Learning Disentangled Graph Convolutional Networks Locally and Globally

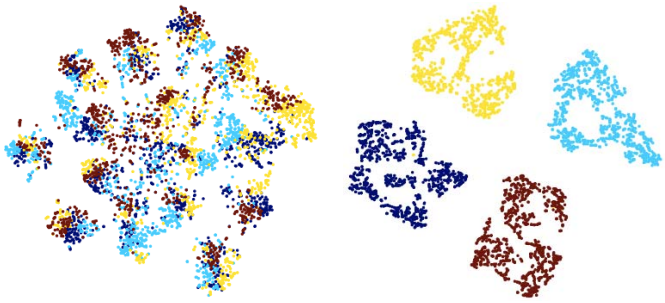
Jingwei Guo, Kaizhu Huang, Xinping Yi, and Rui Zhang

***Abstract*— Graph convolutional networks (GCNs) emerge as the most successful learning models for graph-structured data. Despite their success, existing GCNs usually ignore the entangled latent factors typically arising in real-world graphs, which results in nonexplainable node representations. Even worse, while the emphasis has been placed on local graph information, the global knowledge of the entire graph is lost to a certain extent. In this work, to address these issues, we propose a novel framework for GCNs, termed LGD-GCN, taking advantage of both local and global information for disentangling node representations in the latent space. Specifically, we propose to represent a disentangled latent continuous space with a statistical mixture model, by leveraging neighborhood routing mechanism *locally*. From the latent space, various new graphs can then be disentangled and learned, to overall reflect the hidden structures with respect to different factors. On the one hand, a novel regularizer is designed to encourage *interfactor diversity* for model expressivity in the latent space. On the other hand, the factor-specific information is encoded *globally* via employing a message passing along these new graphs, in order to strengthen *intrafactor consistency*. Extensive evaluations on both synthetic and five benchmark datasets show that LGD-GCN brings significant performance gains over the recent competitive models in both disentangling and node classification. Particularly, LGD-GCN is able to outperform averagely the disentangled state-of-the-arts by 7.4% on social network datasets.**

***Index Terms*— (Semi-)supervised node classification, disentangled representation learning, graph convolutional networks (GCNs), local and global learning.**

I. Introduction

GRAPHS are emerging as an insightful structured modeling technique for capturing the similarity between data samples and identifying the relationship between entities [1]–[5]. To mine the domain-specific knowledge in graph-structured data, graph convolutional networks (GCNs) were proposed to integrate topological patterns and content features for node classification [6]. In the past years, GCNs have demonstrated excellent expressive power that leads to the growing popularity in various graph learning tasks, such as node classification, link prediction, and recommendation [7]–[9].



(a) (b)

Fig. 1. Visualization of the disentangled latent units w.r.t. four latent factors on a synthetic graph. Several works have been made toward graph disentanglement learning, e.g., DisenGCN [17] and independence promoted graph disentangled network (IPGDN) [18]. They are all heavily relying on the local graph information, and here, we only take DisenGCN, the basic one, as an example for intuitive comparison. Points with a different color indicate the disentangled latent units (for all nodes) of a different latent factor. In sharp contrast to DisenGCN, our LGD-GCN displays a highly disentangled pattern with strong interfactor diversity and intrafactor consistency; it indicates high (low) correlations between intrafactor (interfactor) features. (a) DisenGCN (local approach). (b) LGD-GCN (Ours).

Notably, most existing GCN models in the literature [6], [10]–[13] focus on exploiting the local graph information and take a holistic approach, i.e., they interpret the node neighborhood as a perceptual whole while ignoring the within distinctions. Yet a real-world graph typically contains heterogeneous node relations, driven by the entanglement of many latent factors. For example, a user in a social network usually links with others for various reasons, such as family, work, and/or hobby, which typically stores partial information in different types. The holistic approaches fail to capture the expressive partial information due to the neglect of the underlying factors, thereby rendering the learned representations heavily entangled and less informative. On the other hand, while the benefits of modeling data both locally and globally have been well demonstrated in various machine learning models [14], there are a few variants of GCNs (e.g., [15], [16]) incorporating both local and global graph information.

Recently, several works [17]–[21] make attempts to disentangle the latent factors behind graph data through neighborhood partition. Despite the novel design, they mostly rely

Manuscript received 26 June 2021; revised 26 March 2022; accepted 23 July 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61876155; in part by the Jiangsu Science and Technology Program under Grant BE2020006-4; in part by the Key Program Special Fund in XJTLU under Grant KSF-T-06. *(Corresponding author: Kaizhu Huang.)*

Jingwei Guo and Xinping Yi are with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, U.K. (e-mail: jingwei.guo@liverpool.ac.uk; xinping.yi@liverpool.ac.uk).

Kaizhu Huang is with the Department of Electrical and Computer Engineering, Duke Kunshan University, Suzhou 215316, China (e-mail: kaizhu.huang@dukekunshan.edu.cn).

Rui Zhang is with the Department of Foundational Mathematics, Xi'an Jiaotong–Liverpool University, Suzhou 215123, China (e-mail: rui.zhang02@xjtlu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3195336.

Digital Object Identifier 10.1109/TNNLS.2022.3195336

on local node neighborhoods only, similar to most GCNs, which may bring unexpected issues. First, the information from local ranges can be significantly varied across the entire graph. Solely depending on it, they could easily produce latent representations that lose consensus cluster centroids with respect to different factors. This consequently may weaken the intrafactor correlation and interfactor separability between disentangled features, therefore leading to diminished interpretability. Second, the local neighborhood information can be scarce and limited, especially in sparse graphs, which prohibits models from learning informative node aspects and yielding a favorable performance boost. A detailed discussion will be given later in Section II-C.

In this work, to address above-mentioned issues, we propose a novel local and global disentanglement for GCNs (LGD-GCN). The core idea is that we learn disentangled node representations by mining both local and global graph information. In particular, we first present a local disentanglement on nodes by partitioning their observable neighbors with the neighborhood routing mechanism. Then, the global information is attained by modeling the overall densities of nodes while considering different factors and further disclosing the hidden node relations from different angles.

To this end, a statistical mixture modeling is performed on disentangled latent units to derive a latent continuous space. This enables a different density covering all the nodes, specific to a latent factor, in a different subspace [22], [23]. Accordingly, a novel regularizer is developed for promoting *interfactor diversity*. It encourages the separability between these latent units according to different factors and captures the uncorrelated information. After that, we manage to build a different new graph with sparse properties by only connecting nearby neighbors within a different spatial region. These new graphs overall reflect the underlying data structures, i.e., the hidden node relations, in different aspects. Employing a message passing scheme over them can efficiently encode the global information specific to different factors. This further strengthens *intrafactor consistency*, i.e., the correlation between disentangled features w.r.t. the same factor. Therefore, the disentangled informativeness of the model can be enhanced in the output space. In sharp contrast to the local approach for graph disentanglement, Fig. 1 clearly visualizes the benefit of learning disentangled node representations *both locally and globally*.

In a nutshell, our contributions are summarized as follows.

1) We show through empirical analysis that the existing disentangled approaches may produce latent representations with weakly disentangled factors when solely relying on the local graph information. Therefore, the performance gain of these disentangled approaches becomes marginal when it comes to sparse graphs.
2) To overcome the above-mentioned limitations, we propose a novel framework for GCNs (LGD-GCN) to disentangle the latent factors underlying the graph data in a more effective way. Specifically, by leveraging neighborhood routing *locally* and message passing *globally*, LGD-GCN can disentangle node representations with promoted *interfactor diversity* and strengthened *intrafactor consistency*.
3) Extensive evaluations of synthetic and five real-world datasets show the superiority of the proposed LGD-GCN over the state-of-the-arts both quantitatively and qualitatively. Specifically, LGD-GCN averagely outperforms the disentangled state-of-the-arts by 7.5%, 7.2%, 1.7%, 2.3%, and 1.6% on Blogcatalog, Flickr, Cora, Citeseer, and Pubmed datasets, respectively.

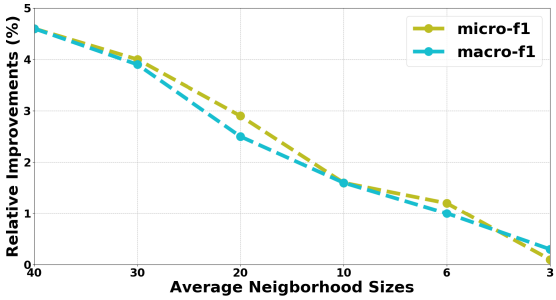


Fig. 2. Relative improvements of DisenGCN upon GCN while varying the average neighborhood sizes of the graph synthesized with four latent factors. It can be observed that the boost performance of DisenGCN is becoming minor as the average neighborhood sizes decrease.

