)\right)^{c_k} \\ X \mid G &\sim N\left(\mu(G;\theta),\sigma^2(G;\theta)\right) \end{aligned} \tag{2}$$

Where $K$ is the number of components in the mixture model and this parameter is identified as a model hyperparameter. $W$ follows a Gaussian distribution with mean zero and covariance matrix $I$. Also, a one-hot vector $C$ shows the mixing coefficients of the Gaussian mixture components which is sampled from mixing probability $\pi$. Here, $\pi=1/k$ to cause $C$ to be uniformly distributed.

As shown in Figure 1, there is one GNN with the parameter of $\beta$ and one GNN with parameters of $\theta$. According to the definition provided, the input of the GNN $p_\beta(G|W)$ parameterized by $\beta$ is $W$ and its output is a Gaussian mixture of $K$ numbers of $\mu_{c_k}$ and $\Sigma_{c_k}$. In the proposed model, $p_\theta(X|G)\sim N\left(\mu(G;\theta),\sigma^2(G;\theta)\right)$ which is parameterized by $\theta$ is a two-layer GCN network that uses the structural information of the graph and the embedding of the neighbouring nodes to reconstruct the feature vector of a node. Training the model with the corresponding reconstruction loss forces the model to produce more representative embeddings for the input data.

**The GCN-based inference module:** This module encodes the input to a distribution over the latent space. However, it is challenging to directly solve the generative model, for example, determining the maximum likelihood estimation (MLE) of the parameters and the maximum a posteriori (MAP) of the latent variables. So, graph convolutional networks GCNs [3] are used to learn the representation and to produce the nodes' representations based on the graph structure $A$ and the input feature matrix $X\in R^{N\times F}$. As shown in Figure 1, the first GNN $(E1)$ as $q_{\phi_f}(Z_f|X,A)$ has a one-layer GCN architecture and maps the input feature matrix $X$ and the graph structure $A$ to the latent representation $z_f$. It should be noted that $z_f$ is the final representation of GMM-VGAE for the classification of the nodes.

Based on the mean-field variational family, we approximate the posterior distribution $p(G,W,C|z_f)$ using a new distribution $q(G,W,C|z_f)$ which is parameterized by trainable parameters $\phi$, and $\beta$. Referring to Figure 2, in the inference model, we assume $q(G,W,C|z_f)$ as

$$q(G,W,C\mid z_f)= \prod_{i=1}^{\mathcal{N}} q_{\phi_g}(g_i\mid z_{f_i})q_{\phi_w}(w_i\mid z_{f_i})p_\beta(c_i\mid g_i,w_i) \tag{3}$$

In this equation, $\beta$ denotes the collection of GNN parameters associated with each GMM component. The set of GNN parameters related to $G$ and $W$ in variational factors are shown by $\phi_g$ and $\phi_w$, respectively. In addition, i indexes over data points and in order to make the notations simpler and assume one node at a time, the i indices are dropped in the following. We take the z-posterior, $p_\beta(C|G,W)$, as:

$$\begin{aligned} p_\beta(c_j=1\mid G,W) &= \frac{p(c_j=1)p(G|c_j=1,W)}{\sum_{k=1}^{K}(p(c_k=1)p(G|c_k=1,W))} \\ &= \frac{\pi_j\mathcal{N}\left(G|\mu_j(W;\beta),\sigma_j^2(W;\beta)\right)}{\sum_{k=1}^{K}\pi_k\mathcal{N}\left(G|\mu_k(W;\beta),\sigma_k^2(W;\beta)\right)} \end{aligned} \tag{4}$$

The Evidence lower bound (ELBO), as the variational inference objective, is used to train the generative model, which can be determined as:

$$L_{ELBO} = E_{q(G,W,C|z_f)}\left[\log\left(\frac{p(X,G,W,C)}{q(G,W,C|z_f)}\right)\right] \tag{5}$$

in which, $p(X,G,W,C)=p(C)p(W)p(G|W,C)p(X|G)$,

so the ELBO loss is defined as:

$$\begin{aligned} L_{ELBO} &= E_{q(G,W,C|z_f)}[logp(X,G,W,C) - q(G,W,C \mid z_f)] \\ &= E_{q(G,W,C|z_f)}[logp(C) + logp(W) + logp(G \mid W,C) \\ &\quad + logp(X \mid G) - q(G \mid z_f) - q(W \mid z_f) - p(C \mid G,W)] \end{aligned} \quad (6)$$

The ELBO can be expressed as follows:

$$\begin{aligned} L_{ELBO} &= E_{q(G|z_f)}[\log p(X \mid G)] \\ &\quad \underbrace{\text{Reconstruction term}} \\ &\quad -E_{q(W|z_f)p(C|G,W)}\left[KL\left(q_{\phi_g}(G \mid z_f) \parallel p_\beta(G \mid W,C)\right)\right] \\ &\quad \underbrace{\text{G-conditional prior term}} \\ &\quad -KL\left(q_{\phi_w}(W \mid z_f) \parallel p(W)\right) \\ &\quad \underbrace{\text{W-prior term}} \\ &\quad -E_{q(G|z_f)q(W|z_f)}\left[KL\left(p_\beta(C \mid G,W) \parallel p(C)\right)\right] \\ &\quad \underbrace{\text{C-prior term}} \end{aligned} \quad (7)$$

The first term in the lower bound is the reconstruction term and the following are G-conditional prior term, W-prior term, and C-prior term [14], respectively. In the ELBO equation, the **reconstruction term** can be estimated by taking Monte Carlo samples from $q_{\phi_g}(G|z_f)$ as $z_g$, and improves the input reconstruction qualities. The reconstruction loss is defined as:

$$E_{q(G|z_f)}[\log p(X \mid G)] \approx E_{z_g \sim q_{\phi_g}(G|z_f)}[\log p_\theta(X \mid z_g)] \quad (8)$$

Where $\phi$ and $\theta$ are parameters of the GNNs. So, the model tries to increase the probability of generating the input $X$ as the output. The reconstruction loss is propagated back through this network. To approximate the **conditional prior term**, Monte Carlo can be utilized without the need to sample from the discrete distribution $p(C|G,W)$ as demonstrated in Equation(5). Therefore, the conditional prior term can be expressed as follows:

$$\begin{aligned} &E_{q(W|z_f)p(C|G,W)}\left[KL\left(q_{\phi_g}(G \mid z_f) \parallel p_\beta(G \mid W,C)\right)\right] \approx \\ &\frac{1}{M}\sum_{j=1}^{M}\sum_{k=1}^{K} p_\beta\left(c_k = 1 \mid g^j, w^j\right) \\ &\left(q_{\phi_g}(G \mid z_f) \parallel p_\beta(G \mid w^j, c_k = 1)\right) \end{aligned} \quad (9)$$

Where $M$ is the number of Monte Carlo samples used to approximate the expectation. The **W-prior term** can be calculated analytically. The **C-prior term** in ELBO aims to reduce the KL divergence between the uniform prior and the C-posterior. In our model, C is a discrete latent variable, and C-posterior measures how far G is from each cluster position by W creates to calculate the likelihood of a cluster being assigned. By maximizing overlap and bringing the means closer together, it would try to bring the clusters together.

In addition, to improve the training process for the node classification task, we introduce a supervised loss term into the objective function. The cross-entropy loss function is defined as:

$$L_{cls} = -\sum_{i=1}^{L} Y_i \log\left(\hat{Y}_i\right) \quad (10)$$

where $\hat{Y}$ Represents the class probabilities estimated by the classifier for the input data and $L$ is labeled data points. Each element $Y_i$ in $Y$ is a one-hot vector representing the true class of the $i$ -th labelled node.

Finally, we combine the ELBO and classification losses to train our GMM-VGAE as follows:

$$L_{GMM-VGAE} = \lambda_1 L_{ELBO} + L_{cls} \quad (11)$$

where $\lambda_1$ is a positive scalar weights that balance the terms in the loss. The pseudo-code of the proposed model is given in Algorithm 1.

### 4- 3- Training Procedure

The training procedure includes the following steps: Initialize the parameters $\phi$, $\beta$, and $\theta$ randomly. For each iteration, encode the input data using the sequence of GCNs to generate $z_f$, $z_g$, and $z_w$. Sample from the latent representations and decode them to reconstruct the input feature matrix and generate GMM parameters. Calculate the reconstruction loss and the GMM-based loss to optimize the model parameters. Finally, update the model parameters using the calculated losses.

## 5- Experimental Results and Evaluations

To assess and analyze the performance of GMM-VGAE in node classification on graphs with a limited number of labelled nodes, several experiments are conducted in this section. Five Citation graph datasets, namely Cora-full[16], PubMed [17], CiteSeer [18], Cora [18], UAI2010 [19], two Co-authorship graph data sets, namely Coauthor CS [20] and Coauthor Physics [20] and two Social networks, namely BlogCatalog [21] and Flickr [22] are selected for the experiments. Table 2 summarizes the statistics of these datasets.