II. Background and Motivation

A. Graph Convolutional Networks

GCNs are powerful machine learning models for tackling analytical tasks on graph-structured data [7]. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denotes a graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$. Given two distinct nodes $i, j \in \mathcal{V}$, we define $(i, j) \in \mathcal{E}$ if nodes i and j are connected by an edge, and the neighborhood of node i as $N_i = \{j|(i, j) \in \mathcal{E}\}$. For attributed graphs, nodes are associated with a raw feature matrix $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}|\times f}$ with the transpose of each raw being the initial features $\mathbf{h}_i \in \mathbb{R}^f$ for each node i, where f is the number of raw features per node. In the past years, an increasing number of GCN variants have been developed, which can be divided into two categories, including spectral-based [6], [24], [25] and spatial-based [10], [11], [26]–[28] methods. Most of them can be generalized with a message passing framework [29], where node attributes are exchanged locally through the edges followed by a neighborhood aggregation

$$\mathbf{h}_i' \leftarrow \mathbf{COMBINE}(\mathbf{h}_i, \mathbf{AGGREGATE}(\{\mathbf{h}_j|\forall j \in N_i\})).$$

The features of neighbors are first aggregated together and then passed to the center node to update its hidden states $\mathbf{h}_i$ through combination. Albeit promising in several learning tasks, most GCNs focus on modeling local neighborhoods only and treat them as a perceptual whole. Consequently, the learned node representations usually lose the global knowledge of the entire graph and tend to be nonexplainable.

B. Disentangled Node Representation Learning

Disentangled representation learning aims to reveal the explanatory latent variables behind data for generating a meaningful representation to admit intuitive explanations [30], [31]. Massive works have been developed on that topic [32]–[36].

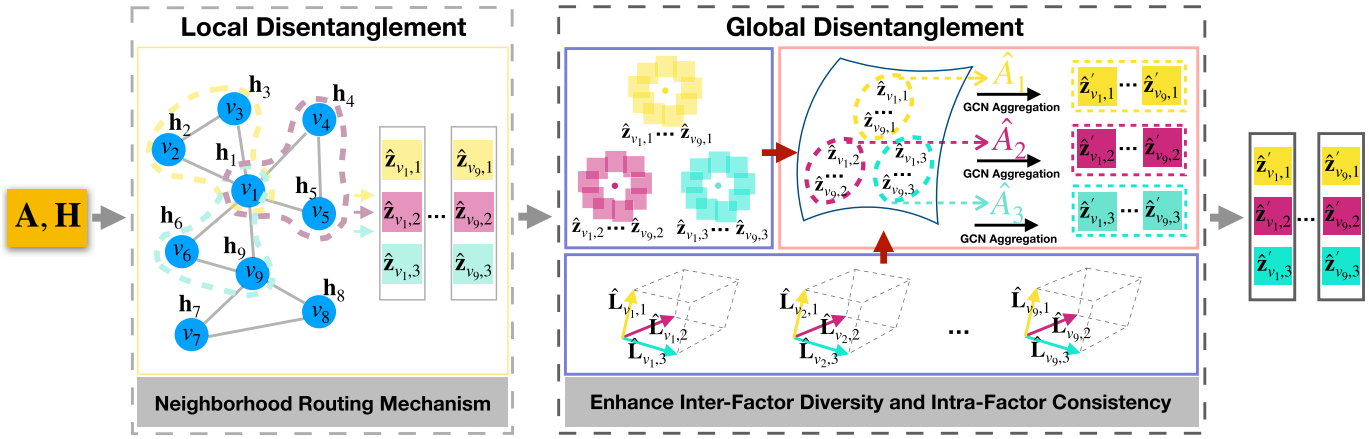


Fig. 3. Illustration of our LGD-GCN layer with $M = 3$ latent factors, where $\mathbf{A}$ and $\mathbf{H}$ denote the adjacency matrix and feature matrix of the input graph, respectively. First, the node representations are locally disentangled by leveraging the neighborhood routing mechanism. These disentangled representations are then modeled in a latent continuous space, promoted with interfactor diversity, from which various new graphs are built for further aggregation to strengthen intrafactor consistency.

It has been proven that disentangled representation is less susceptible to complex variants and more robust to adversarial attacks [30], [37]. However, most works are only applicable to the Euclidean data structure. Recently, DisenGCN [17] made the first attempt toward disentangled node representation learning. Since then, works [18]–[21] extending it start to emerge in the field of graph learning.

DisenGCN [17] hypothesizes that nodes on the graph are connected mainly due to different kinds of relationship caused by different factors m ($m = 1, 2, \ldots, M$). It aims to identify these latent factors in order to learn disentangled node representations. Specifically, the node features, $\{\mathbf{h}_i \in \mathbb{R}^f | \forall i \in V\}$, are first projected onto M subspaces in different channels. In each channel m, the hidden states $\mathbf{z}_{i,m} \in \mathbb{R}^{d/M}$ for each node i is given by

$$\mathbf{z}_{i,m} = \frac{\sigma\left(\mathbf{W}_m^T \mathbf{h}_i + \mathbf{b}_m\right)}{\left\|\sigma\left(\mathbf{W}_m^T \mathbf{h}_i + \mathbf{b}_m\right)\right\|_2} \tag{1}$$

where $\mathbf{W}_m \in \mathbb{R}^{f \times (d/M)}$ and $\mathbf{b}_m \in \mathbb{R}^{d/M}$ are learnable parameters, and σ is an activation function. Then, DisenGCN employs a neighborhood routing mechanism [17, Algorithm-1] (denoted as NRM in this article) to partition each node neighborhood into M clusters, from which the information is aggregated independently to produce disentangled latent units, e.g., $\{\hat{\mathbf{z}}_{i,1}, \hat{\mathbf{z}}_{i,2}, \ldots, \hat{\mathbf{z}}_{i,M}\}$ for node i, describing different node aspects. Finally, the disentangled node representation, $\hat{\mathbf{h}}_i \in \mathbb{R}^d$, can be attained by vector column concatenation

$$\hat{\mathbf{h}}_i = \hat{\mathbf{z}}_{i,1} \oplus \hat{\mathbf{z}}_{i,2} \oplus \cdots \oplus \hat{\mathbf{z}}_{i,M}. \tag{2}$$

C. Limitations of Current Disentangled Approaches

While the current disentangled state-of-the-arts reveal certain latent factors and demonstrate good performance in many scenarios [19]–[21], we argue that they are prone to produce weakly disentangled representations and yield limited performance boost because of their heavy reliance on local graph information. To illustrate, we conduct two empirical investigations over a graph synthesized with four latent factors (see details in Section IV-A2 for data generation). For simplification, we only take DisenGCN [17] as an example, because the other disentangled state-of-the-arts are mainly built on it.

First, we visualize the disentangled latent units of DisenGCN using t-stochastic neighbor embedding (SNE) [38] in Fig. 1(a). At the microlevel, we can observe the separability between points with different colors in some regions. When it comes to the macrolevel, all points unexpectedly fall into discrete clusters and are mixed together, indicating a weak disentanglement. As DisenGCN generates disentangled latent units that preserve some specific micromeanings of the factor but lose the consistent macromeaning (*intrafactor consistency*), this limits its potential in attaining higher performance. Additionally, DisenGCN only considers disentangling representations in different channels without ensuring their diversity w.r.t. different factors (*interfactor diversity*). The learned representations are thus prone to preserve redundant information, as illustrated in Fig. 1(a).

Second, we further augment the synthetic graph by tuning the p value (which controls the density of the synthetic graph as described in Section IV-A2) to generate graphs with different average neighborhood sizes. We then train GCN [6], the typical holistic approach, and DisenGCN for multilabel classification, and plot the relative improvements of DisenGCN upon GCN in Fig. 2. From Fig. 2, the improvements drop from approximately 5% to 0% as the average neighborhood size decreases from 40 to 3. The results are consistent with the theoretical analysis in that DisenGCN relies on local information only. When the graph is sparse (with limited local information), DisenGCN is inevitably getting less effective.

D. Graph Structure Learning

Structure information plays a key role in graph learning and GCNs. Unfortunately, real-world data are typically noisy and incomplete, and hence, tend to generate imperfect or even poor graph structures [39]. To address this issue, many researchers try to remove noisy edges and infer the hidden relations between nodes so that an appropriate graph structure

can be better learned [40]. These studies can generally be divided into two categories. The first one takes the approach of metric learning. It aims to learn a metric to decide whether two nodes are connected or not based on their features, and update the original structure with the new one by, e.g., interpolation. Representative ideas include [24], [41]–[43]. The other one assumes that graphs are generated by sampling edges from a certain distribution [44]–[46]. These works focus on probabilistic modeling and train graph neural networks with the sampled graphs. Our work focuses on inferring the underlying node relations in the disentangled feature space, thereby falling into the group of metric learning. Another work related to ours is FactorGCN [47], which factorizes a graph into multiple subgraphs by edge clipping and uses them to learn graph-level representation from different angles. Different from it, our approach generates multiple new graphs from a latent space and employs a message passing along them to characterize nodes in different aspects.

III. Local and Global Disentanglement

We present a novel framework for GCNs, termed LGD-GCN, to learn disentangled node representations both *locally and globally*, as presented in Fig. 3. By leveraging the neighborhood routing mechanism [17] first, we attain disentangled latent units preserving *local* graph information w.r.t. different factors. Then, we propose to further incorporate *global* graph information. To this end, our LGD-GCN discloses the underlying factor-aware relations between nodes and utilizes them to learn better disentangled representations with promoted *interfactor diversity* and strengthened *intrafactor consistency*. All the details are illustrated in the following subsections with notations defined in Table I for easy reading.

TABLE I
Common Notations

Notations	Descriptions
$\|\cdot\|_2$	L2 norm.
$\mathcal{P}(\cdot)$	Pairwise distance.
$\|\cdot\|$	Cardinality of a set.
$\oplus$	Vector column concatenation.
$\mathcal{V}$	The node set.
$\mathcal{E}$	The edge set.
$\mathcal{V}_{trn}$	The training node set.
$\mathcal{V}_{val}$	The validation node set.
$\mathcal{V}_{tst}$	The testing node set.
N_i	The neighborhood set of node i.
f	The dimension of initial node feature vector.
d	The dimension of hidden node feature vector.
C	The total number of class.
$\mathbf{A} \in \mathbb{R}^{\|V\|\times\|V\|}$	The initial adjacency matrix.
$\mathbf{H} \in \mathbb{R}^{\|V\|\times f}$	The initial feature matrix.
$\mathbf{h}_i \in \mathbb{R}^f$	The initial features of node i.
$\mathbb{Y}_i \in \mathbb{R}^C$	The ground truth label of node i in one hot encoding.
$\mathbf{Y}_i \in \mathbb{R}^C$	The predicted label of node i in one hot encoding.
M	The total number of assumed factors (channels).
NRM	The neighborhood routing mechanism [17, Algorithm-1].
T	The number of routing iterations in NRM.
$\mathcal{N}(;\mu,\Sigma)$	Gaussian distribution with mean μ and covariance Σ.
$\boldsymbol{\mu}_m \in \mathbb{R}^{\frac{d}{M}}$	Mean vector given factor m.
$\boldsymbol{\Sigma}_m \in \mathbb{R}^{\frac{d}{M}\times\frac{d}{M}}$	Covariance matrix given factor m.
$\mathbf{z}_{i,m} \in \mathbb{R}^{\frac{d}{M}}$	The latent units of node i given factor m.
$\mathbf{A}_m \in \mathbb{R}^{\|V\|\times\|V\|}$	The learned adjacency matrix given factor m.
$\mathcal{L}_{space}$	The regularizer for space modeling.
$\mathcal{L}_{div}$	The regularizer for promoting factor diversity.
$\mathcal{L}_t$	The loss for downstream task.
k	The density parameter in k-Nearest-Neighbor algorithm.
L	The total number of stacked layer.
U_r	The update rate for $\{\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m \mid \forall m = 1, 2, ..., M\}$.
λ_{space}	The regularization coefficient for $\mathcal{L}_{space}$.
λ_{div}	The regularization coefficient for $\mathcal{L}_{div}$.
$\mathbf{F}_{\theta^{(l)}}$	The l^{th} LGD-GCN's layer with learnable parameters $\theta^{(l)}$.
$\mathbb{LG}$	The module of latent aggregation.

A. Modeling Latent Continuous Space

We assume that the disentangled latent unit $\hat{\mathbf{z}}$ follows a Gaussian mixture distribution expressed as

$$p(\hat{\mathbf{z}}) = \sum_{m=1}^{M} q(m)\mathcal{N}(\hat{\mathbf{z}}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$$

where $\boldsymbol{\mu}_m \in \mathbb{R}^{d/M}$ and $\boldsymbol{\Sigma}_m \in \mathbb{R}^{(d/M)\times(d/M)}$ are the mean and covariance associated with factor m in latent space, and $q(m)$ is the prior probability of factor m and set as $(1/M)$ for equal consideration. To employ this assumption for space modeling, we maximize the conditional likelihood of disentangled latent units given their associated factor, i.e., $p(\hat{\mathbf{z}}_{i,m}|m)$ for each node i and factor m. It turns out to be equivalent to minimizing the negative log term by removing constants

$$\mathcal{L}_{i,m}^{\text{space}} = (\hat{\mathbf{z}}_{i,m} - \boldsymbol{\mu}_m)^T \boldsymbol{\Sigma}_m^{-1} (\hat{\mathbf{z}}_{i,m} - \boldsymbol{\mu}_m). \tag{3}$$

Its functionality is quite similar to some supervised embedding methods [23], [48]–[52] that collapse categories into a low-dimensional embedding with the reduced within-class pairwise distance. Instead of constraining pairwise samples, we introduce the Mahalanobis distance [53] between each latent unit $\hat{\mathbf{z}}_{i,m}$ and its globally inferred center $\boldsymbol{\mu}_m$. Accordingly, a latent continuous space can be derived with more compact data (sub-)manifolds, where the latent units are encouraged to be more discriminative with respect to their factor density.

Finally, the regularization for space modeling is given by averaging over nodes and factors

$$\mathcal{L}_{\text{space}} = \frac{1}{|V|M} \sum_{i\in V} \sum_{m=1}^{M} \mathcal{L}_{i,m}^{\text{space}}. \tag{4}$$

B. Promoting Interfactor Diversity

Diversity-promoting learning aims to encourage different components in latent space models to stay mutually uncorrelated and different and has been widely studied [54], [55]. In Section III-A, we have derived a latent continuous space by independently modeling the density of each disentangled factor. However, the approximated distributions with different factors could still be overlapped [54]. Consequently, the disentangled latent units may preserve redundant information and lose informativeness. To cope with this problem, we propose to promote diversity among different latent factors in order to capture the uncorrelated information.

Particularly, we define factor diversity with respect to the probabilities of a sampled latent unit staying close to different factor densities. Inspired by the determinant point process [56],

Algorithm 1 LGD-GCN's Layer

Input: $\{\mathbf{h}_i \in \mathbb{R}^{d_{in}} | \forall i \in \mathcal{V}\}$, where $d_{in} = f$ in the first layer and $d_{in} = d$ in the hidden layer.
Output: $\{\mathbf{h}'_i \in \mathbb{R}^d | \forall i \in \mathcal{V}\}$.
for $i \in \mathcal{V}$ **do**
  $\mathbf{z}_{i,1}, \mathbf{z}_{i,2}, \ldots, \mathbf{z}_{i,M} \leftarrow \mathbf{h}_i$ by Eq. (1).
end for
for $i \in \mathcal{V}$ **do**
  // Leverage NRM with T routing iterations.
  $\hat{\mathbf{z}}_{i,m} \leftarrow \{\mathbf{z}_{i,m}\} \cup \{\mathbf{z}_{j,m} | \forall j \in N_i\}, \forall m = 1, 2, \ldots, M.$
end for
// Promoting Inter-factor Diversity.
Minimize $\mathcal{L}_{space}$ and $\mathcal{L}_{div}$ by Eq. (4) and Eq. (6).
// Strengthening Intra-factor Consistency.
for $m = 1, 2, \ldots, M$ **do**
  $\mathbf{A}_m \leftarrow \{\hat{\mathbf{z}}_{i,m} | \forall i \in \mathcal{V}\}$ by k-Nearest-Neighbors.
  $\{\hat{\mathbf{z}}'_{i,m} | \forall i \in V\} \leftarrow \{\hat{\mathbf{z}}_{i,m} | \forall i \in \mathcal{V}\}$ with $\mathbf{A}_m$ by Eq. (7).
end for
$\mathbf{h}'_i \leftarrow \hat{\mathbf{z}}'_{i,1} \oplus \hat{\mathbf{z}}'_{i,2} \oplus \cdots \oplus \hat{\mathbf{z}}'_{i,M}, \forall i \in \mathcal{V}.$

we formulate the factor diversity for each node i as

$$\mathcal{F}_i^{\text{div}} = \det\left(\hat{\mathbf{L}}_i^T \hat{\mathbf{L}}_i\right) \tag{5}$$

where $\hat{\mathbf{L}}_i = [(\mathbf{L}_{i,1}/\|\mathbf{L}_{i,1}\|_2), (\mathbf{L}_{i,2}/\|\mathbf{L}_{i,2}\|_2), \ldots, (\mathbf{L}_{i,M}/\|\mathbf{L}_{i,M}\|_2)] \in \mathbb{R}^{M\times M}$, and $\mathbf{L}_{i,m} = [\mathcal{N}(\hat{\mathbf{z}}_{i,m}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \mathcal{N}(\hat{\mathbf{z}}_{i,m}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2), \ldots, \mathcal{N}(\hat{\mathbf{z}}_{i,m}; \boldsymbol{\mu}_M, \boldsymbol{\Sigma}_M)]^T$ is a vector in M dimensions, which contains the conditional likelihoods of a disentangled latent unit $\hat{\mathbf{z}}_{i,m}$ given M different factors. By the property of determinant [57], $\mathcal{F}_i^{\text{div}}$ is equal to the square of the volume spanned by the set $\{(\mathbf{L}_{i,1}/\|\mathbf{L}_{i,1}\|_2), (\mathbf{L}_{i,2}/\|\mathbf{L}_{i,2}\|_2), \ldots, (\mathbf{L}_{i,M}/\|\mathbf{L}_{i,M}\|_2)\}$, which offers elegantly an intuitive geometric interpretation, as shown in Fig. 3. To promote factor diversity, we introduce the diversity promoting regularizer as

$$\mathcal{L}_{\text{div}} = -\frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}} \log\left(\mathcal{F}_i^{\text{div}}\right) \tag{6}$$

with the following proposition.

Proposition 1: Minimizing $\mathcal{L}_{\text{space}}$ and $\mathcal{L}_{\text{div}}$ simultaneously as $\mathcal{L} = \lambda_{\text{space}}\mathcal{L}_{\text{space}} + \lambda_{\text{div}}\mathcal{L}_{\text{div}}$ encourages the separatability between different factor densities, where λ_{space} and λ_{div} are positive regularization constants.