Within the Citation datasets, each node represents an article, and the presence of an edge between two nodes indicates that one article cites the other. The topic of each article is denoted by a label. In this context, class labels serve as representations of the primary research area of the author, and node features encapsulate the keywords associated with the papers.

In the Co-authorship datasets, which derive from the

**Algorithm 1** The learning process of GMM-VGAE

**Input:** $A$ (Adjacency Matrix), $X$ (Feature Matrix), $T$ (number of iterations),$K'$ (The number of classes in the graph)
Initialize $\phi$,$\beta$, and $\theta$ randomly.
**for** iteration $= 1, 2, ..., T$ **do**
    $z_f \leftarrow$ Encode with $q_{\phi_f}(z_f \mid X, A)$ ▷ Encode input data into representation $z_f$
    $z_g \leftarrow$ Encode with $q_{\phi_g}(G \mid z_f)$ ▷ Encode $z_f$ to obtain latent representation $z_g$
    $z_w \leftarrow$ Encode with $q_{\phi_w}(W \mid z_f)$ ▷ Encode $z_f$ to obtain latent representation $z_w$
    Sample $z_w$ and Encode $z_w$ to $p_\beta(G \mid W)$
    Sample $z_g$ and Decode $z_g$ into $p_\theta(X \mid G)$ ▷ Reconstruct input data $X$ from $z_g$
    $\hat{Y}_{K'+1} \leftarrow$ Predict with classifier$(z_f)$
    Update the parameters of GMM-VGAE
**end for**

**Table 2. Datasets used for the experiments.**

| Dataset | Type | #Nodes | #Edges | #Classes | #Features |
|---|---|---|---|---|---|
| CiteSeer | Citation network | 3327 | 4732 | 6 | 3703 |
| Cora | Citation network | 2708 | 5429 | 7 | 1433 |
| Pubmed | Citation network | 19717 | 44338 | 3 | 500 |
| CoraFull | Citation network | 19793 | 65311 | 70 | 8710 |
| UAI2010 | Citation network | 3067 | 28311 | 19 | 4973 |
| Coauthor CS | Co-authorship | 18333 | 81894 | 15 | 6805 |
| Coauthor Physics | Co-authorship | 34493 | 247962 | 5 | 8415 |
| BlogCatalog | Social network | 5196 | 171743 | 6 | 8189 |
| Flickr | Social network | 7575 | 239738 | 9 | 12047 |

Microsoft Academic Graph in the KDD Cup 2016 challenge, nodes represent authors. Each edge signifies that two corresponding authors have collaborated in coauthoring an article.

In social network datasets, we use Flickr and BlogCatalog datasets. Flickr is a social network of the online photo-sharing platform where nodes show users and edges represent friendship among users through photo sharing. The labels demonstrate the interest groups of the users, and features are determined by a list of tags reflecting the interests of the users. BlogCatalog is a Social network with bloggers and their social connections from the BlogCatalog website where nodes' attributes are constructed by keywords generated by the users as a short description of the blogs. The labels represent the topic categories provided by the authors.

### 5- 1- Experimental Setup

In our GMM-VGAE, the first GNN has one graph convolutional layer with 32 units and the other networks have one convolutional layer with 16 hidden units. The activation function of all layers is $ReLU$ . We have trained GMM-VGAE for a maximum of 200 epochs using the Adam optimizer [23]. with an initial learning rate of 0.01, L2 regularization weight of $5\times10^{-4}$ and dropout rates of 0.6 for CiteSeer, Cora and Pubmed and 0.5 for other datasets. We stop training when validation accuracy does not increase, and announce the accuracy on the test set when the accuracy of the validation reaches its maximum. Every experiment is carried out ten times.

**Table 3. Node classification accuracies on the standard data splits of the three test datasets. The results are reported as mean accuracy ± standard deviation over multiple runs.**