Proof: Penalizing $\mathcal{L}_{\text{space}}$ in (4) models the density of each factor with a latent continuous space, where most latent units stay close to their centers. In other words, they will have the largest conditional likelihood according to their factors, i.e., $\mathbf{L}_{i,m} = \max\{\mathbf{L}_{i,1}, \mathbf{L}_{i,2}, \ldots, \mathbf{L}_{i,M}\}$ for $\hat{\mathbf{z}}_{i,m}$. On the other hand, as $(\mathbf{L}_{i,m}/\|\mathbf{L}_{i,m}\|_2)$ is normalized, the maximum value of $\mathcal{F}_i^{\text{div}}$ is 1 and can only be attained when $\mathbf{L}_{i,1}, \mathbf{L}_{i,2}, \ldots, \mathbf{L}_{i,M}$ are orthogonal to each other. Therefore, minimizing $\mathcal{L}_{\text{div}}$ in (6) further emphasizes the maximal value in vector $\mathbf{L}_{i,m}$ and promotes its discretization, thereby disjointing the possible overlaps between different factor densities. As a consequence, the disentangled latent units are encouraged to be separated spatially with respect to different factors. □

This process can essentially prune the redundancy, enhance the disentangled informativeness, and finally promote the *interfactor diversity*.

C. Strengthening Intrafactor Consistency

Although node relations can naturally be available in a graph, we believe that they are imperfect for disentangled graph learning due to data corruption or missing information. Take a huge and sparse graph as an example. It is difficult for most nodes to absorb sufficient information from their small neighborhood, especially in the case of the average neighborhood size being much less than the number of latent factors to be disentangled. On the other hand, the original graph is essentially constructed from the raw feature space of nodes and may not contain the desired topology after projecting node features in different channels for disentangling. To alleviate this issue, we propose to disclose the hidden relations between nodes from a latent space and utilize them to encode more information, which proves beneficial.

The modeled latent space as described in Section III-A embeds the disentangled latent units of all the nodes, specific to a different factor, into a different subspace, from which a new graph can naturally be constructed by connecting nearby neighbors. These graphs are expected to reflect the overall structured information from different angles and disclose the hidden node relations w.r.t. different factors. Then, the disentangled latent units are allowed to propagate on their latent graphs followed by a neighborhood aggregation. As such, the factor-specific information can be encoded globally and selectively for nodes, further strengthening the *intrafactor consistency*. Actually, there are many ways to construct latent graphs, and we list three popular ones here.

1) *k-Nearest-Neighbors [58] (kNN):* Two samples i and j are connected in a kNN graph if either of them belongs to k-nearest neighbors of the other. Formally, the adjacency matrix is given as

$$\mathbf{A}^{\text{kNN}}_{[i,j]} = \begin{cases} 1, & \mathcal{P}(i,j) \le \mathcal{P}(i,i_k) \text{ or } \mathcal{P}(j,j_k) \\ 0, & \text{otherwise} \end{cases}$$

where $\mathcal{P}(\cdot)$ denotes pairwise distance, and i_k and j_k are the kth nearest neighbors of samples i and j, respectively.

2) *Continuous k-Nearest-Neighbors [59] (CkNN):* It provides a less discrete version of kNN in case of modeling data samples, which are not uniformly distributed

$$\mathbf{A}^{\text{CkNN}}_{[i,j]} = \begin{cases} 1, & \mathcal{P}(i,j) < \delta\sqrt{\mathcal{P}(i,i_k)\mathcal{P}(j,j_k)} \\ 0, & \text{otherwise} \end{cases}$$

where δ is a scalar parameter controlling the density of the generated graphs.

3) *ϵ-Ball [60]:* Two samples i and j are connected in a ϵ-Ball graph if their distance is smaller than some scalar value ϵ

$$\mathbf{A}^{\epsilon\text{-Ball}}_{[i,j]} = \begin{cases} 1, & \mathcal{P}(i,j) < \epsilon \\ 0, & \text{otherwise.} \end{cases}$$

In this article, we uniformly apply kNN [58] algorithm on $\{\mathbf{z}_{i,m}|i \in \mathcal{V}\}$ with $\mathcal{P}(\cdot)$ being the Euclidean distance and attain a latent graph for each factor m with adjacency matrix $\mathbf{A}_m$. On the other hand, there are also multiple choices on message passing frameworks from the basic to state-of-the-arts, e.g., GCN [6], graph attention network (GAT) [11], graph isomorphism network (GIN) [61], FAGCN [62], and so on, for encoding information on these latent graphs. In our LGD-GCN, we found that simply applying the basic GCN-aggregator gives us satisfactory results and the privilege of low computational cost. Therefore, the disentangled latent units are updated as follows:

$$\hat{\mathbf{Z}}'_m \leftarrow \hat{\mathbf{D}}_m^{-\frac{1}{2}} \hat{\mathbf{A}}_m \hat{\mathbf{D}}_m^{-\frac{1}{2}} \hat{\mathbf{Z}}_m \tag{7}$$

where $\hat{\mathbf{A}}_m = \mathbf{A}_m + \mathbf{I}$, $\hat{\mathbf{D}}_m = \mathbf{D}_m + \mathbf{I}$, $\mathbf{D}_m$ is the degree matrix, and $\hat{\mathbf{Z}}_m$ represents the feature matrix with each row being $\hat{\mathbf{z}}_{i,m}^T$ for each node i in V. Particularly, we call this proposed module latent aggregation denoted $\mathbb{LG}$.

Algorithm 2 LGD-GCN's Optimization Procedure

Params: $\{(\boldsymbol{\mu}_m^{(l)} \in \mathbb{R}^{\frac{d}{M}}, \boldsymbol{\Sigma}_m^{(l)} \in \mathbb{R}^{\frac{d}{M} \times \frac{d}{M}})|\forall m = 1, 2, \ldots, M, \forall l = 1, 2, \ldots, L\}$ and $\Theta = \{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(L)}\}$

Initialize $\{(\boldsymbol{\mu}_m^{(l)}, \boldsymbol{\Sigma}_m^{(l)})|\forall m = 1, 2, \ldots, M, \forall l = 1, 2, \ldots, L\}$ and Θ with random values.
Initialize $\mathbf{h}_i^{(0)}$ with $\mathbf{h}_i$.
for number of training epochs **do**
  // Forward propagation
  for $l = 1, 2, \ldots, L$ **do**
    $\{\mathbf{h}_i^{(l)}|\forall i \in \mathcal{V}\} \leftarrow \mathbf{F}_{\theta^{(l)}}(\{\mathbf{h}_i^{(l-1)}|\forall i \in \mathcal{V}\})$
    Calculate $\mathcal{L}_{space}^{(l)}$ and $\mathcal{L}_{div}^{(l)}$ by Eq. (4) and Eq. (6).
  end for
  Calculate $\mathcal{L}_{total}$ with $(\lambda_{space}, \lambda_{div})$ by Eq. (8).
  // Back propagation
  for $l = 1, 2, \ldots, L$ **do**
    Update $\theta^{(l)}$ with $-\nabla_{\theta^{(l)}} \mathcal{L}_{total}$.
    Update $\boldsymbol{\mu}_m^{(l)}, \boldsymbol{\Sigma}_m^{(l)}, \forall m = 1, 2, \ldots, M$:
    $\{\mathbf{h}_i^{(l)}|\forall i \in \mathcal{V}\} \leftarrow \mathbf{F}_{\theta^{(l)}}(\{\mathbf{h}_i^{(l-1)}|\forall i \in \mathcal{V}\})$
    $\mathbf{z}_{i,1}^{(l)} \oplus \mathbf{z}_{i,2}^{(l)} \oplus \cdots \oplus \mathbf{z}_{i,M}^{(l)} \leftarrow \mathbf{h}_i^{(l)}, \forall i \in \mathcal{V}$
    for $m = 1, 2, \ldots, M$ **do**
      $\boldsymbol{\mu}_m^{new} \leftarrow \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{z}_{i,m}^{(l)}$
      $\boldsymbol{\Sigma}_m^{new} \leftarrow \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} (\mathbf{z}_{i,m}^{(l)} - \boldsymbol{\mu}_m^{new})(\mathbf{z}_{i,m}^{(l)} - \boldsymbol{\mu}_m^{new})^T$
      $\boldsymbol{\mu}_m^{(l)} \leftarrow (1 - U_r)\boldsymbol{\mu}_m^{(l)} + U_r \boldsymbol{\mu}_m^{new}$
      $\boldsymbol{\Sigma}_m^{(l)} \leftarrow (1 - U_r)\boldsymbol{\Sigma}_m^{(l)} + U_r \boldsymbol{\Sigma}_m^{new}$
    end for
  end for
end for

D. Network Architecture

We detail the general network architecture of the proposed LGD-GCN in this section. The pseudocode of an LGD-GCN's layer is presented in Algorithm 1, which can be stacked to exploit graph data sufficiently. In this work, by appending one single layer of our model after DisenGCN, we can even observe significant performance gain as later discussed in Section IV. Specifically, we adopt the ReLU activation function in (1) and apply dropout [63] at the end of each LGD-GCN's layer, and is only enabled in training. We can then have the output of layer l as $\{\mathbf{h}_i^{(l)}|\forall i \in \mathcal{V}\} = \text{Dropout}(\mathbf{F}_{\theta^{(l)}}(\{\mathbf{h}_i^{(l-1)}|\forall i \in \mathcal{V}\}))$, where $1 \leq l \leq L$, $\mathbf{h}_i^{(0)}$ is initialized with the raw features $\mathbf{h}_i$, L denotes the layer number, and $\mathbf{F}_{\theta^{(l)}}$ refers to LGD-GCN's lth layer. This work focuses on node classification tasks with the output representations $\{\mathbf{h}_i^{(L)} \in \mathbb{R}^d|\forall i \in \mathcal{V}\}$. We denote $\mathbf{Y}_i \in \mathbb{R}^C$ as the class prediction for node i, which can then be calculated as $\sigma(\mathbf{W}^T \mathbf{h}_i^{(L)} + \mathbf{b})$ with σ being softmax and sigmoid, respectively, for single-label (multiclass) and multilabel node classification, where $\mathbf{W} \in \mathbb{R}^{d \times C}$, $\mathbf{b} \in \mathbb{R}^C$, and C denotes the number of class. Suppose we have training set $\mathcal{V}_{\text{trn}}$ and the ground truth label set $\{\mathbb{Y}_i \in \mathbb{R}^C|\forall i \in \mathcal{V}_{\text{trn}}\}$ in one hot encoding. Then, the loss for classification task $\mathcal{L}_t$ can be expressed as $-(1/|\mathcal{V}_{\text{trn}}|) \sum_i^{\mathcal{V}_{\text{trn}}} \mathbb{Y}_i^T \log(\mathbf{Y}_i)$ and $-(1/|\mathcal{V}_{\text{trn}}|) \sum_i^{\mathcal{V}_{\text{trn}}} [\mathbb{Y}_i^T \log((\mathbf{Y}_i)) + (1 - \mathbb{Y}_i)^T \log(1 - \mathbf{Y}_i)]$ for single-label (multiclass) and multilabel node classification, separately. Combing the derived regularization terms in (4) and (6), we have the final optimization objective

$$\mathcal{L}_{\text{total}} = \mathcal{L}_t + \sum_{l=1}^{L} \lambda^{(l)} \left(\lambda_{\text{space}} \mathcal{L}_{\text{space}}^{(l)} + \lambda_{\text{div}} \mathcal{L}_{\text{div}}^{(l)} \right) \tag{8}$$

where $\mathcal{L}_{\text{space}}^{(l)}$ and $\mathcal{L}_{\text{div}}^{(l)}$ are the regularization terms calculated in the lth layer, λ_{space} and λ_{div} are the corresponding regularization coefficients, and $\lambda^{(l)}$ is taken as 10^{l-L} to grow the impact of $\mathcal{L}_{\text{space}}^{(l)}$ and $\mathcal{L}_{\text{div}}^{(l)}$ as the layer goes deeper within a proper range.

TABLE II
DATASET STATISTICS

Datasets	Blogcatalog	Flickr	Cora	Citeseer	Pubmed
# Nodes	5,196	7,575	2,708	3,327	19,717
# Edegs	171,743	239,738	5,429	4,732	44,338
# Features	8,189	12,047	1,433	3,703	500
# Classes	6	9	7	6	3
# Train	519	757	140	120	60
# Validation	1,039	1,515	500	500	500
# Test	3,638	5,303	1,000	1,000	1,000

E. Computational Analysis

In this section, we provide a brief computational analysis of the proposed LGD-GCN. Compared with DisenGCN in training, our LGD-GCN additionally needs to compute the means and covariance matrixes of Gaussian mixtures in (4). In our work, instead of learning them with stochastic gradient, we employ iterative updating with newly computed values from latent features as detailed in Algorithm 2. This optimization technique is not unique to our work but has been widely adopted in multiple research fields [64]–[66]. It not only enables a lighter computational cost in a low-dimensional latent space but also provides a stable convergent property as empirically verified in our experiments. On the other hand, we have theoretically analyzed the time complexity of our model as $\mathcal{O}(fd|\mathcal{V}| + 2|\mathcal{E}|d + (|\mathcal{V}| + (d - 1)k)|\mathcal{V}|)$, where the overhead part compared with DisenGCN is $\mathcal{O}((|\mathcal{V}| + (d - 1)k)|\mathcal{V}|)$ brought by the inference of latent graphs. It suggests that the additional computational cost is mostly

influenced by data size, of which the empirical study is provided in Sections IV-A–IV-E. We argue that our proposed LGD-GCN is still reasonably efficient in practice, especially when we consider the significant performance gains as verified in our experiments.

IV. Experiments

In this section, we show the effectiveness of our LGD-GCN with experiments on five real-world and one synthetic datasets in node classification and factor disentanglement. We also study the convergence behavior and computational complexity of our model in comparison with DisenGCN. Finally, parameter sensitivity and module ablation study are provided.

A. Experimental Setup

1) Real-World Datasets: Blogcatalog [67] is a community of online blogging where users are connected by following each other, labeled by predefined categories of interests, and given features generated based on their personal descriptions. Flickr [67] is a multimedia sharing platform, where the users follow each other online with interest tags and joined groups, respectively, being their features and labels. Cora, Citeseer, and Pubmed [68] are three typically sparse citation networks whose average neighborhood sizes are 3.9, 2.8, and 4.5, respectively. Their nodes are documents connected by undirected citations and assigned with one topic for each as well as features of bags-of-words. Data statistics are listed in Table II.

2) Synthetic Dataset: To investigate the behavior of LGD-GCN on graphs with an arbitrary number of latent factors, we also construct synthetic graphs. In detail, we first generate m Erdős-Rényi random graphs with 1000 nodes and 16 classes, where nodes connect each other with probability p if they are in the same class, with probability q otherwise. Then, we merge these generated graphs, by summing the adjacency matrix and turning the element-value bigger than zero to one, to obtain the final synthetic graphs with m latent factors. There are $16 \times m$ classes, and each node is assigned with m labels according to the original ones in the m random graphs. The rows of the adjacency matrix are taken as the node representations. Following [17], we set q to $3e^{-5}$ and tune p value such that the average neighborhood size is around 40.

3) Baselines: We compare our model with a number of mainstream GCN methods, including the state-of-the-arts, as the baselines: 1) MoNet [69] makes the first attempt to generalize convolutional neural networks to non-Euclidean graph data; 2) GCN [6] approximates the graph Laplacian with Chebyshev expansion; 3) GraphSAGE [10] is an inductive framework for large graph learning where we only consider one of its variant with GCN aggregator; 4) GAT [11] combines the attention mechanism with graph neural networks to aggregate information with important neighbors; 5) simple graph convolution (SGC) [12] simplifies GCN by removing nonlinearities; 6) JK-Net [13] leverage multihop neighborhood of nodes to capture structure-aware information; 7) DisenGCN [17] partitions node neighborhood to learn disentangled node representations; 8) IPGDN [18] further extends DisenGCN [17] with promoted independence between different factors; and 9) FactorGCN [47] employs a graph factorization to disentangle different graph aspects.

TABLE III
Model Hyperparameters

Datasets	Blogcatalog	Flickr	Cora	Citeseer	Pubmed
d	64	64	64	64	64
M	16	8	4	4	4
L	1	1	4	4	4
dropout	0.4	0.5	0.05	0.05	0.05
weight decay	0.01	0.04	0.08	0.26	1e-3
learning rate	7e-3	7e-3	0.05	0.03	0.12
U_r	0.60	0.88	0.35	0.54	0.69
λ_{space}	1	0.2	0.88	0.53	0.95
λ_{div}	0.019	0.01	0.033	0.02	4e-4
k	10	9	4	5	13

4) Implementation Details: For all the baselines and our model, we set $d = 64$ as the hidden dimension for a fair comparison, and tune the hyperparameters on the validation split of each dataset using Optuna [70] for efficiency. Following DisenGCN, we set $T = 7$ as the number of routing iterations. For semisupervised node classification on real-world datasets, we apply dropout $\sim \{0, 0.05, \ldots, 1\}$ with step 0.05, learning rate $\sim$ [1e−3, 5e−1], weight decay $\sim$ [1e−4, 5e−1], update rate $\sim [0.1, 0.9]$ for $\boldsymbol{\mu}_m$ and $\boldsymbol{\Sigma}_m$, the number of layers $\sim \{1, 2, \ldots, 10\}$, the number of channel $M \sim \{2, 4, \ldots, 16\}$ with step 2, and $m\lfloor(d/m)\rfloor$ as the hidden dimension in our model when m is not divisible by d. For multilabel classification on the synthetic dataset, with a slight difference, we apply learning rate $\sim$ [5e−4, 5e−3], and weight decay $\sim$ [1e−3, 1e−2]. With the best hyperparameters, we train models within 1000 epochs using the early-stopping strategy with patience of 100 epochs and report the average performance in ten runs on the test split. For reproducibility, we specify our used hyperparameters in Table III, and our implementation can be found at https://github.com/jingweio/LGD-GCN.

B. Quantitative Evaluation

In this section, we evaluate our model quantitatively in tasks of semisupervised node classification and multilabel node classification.

1) Semisupervised Node Classification: We consider two settings of data split, to avoid the experimental bias as argued in [72] and [73]. One is a standard split. For Blogcatalog and Flickr, we adopt the split from [46] as their standard splits. For Cora, Citeseer, and Pubmed, we follow the experimental protocol established by [6] and [11]. Another is multiple random splits for cross validation. For each dataset, we uniformly sample the same number of instances as the standard split and repeat it ten times. Then, the hyperparameters are only searched over the standard split, and the average performance in 100 runs is reported with ten random splits and ten different model initializations. The classification accuracies are summarized in Table IV.