| Method | Cora | CiteSeer | PubMed |
|---|---|---|---|
| GCN[3] | 81.5 | 70.3 | 79.0 |
| $G^3$NN[25] | 82.5 ± 0.2 | 74.4 ± 0.3 | 77.9 ± 0.4 |
| Eigen-GCN [26] | 78.9 ± 0.7 | 66.5 ± 0.3 | 78.6 ± 0.1 |
| GNN-LF/HF [27] | 84.0 ± 0.2 | 72.3 ± 0.3 | 80.5 ± 0.3 |
| OAGS[28] | 83.9 ± 0.5 | 73.7 ± 0.7 | 81.9 ± 0.9 |
| GAUG+GCN[32] | 83.6 | 73.11 | —- |
| JKNet[29] | 82.7 ± 0.4 | 73.0 ± 0.5 | 77.9 ± 0.4 |
| GCNII[30] | **85.5 ± 0.5** | 73.4 ± 0.6 | 80.2 ± 0.4 |
| ACMP[31] | 84.9 ± 0.6 | 74.5 ± 1.0 | 79.4 ± 0.4 |
| GM-VGAE[11] | 81.9 ± 0.7 | 72.1 ± 0.6 | 80.4 ± 0.3 |
| GMM-VGAE | 84.1 ± 0.4 | **74.6 ± 0.2** | **81.2 ± 0.1** |

### 5- 1- 1- Results and Analysis

The proposed model's performance is compared with that of state-of-the-art models. We utilize the fixed data splits from [24], as they represent the standard benchmark data splits, and the results of other methods are predominantly reported on these splits. In addition, an ablation study is undertaken to verify the effect of GMM-VGAE.

### 5- 1- 2- Comparing the State-of-the-art Methods

In this experiment, we adopt the fixed data splits recommended in [24]. For each class, 20 labelled nodes are chosen as the fixed split for training. The training sets for PubMed, CiteSeer, and Cora have sizes of 60, 120, and 140, respectively. For all datasets have identical validation and test sets, with 500 nodes for validation and 1000 nodes for testing.

The results of various methods on the standard splits are presented in Table 3. The numbers indicate the classification accuracies of the models on the test sets of the mentioned three datasets, and the best results are highlighted in bold. Additionally, we conduct a comparative analysis of various supervised methods, broadly classified into two categories. The first category encompasses shallow Graph Convolutional Network (GCN) methods, such as GCN [3], G3NN[25], Eigen-GCN [26], GNN-LF/HF [27] and OAGS [28]. Subsequently, we extend our comparison to include deep GCN methods, namely JKNet[29], GCNII[30], ACMP[31] and GM-VGAE[11] . The results of the aforementioned 9 competitive models are directly extracted from the related papers.

As evident from the results, our proposed model consistently outperforms the base-line GCN method, demonstrating improvements of 2.6%, 4.3%, and 2.2% on Cora, CiteSeer, and PubMed, respectively. These improvements reflect the differences compared to the basic GCN method. When compared to other advanced methods, such as GCNII and JKNet, the improvements, while present, are not as pronounced. Specifically, our model achieved improvements of 1.2% and 1.0% on CiteSeer and PubMed, respectively, compared to GCNII. However, it should be noted that for the Cora dataset, GCNII outperforms our model slightly by 1.4%.

Table 4 shows classification accuracies on Co-authorship datasets for some baseline methods and the proposed model. To guarantee a fair comparison with baselines, we execute 100 runs for random training/validation/test splits and use 20 labelled nodes per class for the training set, 30 nodes per class for the validation set, and the remaining nodes for the test set. The results of the baselines are obtained from [33]. We have divided all models into two categories: GNN variants ( GCN [3], MoNet[34], GAT[35], GraphSAGE[36], DAGNN[33], GraphMix[37]), and baseline methods (MLP, LogReg, LabelProp, LabelProp NL[38]).

From the results of Table 4, it can be concluded that GNN-based approaches which combine the structural information and feature information have better performances compared to the baseline methods that only consider the features or the structure. Among the GNN variants, GraphSAGE and GCN provide the same performance for almost all datasets. As evident from the results, our proposed model, GMM-VGAE, achieves the best performance on the Coauthor CS dataset. On

**Table 4. Classification accuracies on Co-authorship datasets.The results are reported as mean accuracy ± standard deviation over multiple runs.**

| Method | Cora | CiteSeer | PubMed |
|---|---|---|---|
| GCN[3] | 81.5 | 70.3 | 79.0 |
| $G^3$NN[25] | 82.5 ± 0.2 | 74.4 ± 0.3 | 77.9 ± 0.4 |
| Eigen-GCN [26] | 78.9 ± 0.7 | 66.5 ± 0.3 | 78.6 ± 0.1 |
| GNN-LF/HF [27] | 84.0 ± 0.2 | 72.3 ± 0.3 | 80.5 ± 0.3 |
| OAGS[28] | 83.9 ± 0.5 | 73.7 ± 0.7 | 81.9 ± 0.9 |
| GAUG+GCN[32] | 83.6 | 73.11 | —- |
| JKNet[29] | 82.7 ± 0.4 | 73.0 ± 0.5 | 77.9 ± 0.4 |
| GCNII[30] | **85.5 ± 0.5** | 73.4 ± 0.6 | 80.2 ± 0.4 |
| ACMP[31] | 84.9 ± 0.6 | 74.5 ± 1.0 | 79.4 ± 0.4 |
| GM-VGAE[11] | 81.9 ± 0.7 | 72.1 ± 0.6 | 80.4 ± 0.3 |
| GMM-VGAE | 84.1 ± 0.4 | **74.6 ± 0.2** | **81.2 ± 0.1** |

**Table 5. Node classification results on UAI, CoraFull datasets.**

| Dataset | UAI | | | | | | CoraFull | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| L/C | 20 | | 40 | | 60 | | 20 | | 40 | | 60 | |
| Metric | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 |
| DeepWalk | 42.0 | 32.9 | 51.2 | 46.0 | 54.3 | 44.4 | 29.23 | 28.05 | 36.23 | 33.2 | 46.6 | 37.9 |
| LINE | 43.7 | 37.0 | 45.3 | 39.6 | 51.0 | 43.7 | 17.7 | 18.2 | 25.0 | 25.4 | 29.6 | 30.8 |
| Cheby shev | 50.0 | 33.6 | 58.1 | 38.8 | 59.8 | 40.6 | 53.3 | 74.5 | 58.2 | 53.4 | 59.8 | 54.1 |
| GCN | 49.8 | 32.8 | 51.8 | 33.8 | 54.4 | 34.1 | 56.6 | 52.4 | 60.6 | 55.5 | 62.0 | 56.2 |
| KNN-GCN | 66.06 | 52.43 | 68.74 | 54.45 | 71.64 | 54.78 | | | | | | |
| AMGCN | 70.10 | 55.61 | 73.14 | 64.88 | 74.40 | 65.99 | | | | | | |
| GAT | 56.9 | 39.6 | 63.7 | 45.0 | 68.4 | 48.9 | 58.4 | 54.4 | 62.9 | 58.3 | 64.3 | 59.6 |
| DEMO Net | 23.4 | 16.8 | 30.2 | 26.3 | 34.1 | 29.0 | 54.5 | 50.4 | 60.2 | 56.2 | 61.5 | 57.2 |
| MixHop | 61.5 | 49.1 | 65.0 | 53.8 | 67.6 | **56.3** | 47.4 | 45.0 | 57.2 | 53.5 | 60.1 | 56.4 |
| GRACE | 65.54 | 48.38 | 66.67 | 49.50 | 68.68 | 51.51 | | | | | | |
| GMI | 60.69 | 46.75 | 63.14 | 49.10 | 64.73 | 44.36 | | | | | | |
| SLAPS | 46.82 | 41.60 | 34.62 | 25.28 | 62.51 | 51.81 | | | | | | |
| GCA | 72.55 | 56.97 | 73.27 | 54.55 | 73.60 | 56.00 | | | | | | |
| GM-VGAE | 71.0 | 55.4 | 72.1 | 53.1 | 71.4 | 53.9 | 58.3 | 54.2 | 62.6 | 57.3 | 63.4 | 58.9 |
| GMM-VGAE | **73.5** | **57.15** | **74.3** | **55.0** | **74.2** | **56.8** | **61.4** | **56.4** | **65.0** | **60.1** | **66.1** | **61.7** |