As observed, for social networks, the disentangled approaches including DisenGCN, IPGDN, and ours outperform the holistic approaches. This is partly because the users tend to have multiple different relationships (family, friend,

TABLE IV

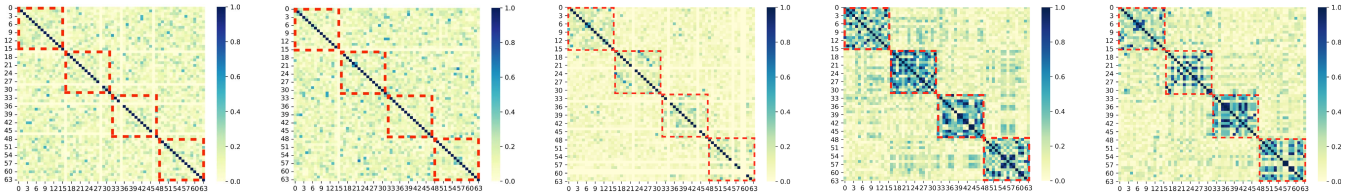
SEMISUPERVISED CLASSIFICATION ACCURACIES (%) ON THE STANDARD SPLIT (LEFT) AND MULTIPLE RANDOM SPLITS (RIGHT)

Methods	Blogcatalog		Flickr		Cora		Citeseer		Pubmed	
MoNet [69]	74.7±0.4	74.6±0.5	61.7±0.7	61.6±1.0	79.6±1.5	80.5±1.6	70.2±1.3	67.7±1.7	78.0±0.5	75.7±2.0
GCN [6]	73.8±0.3	72.9±0.4	56.6±0.4	57.6±0.3	81.8±1.0	82.3±1.6	71.8±1.3	70.7±1.3	78.7±0.5	78.5±1.6
GraphSAGE [10]	73.7±0.3	73.0±0.4	56.3±0.4	57.0±0.4	81.9±0.9	81.9±1.6	71.3±1.3	69.2±1.4	79.0±0.6	78.4±1.6
GAT [11]	56.7±5.0	57.5±3.2	45.1±1.0	45.1±1.4	81.9±0.8	81.7±1.4	73.1±0.8	70.9±1.3	78.8±0.7	78.4±1.9
SGC [12]	74.5±0.3	73.7±0.4	61.4±0.2	60.6±0.3	82.4±0.5	82.3±1.7	72.4±0.5	66.0±1.3	79.4±0.2	77.2±2.6
JK-Net [13]	76.5±0.3	75.8±0.5	64.6±0.4	64.1±0.4	82.0±0.9	82.5±1.6	73.0±0.9	70.1±1.2	79.1±0.4	78.2±1.8
DisenGCN [17]	86.5±1.3	86.4±1.2	75.8±0.6	76.7±0.6	81.9±0.9	81.8±1.4	72.5±0.8	70.0±1.3	79.7±0.6	79.0±1.9
IPGDN [18]	86.9±0.9	86.1±1.1	75.9±0.5	76.8±0.6	83.0±0.5	82.1±1.7	72.7±1.4	69.9±1.4	80.0±0.5	78.8±2.2
FactorGCN [47]	78.4±1.3	77.6±2.1	47.0±1.7	45.4±2.0	72.9±2.2	69.7±3.1	59.6±1.8	54.7±2.8	74.4±0.8	70.8±3.0
LGD-GCN (ours)	**93.7±0.4**	**93.9±0.4**	**85.5±0.6**	**84.0±0.8**	**85.2±0.6**	**83.8±1.3**	**74.4±0.5**	**72.3±1.5**	**81.5±0.6**	**80.6±2.2**

TABLE V

MICRO-F1 (LEFT) AND MACRO-F1 (RIGHT) SCORES (%) ON SYNTHETIC GRAPHS WITH DIFFERENT NUMBER OF LATENT FACTORS

Methods	4		6		8		10		12	
MLP	79.3±0.5	77.9±0.7	55.5±0.4	54.8±0.6	37.0±0.8	36.0±0.8	25.9±0.6	24.5±0.7	21.2±0.8	20.1±0.9
GCN [6]	74.5±0.8	78.3±0.9	56.3±0.7	55.6±0.9	38.2±0.9	37.2±1.0	28.0±0.7	26.9±0.5	23.1±0.8	22.2±0.9
DisenGCN [17]	84.1±1.0	82.9±1.1	60.4±0.9	59.9±1.0	41.4±1.3	40.2±1.2	29.4±0.7	28.1±0.7	24.2±0.8	23.4±0.7
LGD-GCN (ours)	**87.2±0.5**	**86.1±0.5**	**65.0±0.5**	**64.2±0.6**	**43.6±0.7**	**42.5±0.6**	**30.2±0.5**	**28.8±0.5**	**26.1±0.5**	**25.1±0.5**



(a) (b) (c) (d) (e)

Fig. 4. Feature correlation analysis. The latent features are obtained on the test split of the graph, synthesized with four latent factors, by the trained DisenGCN and our LGD-GCN. LGD-GCN* denotes LGD-GCN w/o the module $\mathbb{LG}$. (a) DisenGCN (first Layer). (b) DisenGCN (second Layer). (c) LGD-GCN (first Layer). (d) LGD-GCN (second Layer). (e) LGD-GCN* (second Layer).

and/or college) between their neighbors, and learning representations that recognize and disentangle the underlying factors could better describe the users from different angles. Although FactorGCN also considers different types of node relations, it fails on some networks and even performs worse than the holistic approaches. One main reason is that FactorGCN focuses on capturing different graph aspects from a global view while ignoring the local details important for node-level classification. On the other hand, our model achieves significant performance gains upon the disentangled state-of-the-arts averagely by 7.5% and 7.2% on Blogcatalog and Flickr, respectively. This demonstrates the benefits brought by further capturing rich global information. Importantly, real-world social networks may be updated quickly, i.e., the users could frequently follow, or meet new friends. Therefore, the static network in a certain state cannot reflect the "true" relations between users. In this circumstance, the proposed LGD-GCN is able to connect the far-reached but potentially related users by globally inferring the underlying graph structures, which may explain why significant performance improvement can be attained. In particular, for citation networks which are typically sparse, our model is able to boost the performance by a margin of 1.9% on average, showing its effectiveness in absorbing extra information from a global range.

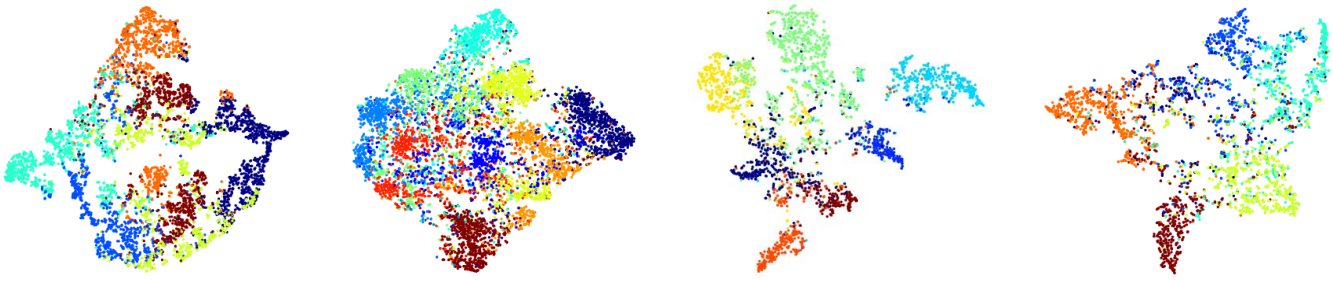
2) Multilabel Node Classification: To validate the disentangling ability of the proposed LGD-GCN quantitatively, we apply MLP, i.e., a multilayer perception, GCN, DisenGCN, and our model to train synthetic graphs with a various number of latent factors for multilabel node classification. Specifically, we randomly split each dataset into train/validation/test as 0.6/0.2/0.2, measured model performance in both micro-F1 and macro-F1 scores, and report them in Table V. From the results, our model consistently outperforms others while varying the number of latent factors. Especially, LGD-GCN significantly outperforms DisenGCN by (Micro-F1) 4.6% and (Macro-F1) 4.3% on the synthetic graph with six latent factors.

C. Qualitative Evaluation

To gain more understanding of our proposed method, we conduct various qualitative experiments to take closer examinations in parallel with DisenGCN. These evaluations focus on the disentanglement performance and the learned embeddings' informativeness.