**Table 6. Node classification results on Flickr, and BlogCatalog datasets.**

| Dataset | Flickr | | | | | | BlogCatallog | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| L/C | 20 | | 40 | | 60 | | 20 | | 40 | | 60 | |
| Metric | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 | ACC | F1 |
| DeepWalk | 24.3 | 21.3 | 28.7 | 26.9 | 30.1 | 27.2 | 38.6 | 34.9 | 50.8 | 48.6 | 55.0 | 53..5 |
| LINE | 33.2 | 31.1 | 36.6 | 37.1 | 38.5 | 37.7 | 58.7 | 57.7 | 61.1 | 60.7 | 64.5 | 63.8 |
| Cheby shev | 23.2 | 21.2 | 35.1 | 33.5 | 41.7 | 40.1 | 38.0 | 33.3 | 56.2 | 53.8 | 70.0 | 68.3 |
| GCN | 41.4 | 39.9 | 45.4 | 43.2 | 47.9 | 46.5 | 69.8 | 68.7 | 71.2 | 70.7 | 72.6 | 71.8 |
| kNN-GCN | 69.28 | 70.33 | 75.08 | 75.40 | 77.94 | 77.97 | 75.49 | 72.53 | 80.84 | 80.16 | 82.46 | 81.90 |
| AMGCN | 75.26 | 74.63 | 80.06 | 79.36 | 82.10 | 81.81 | 81.89 | 81.36 | 84.94 | 84.32 | 87.30 | 86.94 |
| GAT | 38.5 | 37.0 | 38.4 | 36.9 | 38.9 | 37.3 | 64.0 | 63.3 | 67.4 | 66.3 | 69.9 | 69.0 |
| DEMO Net | 34.8 | 33.5 | 46.5 | 45.2 | 57.3 | 56.4 | 54.1 | 52.7 | 63.4 | 63.0 | 76.8 | **76.7** |
| MixHop | 39.5 | 40.1 | **55.1** | **56.2** | **64.9** | **65.7** | 65.4 | 64.8 | 71.6 | 70.8 | **77.4** | 76.3 |
| GRACE | 49.42 | 48.18 | 53.64 | 52.61 | 55.67 | 54.61 | 76.56 | 75.56 | 76.66 | 75.88 | 77.66 | 77.08 |
| GMI | 49.17 | 28.43 | 52.74 | 30.94 | 53.78 | 31.50 | 66.46 | 39.2 | 68.01 | 40.42 | 72.59 | 43.24 |
| SLAPS | **72.20** | **72.48** | **79.00** | **78.90** | **76.20** | **76.50** | 87.80 | 87.34 | 88.50 | 87.57 | **89.50** | **89.22** |
| GCA | 63.44 | 63.26 | 63.90 | 64.60 | 64.43 | 64.64 | 80.51 | 81.28 | 84.89 | 84.04 | 86.34 | 86.19 |
| GM-VGAE | 58.4 | 61.5 | 61.8 | 62.9 | 63.5 | 62.8 | 87.1 | 86.5 | 87.6 | 86.5 | 84.3 | 84.0 |
| GMM-VGAE | 64.3 | 66.6 | 64.0 | 65.0 | 66.4 | 65.4 | **88.9** | **88.5** | **89.1** | **88.1** | 86.9 | 86.4 |

**Table 7. The results of different values of the hyperparameter K for GMM-VGAE.**

| Dataset | k=1 | k=2 | k=3 | k=4 | k=5 | k=6 | k=7 |
|---|---|---|---|---|---|---|---|
| Cora | 0.833 | **0.844** | 0.8303 | 0.8395 | 0.8376 | 0.831 | 0.830 |
| Citeseer | 0.733 | 0.741 | **0.746** | 0.707 | 0.716 | 0.720 | 0.722 |
| Pubmed | 0.801 | 0.809 | **0.812** | 0.80 | 0.798 | 0.792 | 0.797 |

the Coauthor Physics dataset, while GMM-VGAE performs exceptionally well, GraphMix slightly outperforms it with an accuracy of $94.4 \pm 0.8\%$ compared to GMM-VGAE's $94.5 \pm 0.9\%$

To assess the performance of our model in greater detail, the effect of different numbers (rates) of training nodes on our model is investigated on four other datasets.

We chose 1000 nodes as the test and three label rates (20, 40, and 60 labelled nodes per class) for the training set from [39]. The same parameters specified by the arti- cles' authors are used to initialize all baselines. Tables 5 and 6 show classification accuracies and F1 scores where L/C refers to the number of labeled nodes per class. We compare GMM-VGAE with seven graph neural network algorithms (Chebyshev [40], GCN[3],kNN-GCN [39], GAT [35], DEMO-Net[41], MixHop[42], AMGCN[39]), two network embedding methods (DeepWalk[43], LINE[44]) and four self-supervised models (GRACE[45], GCA [46], GMI [47], SLAPS [48]).

As the results show, the proposed model almost gets the best performance on all datasets with all label rates.

### 5- 1- 3- Further analysis

Table 5 summarizes the results of the proposed model for different values of the hyperparameter $K$ in node classification tasks. As can be seen, the best value of $K$ depends on the target dataset. For example, the best value of $K$ for Cora ,Citetseer and Pubmed are 2, 3, and 3, respectively. It is clear from the results, that GMMs ( $K > 1$ ) outperform the single Gaussian model ( $K = 1$ ) and the best results for citation datasets are obtained for $K = 2$ and $K = 3$.

## 6- Conclusions

In this paper, we proposed an efficient framework, namely GMM-VGAE, for node classification in graphs by combining Graph Convolutional Networks and Variational Autoencoders. By assuming the Gaussian mixture models as the prior distribution of VGAE to capture the inherent complex data distributions, the GMM-VGAE profits from both labelled and unlabeled data to learn continuous latent representations for the nodes. The classification losses of the model improve the separation of the classes in the latent space while the ELBO regularize and smooth this space.

## Competing Interests

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.

## Authors' Contribution Statement

Mohadeseh ghayekhloo: Data curation, Writing- Original draft preparation, Software, Validation, Visualization, Investigation, Conceptualization, Methodology, Software Ahmad Nickabadi: Writing- Reviewing and Editing, Visualization, Investigation, Supervision.

## Ethical and Informed Consent for Data Used

The present study involving human subjects strictly adhered to ethical guidelines. Informed consent was obtained from all participants involved in the study, specifically for tasks related to graph-based representation learning and node classification. For the purpose of this research, all data used were anonymized and handled in compliance with data protection regulations. Confidentiality and privacy of the participants were maintained throughout the study.

## Data Availability Statement

The data used in this study, the "Cora, Citeseer and Pubmed dataset" are available at the following public repository: https://github.com/tkipf/gcn/tree/master/gcn/data.

## Funding declaration

The authors state that no funding is involved.

## References

[1] Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., & Zhang, C. (2018). Adversarially regularized graph autoencoder for graph embedding. arXiv preprint arXiv:1802.04407.

[2] Kipf, T. N., & Welling, M. (2016). Variational graph auto-encoders. arXiv preprint arXiv:1611.07308.

[3] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

[4] Wang, C., Pan, S., Long, G., Zhu, X., & Jiang, J. (2017). MGAE: Marginalized graph autoencoder for graph clustering, 889–898.

[5] Li, F., Zhu, Z., Zhang, X., Cheng, J., & Zhao, Y. (2019). Diffusion-induced graph representation learning. Neurocomputing, 360, 220–229.

[6] Park, J., Lee, M., Chang, H. J., Lee, K., & Choi, J. Y. (2019). Symmetric graph convolutional autoencoder for unsupervised graph representation learning, 6519–6528.

[7] Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In International Conference on Machine Learning (pp. 1278–1286). PMLR.

[8] Xie, Q., Huang, J., Du, P., & Peng, M. (2021). Graph relational topic model with higher-order graph attention auto-encoders, 2604–2613.

[9] He, L., Bai, L., Yang, X., Du, H., & Liang, J. (2023). High-order graph attention network. Information Sciences, 630, 222–234.

[10] Li, B., Jiao, P., He, D., Zhang, W., & 0001, D. J. (2019). Network-specific variational autoencoder for embedding in attribute networks, 2663–2669.

[11] Niknam, G., Molaei, S., Zare, H., Clifton, D., & Pan, S. (2023). Graph representation learning based on deep generative Gaussian mixture models. Neurocomputing, 523, 157–169.

[12] Davidson, T. R., Falorsi, L., De Cao, N., Kipf, T., & Tomczak, J. M. (2018). Hyperspherical variational auto-encoders. arXiv preprint arXiv:1804.00891.

[13] Zheng, S., Zhu, Z., Zhang, X., Liu, Z., Cheng, J., & Zhao, Y. (2020). Distribution-induced bidirectional generative adversarial network for graph representation learning, 7224–7233.

[14] Dilokthanakul, N., Mediano, P. A., Garnelo, M., Lee, M. C., Salimbeni, H., Arulkumaran, K., & Shanahan, M. (2016). Deep unsupervised clustering with Gaussian mixture variational autoencoders. arXiv preprint arXiv:1611.02648.

[15] Kingma, D. P., & Welling, M. (2013). Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114.

[16] Bojchevski, A., & Günnemann, S. (2017). Deep Gaussian embedding of graphs: Unsupervised inductive learning via ranking. arXiv preprint arXiv:1707.03815.

[17] Namata, G., London, B., Getoor, L., & Huang, B. (2012). Query-driven active surveying for collective classification, 8(1).

[18] Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., & Eliassi-Rad, T. (2008). Collective classification in network data. AI Magazine, 29(3), 93–93.

[19] Wang, W., Liu, X., Jiao, P., Chen, X., & Jin, D. (2018). A unified weakly supervised framework for community detection and semantic matching, 218–230. Springer.

[20] Shchur, O., Mumme, M., Bojchevski, A., & Günnemann, S. (2018). Pitfalls of graph neural network evaluation. arXiv preprint arXiv:1811.05868.