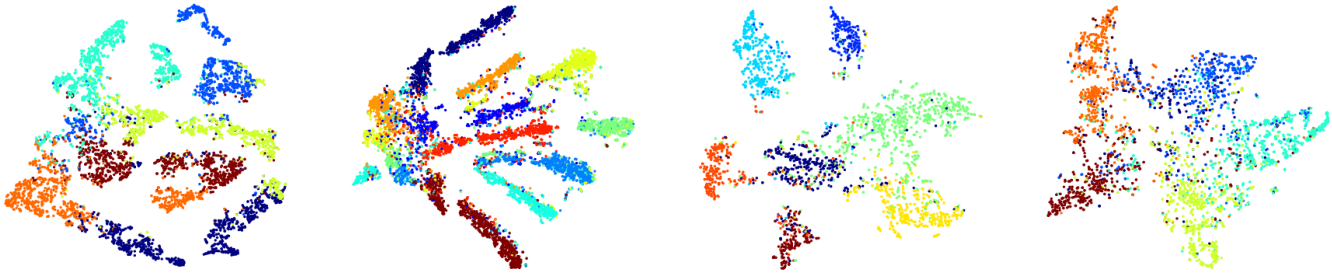
1) Visualization of Disentangled Representations: We plot in Fig. 1(b) a 2-D visualization of the learned representations w.r.t. four latent factors on the synthetic graph. Compared with that of DisenGCN in Fig. 1(a), our model displays a highly disentangled pattern, evidenced by the intrafactor compactness and interfactor separability. It also indicates that the common type of factor-specific information is captured, and globally shared by all nodes.

2) Correlation of Disentangled Features: The correlation analysis of the latent features learned by DisenGCN and our model is presented in Fig. 4. As observed, our model showcases a more blockwise correlation pattern, which becomes



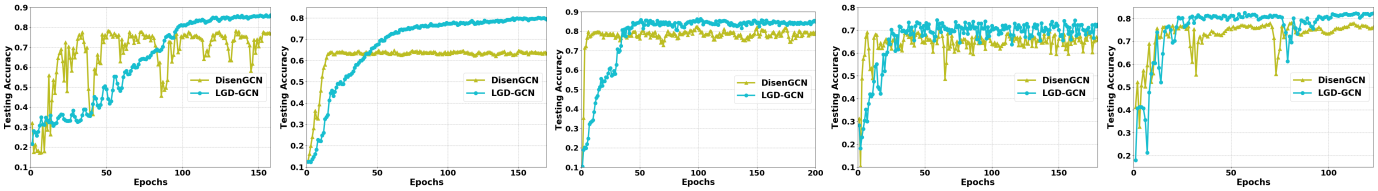
(a) (b) (c) (d)

Fig. 5. Visualization of node embeddings learned by DisenGCN. (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer.



(a) (b) (c) (d)

Fig. 6. Visualization of node embeddings learned by LGD-GCN (ours). (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer.



(a) (b) (c) (d) (e)

Fig. 7. Convergence behavior of DisenGCN and the proposed LGD-GCN. (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer. (e) Pubmed.

denser in the second layer. We also analyze the feature correlation of our model while ablating the module $\mathbb{LG}$, denoted as LGD-GCN* in Fig. 4(e). Though the blockwise pattern in Fig. 4(e) can still be observed, it is obviously weaker than that of LGD-GCN in Fig. 4(d). This verifies the significance of $\mathbb{LG}$. The captured global information, specific to each latent factor, strengthens the correlation between the intrafactor features and enhances the interpretability and disentangling power.

3) Visualization of Node Embeddings: Figs. 5 and 6 provide an intuitive comparison between the learned node embeddings of DisenGCN and our model. It can be observed that the proposed LGD-GCN generally learns better node embeddings and exhibits a high intraclass similarity and interclass difference. By absorbing rich global information specific to each latent factor, our model learns more informative node aspects, and thus, leads to superior discriminative power.

D. Convergence Behavior and Complexity Analysis

In Fig. 7, we plot the evolution of the testing accuracy for DisenGCN and the proposed LGD-GCN. In general, DisenGCN fluctuates significantly by epochs and is prone to get stuck into a local optimum, while our model appears more stable and able to converge to a higher peak. For complexity analysis, we first report the average training time (ms) per epoch in Table VI. On average, LGD-GCN is around

TABLE VI
AVERAGE TRAINING TIME (MS) PER EPOCH

Datasets	Blogcatalog	Flickr	Cora	Citeseer	Pubmed
DisenGCN	15.9	22.0	28.4	35.4	116.8
LGD-GCN (Ours)	26.4	33.8	31.6	39.5	163.9

11.4% and 53.3% slower than DisenGCN on small datasets (Cora and Citeseer) and large datasets (Pubmed, Blogcatalog, and Flickr). That is mainly caused by the computation of latent graphs with the complexity of $\mathcal{O}(|\mathcal{V}|^2)$. However, with the significant performance gain, we believe that such costs may be worthwhile especially when the computational capability and stability are being steadily empowered. On the other hand, as we have the time complexity of our model as $\mathcal{O}(fd|\mathcal{V}| + 2|\mathcal{E}|d + (|\mathcal{V}| + (d-1)k)|\mathcal{V}|)$, there should be no tradeoff between the channel number M and running time. However, our experimental findings show that the inference time of LGD-GCN gets approximately 0.02 ms increased with one channel added, which is mainly owing to the additional time consumed by looping channels in our implementation.

E. Parameter and Ablation Analysis

In this section, we investigate the sensitivity of three essential hyperparameters and perform ablation analysis over the proposed different modules.

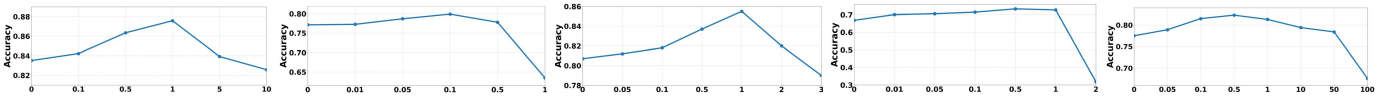


Fig. 8. Analysis of parameter λ_{space}. (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer. (e) Pubmed.

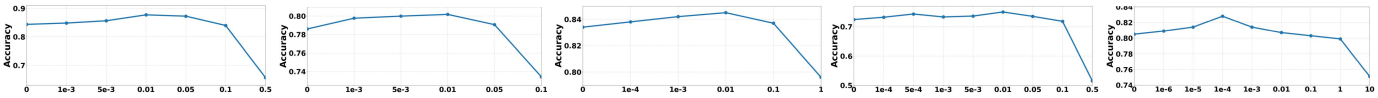


Fig. 9. Analysis of parameter λ_{div}. (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer. (e) Pubmed.

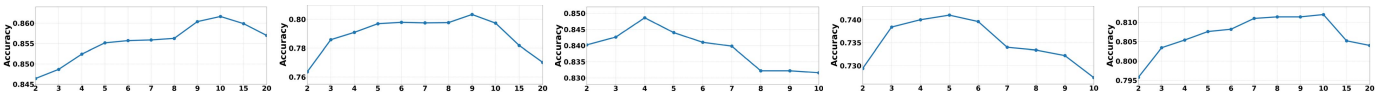


Fig. 10. Analysis of parameter k. (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer. (e) Pubmed.

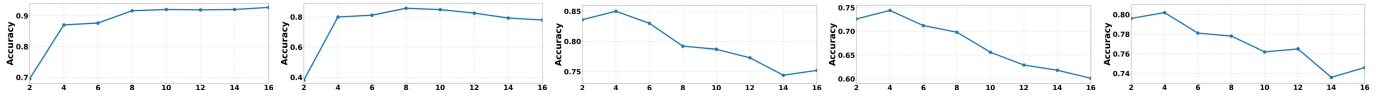


Fig. 11. Analysis of parameter M. (a) Blogcatalog. (b) Flickr. (c) Cora. (d) Citeseer. (e) Pubmed.

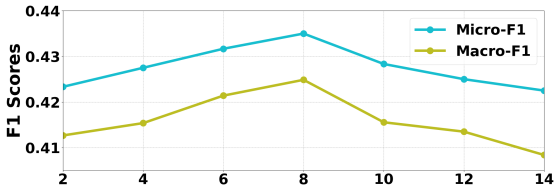


Fig. 12. Analysis of parameter M on a synthetic graph.

TABLE VII
ABLATION ANALYSIS

Components	Blogcatalog	Flickr	Cora	Citeseer	Pubmed
-	86.1±0.5	69.6±0.6	81.2±1.2	68.9±1.5	78.5±1.1
$\mathcal{L}_{space}$	91.1±0.5	70.2±1.1	83.2±0.4	72.4±0.6	78.9±0.9
$\mathcal{L}_{space}$+$\mathcal{L}_{div}$	91.5±0.3	70.5±1.1	83.4±0.4	73.0±0.9	79.2±0.8
$\mathcal{L}_{space}$+$\mathbb{LG}$	93.7±0.5	85.2±0.7	84.0±0.9	74.0±0.8	80.8±0.7
$\mathcal{L}_{space}$+$\mathcal{L}_{div}$+$\mathbb{LG}$	**93.7±0.4**	**85.5±0.6**	**85.2±0.6**	**74.4±0.5**	**81.5±0.6**

1) Analysis of Space Modeling Coefficient λ_{space}: We plot the learning performance of our model w/o $\mathcal{L}_{\text{div}}$ while varying λ_{space} in (8). For example, we adopt a range of $\{0, 0.01, 0.05, 0.1, 0.5, 1\}$ on Flickr and report the learning performance in Fig. 8(b). In general, the accuracy goes up first and then drops; a promising result can be attained by choosing λ_{space} from $[0.05, 0.5]$. Similar trends can also be observed in the other four datasets.

2) Analysis of Diversity Coefficient λ_{div}: We also examine the effect of λ_{div} by varying its value. For example, λ_{div} is changed from 0 to 0.5 on Citeseer. The results are shown in Fig. 9. Basically, λ_{div} is relatively robust within $[0, 0.1]$ for all the datasets except for Flickr whose stable range is $[0, 0.05]$. Once out of that range, the results drop to a low point, suggesting that overly emphasizing diversity could be harmful to model performance.