[21] Tang, L., & Liu, H. (2009). Relational learning via

latent social dimensions, 817–826.

[22] Huang, X., Li, J., & Hu, X. (2017). Label informed attributed network embedding, 731–739.

[23] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[24] Yang, Z., Cohen, W., & Salakhudinov, R. (2016). Revisiting semi-supervised learning with graph embeddings, 40–48. PMLR.

[25] Ma, J., Tang, W., Zhu, J., & Mei, Q. (2019). A flexible generative framework for graph-based semi-supervised learning. Advances in Neural Information Processing Systems, 32.

[26] Zhang, Z., Cui, P., Pei, J., Wang, X., & Zhu, W. (2021). Eigen-GNN: A graph structure preserving plug-in for GNNs. IEEE Transactions on Knowledge and Data Engineering.

[27] Zhu, M., Wang, X., Shi, C., Ji, H., & Cui, P. (2021). Interpreting and unifying graph neural networks with an optimization framework. In Proceedings of the Web Conference 2021 (pp. 1215–1226).

[28] Song, Z., Zhang, Y., & King, I. (2022). Towards an optimal asymmetric graph structure for robust semi-supervised node classification. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 1656–1665).15 161616

[29] Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., & Jegelka, S. (2018). Representation learning on graphs with jumping knowledge networks. In International Conference on Machine Learning (pp. 5453–5462). PMLR.

[30] Chen, M., Wei, Z., Huang, Z., Ding, B., & Li, Y. (2020). Simple and deep graph convolutional networks. In International Conference on Machine Learning (pp. 1725–1735). PMLR.

[31] Wang, Y., Yi, K., Liu, X., Wang, Y. G., & Jin, S. (2022). ACMP: Allen-Cahn message passing with attractive and repulsive forces for graph neural networks. In The Eleventh International Conference on Learning Representations.

[32] Zhao, T., Liu, Y., Neves, L., Woodford, O., Jiang, M., & Shah, N. (2021). Data augmentation for graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, 35, 11015–11023.

[33] Liu, M., Gao, H., & Ji, S. (2020). Towards deeper graph neural networks, 338–348.

[34] Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., & Bronstein, M. M. (2017). Geometric deep learning on graphs and manifolds using mixture model CNNs, 5115–5124.

[35] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. stat, 1050, 20.

[36] Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, 30.

[37] Verma, V., Qu, M., Kawaguchi, K., Lamb, A., Bengio, Y., Kannala, J., & Tang, J. (2021). GraphMix: Improved training of GNNs for semi-supervised learning. In Proceedings of the AAAI Conference on Artificial Intelligence, 35, 10024–10032.

[38] Chapelle, O., Scholkopf, B., & Zien, A. (2009). Semi-supervised learning (Chapelle, O. et al., eds.; 2006)[book reviews]. IEEE Transactions on Neural Networks, 20(3), 542–542.

[39] Wang, X., Zhu, M., Bo, D., Cui, P., Shi, C., & Pei, J. (2020). AM-GCN: Adaptive multi-channel graph convolutional networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 1243–1253).

[40] Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. Advances in Neural Information Processing Systems, 29.

[41] Wu, J., He, J., & Xu, J. (2019). Net: Degree-specific graph neural networks for node and graph classification, 406–415.

[42] Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Ver Steeg, G., & Galstyan, A. (2019). MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing, 21–29. PMLR.

[43] Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). DeepWalk: Online learning of social representations, 701–710.

[44] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). LINE: Large-scale information network embedding, 1067–1077.

[45] Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., & Wang, L. (2020). Deep graph contrastive representation learning. arXiv preprint arXiv:2006.04131.

[46] Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., & Wang, L. (2021). Graph contrastive learning with adaptive augmentation. In Proceedings of the Web Conference 2021 (pp. 2069–2080).

[47] Peng, Z., Huang, W., Luo, M., Zheng, Q., Rong, Y., Xu, T., & Huang, J. (2020). Graph representation learning via graphical mutual information maximization. In Proceedings of The Web Conference 2020 (pp. 259–270).

[48] Fatemi, B., El Asri, L., & Kazemi, S. M. (2021). SLAPS: Self-supervision improves structure learning for graph neural networks. Advances in Neural Information Processing Systems, 34, 22667–22681.

**HOW TO CITE THIS ARTICLE**
*M. Ghayekhlou, A. Nickabadi, A Guassian Mixture Variational Graph Autoencoder for Node Classification, AUT J. Model. Simul., 56(1) (2024) 103-116.*

**DOI:** 10.22060/miscj.2024.23107.5358