3) Analysis of Density Parameter k: Fig. 10 displays the impact of k. The results are relatively stable while selecting k around 4 for Cora as well as Citeseer and 10 for the rest. However, as k is larger, the accuracy performance deteriorates obviously. Such a trend may be caused by noisy edges in cases of a large k, which leads to inappropriate information sharing.

4) Analysis of Channel Number M: We test the effect of channel number M on real-world datasets in Fig. 11. As can be observed, LGD-GCN attains its highest accuracy with the channel number around 16 for Blogcatalog and 8 for Flickr. In comparison, the ideal channel number is relatively smaller on citation networks, where LGD-GCN performs the best with M being 4. We also study its influence on the synthetic graph with eight predefined factors as a typical example. From Fig. 12, our model performs the best when the number of channels is around 8, the true number of the latent factors.

5) Ablation Analysis: We validate the contributions of the proposed modules denoted by $\mathcal{L}_{\text{space}}$, $\mathcal{L}_{\text{div}}$, and $\mathbb{LG}$ in node classification. From Table VII, we can see that both modules can independently and jointly improve the accuracy.

V. CONCLUSION

We argue that most GCNs have inherited issues due to their entangled representations and/or heavy reliance on local graph information. Motivated by this problem, we propose a novel framework termed LGD-GCN to learn disentangled node representations both *locally and globally*. LGD-GCN is capable of disclosing the hidden node relations pertinent to each latent factor. Specifically, we first present a disentangled latent continuous space with Gaussian mixtures, from which various new graphs w.r.t. different factors can be learned and disentangled. These graphs reflect the latent structure information, i.e., the hidden relations between nodes, overall from different angles. We then utilize them to aggregate and capture the factor-specific information *globally*, which strengthens the

intrafactor consistency. Moreover, to avoid the mistakenly preserved confounding of the factors, we also promote the *interfactor diversity* by a novelly designed regularizer along with the latent space modeling. Extensive experiments over synthetic and five real-world datasets well demonstrate the improved classification accuracy and disentangling ability over the state-of-the-arts both quantitatively and qualitatively.

In this article, we model the disentangled latent space with the Gaussian mixture model and weigh each mixture equally for simplicity. Despite its efficiency, this may not be optimal or even valid in many real scenarios. In the future, it would be important and beneficial to investigate a more flexible and applicable way of describing the disentangled latent space. Additionally, compared with other disentangled state-of-the-arts only relying on the local graph information, our LGD-GCN shows a higher algorithm complexity because of its denser computational cost of inferring the latent graph information globally. Therefore, another interesting direction for future work is to deploy a lighter algorithm to learn the latent graphs without losing informativeness.

REFERENCES

[1] Y.-M. Zhang, K. Huang, G.-G. Geng, and C.-L. Liu, "MTC: A fast and robust graph-based transductive learning method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 9, pp. 1979–1991, Sep. 2015.

[2] M. Shi, Y. Tang, and X. Zhu, "MLNE: Multi-label network embedding," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3682–3695, Sep. 2020.

[3] U. S. Shanthamallu, J. J. Thiagarajan, H. Song, and A. Spanias, "GrAMME: Semisupervised learning using multilayered graph attention models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 3977–3988, Oct. 2020.

[4] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 1–21, Feb. 2021.

[5] J. Wu, S. Pan, X. Zhu, C. Zhang, and P. S. Yu, "Multiple structure-view learning for graph classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3236–3251, Jul. 2018.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–14.

[7] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Mar. 2020.

[8] T. Chen and R. C.-W. Wong, "Handling information loss of graph neural networks for session-based recommendation," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1172–1180.

[9] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 249–270, Jan. 2022.

[10] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017, pp. 1–11.

[11] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–12. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[12] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6861–6871.

[13] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *Proc. ICML*, 2018, pp. 5453–5462.

[14] K. Huang, H. Yang, I. King, and M. Lyu, *Modeling Data Locally and Globally*. Springer, 2008.

[15] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. Devon Hjelm, "Deep graph infomax," 2018, *arXiv:1809.10341*.

[16] H. Li, B. Wang, L. Cui, L. Bai, and E. R. Hancock, "LGL-GNN: Learning global and local information for graph neural networks," in *Structural, Syntactic, and Statistical Pattern Recognition*. Cham, Switzerland: Springer, 2021, pp. 129–138.

[17] J. Ma, P. Cui, K. Kuang, X. Wang, and W. Zhu, "Disentangled graph convolutional networks," in *Proc. ICML*, 2019, pp. 4212–4221.

[18] Y. Liu, X. Wang, S. Wu, and Z. Xiao, "Independence promoted graph disentangled networks," in *Proc. AAAI*, 2020, pp. 4916–4923.

[19] L. Hu *et al.*, "Graph neural news recommendation with unsupervised preference disentanglement," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 4255–4264.

[20] X. Wang, H. Jin, A. Zhang, X. He, T. Xu, and T.-S. Chua, "Disentangled graph collaborative filtering," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1001–1010.

[21] X. Wang, R. Wang, C. Shi, G. Song, and Q. Li, "Multi-component graph convolutional collaborative filtering," in *Proc. AAAI*, 2020, pp. 6267–6274.

[22] Z. Ghahramani and G. E. Hinton, "The EM algorithm for mixtures of factor analyzers," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. CRG-TR-96-1, 1996, vol. 60.

[23] L. Hajderanj, I. Weheliye, and D. Chen, "A new supervised t-SNE with dissimilarity measure for effective data visualization and classification," in *Proc. 8th Int. Conf. Softw. Inf. Eng.*, Apr. 2019, pp. 232–236.

[24] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in *Proc. AAAI*, 2018, pp. 1–8.

[25] C. Zhuang and Q. Ma, "Dual graph convolutional networks for graph-based semi-supervised classification," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 499–508.

[26] J. Chen, T. Ma, and C. Xiao, "FastGCN: Fast learning with graph convolutional networks via importance sampling," 2018, *arXiv:1801.10247*.

[27] W. B. Huang, T. Zhang, Y. Rong, and J. Huang, "Adaptive sampling towards fast graph representation learning," in *Proc. NeurIPS*, 2018, pp. 1–10.

[28] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "Graph-SAINT: Graph sampling based inductive learning method," 2019, *arXiv:1907.04931*.

[29] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.

[30] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[31] X. Chen *et al.*, "Variational lossy autoencoder," 2016, *arXiv:1611.02731*.

[32] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum, "Deep convolutional inverse graphics network," in *Proc. NIPS*, 2015, pp. 1–9.

[33] S. Narayanaswamy *et al.*, "Learning disentangled representations with semi-supervised deep generative models," in *Proc. NIPS*, 2017, pp. 1–11.

[34] R. Lopez, J. Regier, N. Yosef, and M. I. Jordan, "Information constraints on auto-encoding variational Bayes," in *Proc. NeurIPS*, 2018, pp. 1–12.

[35] I. Higgins *et al.*, "Beta-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. ICLR*, 2017, vol. 2, no. 5, p. 6.

[36] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. NIPS*, 2016, pp. 1–9.

[37] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," 2016, *arXiv:1612.00410*.

[38] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[39] D. Luo *et al.*, "Learning to drop: Robust graph neural network via topological denoising," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 779–787.

[40] Y. Zhu *et al.*, "A survey on graph structure learning: Progress and opportunities," 2021, *arXiv:2103.03036*.

[41] D. Yu, R. Zhang, Z. Jiang, Y. Wu, and Y. Yang, "Graph-revised convolutional network," in *Proc. ECML/PKDD*, 2020, pp. 378–393.

[42] B. Jiang, Z. Zhang, D. Lin, J. Tang, and B. Luo, "Semi-supervised learning with graph learning-convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11313–11320.

[43] Y. Chen, L. Wu, and M. Zaki, "Iterative deep graph learning for graph neural networks: Better and robust node embeddings," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 19314–19326.

[44] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Learning discrete structures for graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1972–1982.

[45] C. Zheng *et al.*, "Robust graph representation learning via neural sparsification," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 11458–11468.
[46] T. Zhao, Y. Liu, L. Neves, O. J. Woodford, M. Jiang, and N. Shah, "Data augmentation for graph neural networks," in *Proc. AAAI*, 2021, pp. 11015–11023.
[47] Y. Yang, Z. Feng, M. Song, and X. Wang, "Factorizable graph convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 20286–20296.
[48] M. Vlachos, C. Domeniconi, D. Gunopulos, G. Kollios, and N. Koudas, "Non-linear dimensionality reduction techniques for classification and visualization," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2002, pp. 645–651.
[49] X. Geng, D.-C. Zhan, and Z.-H. Zhou, "Supervised nonlinear dimensionality reduction for visualization and classification," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 35, no. 6, pp. 1098–1107, Nov. 2005.
[50] S.-Q. Zhang, "Enhanced supervised locally linear embedding," *Pattern Recognit. Lett.*, vol. 30, no. 13, pp. 1208–1218, 2009.
[51] J. Cheng, H. Liu, F. Wang, H. Li, and C. Zhu, "Silhouette analysis for human action recognition based on supervised temporal t-SNE and incremental learning," *IEEE Trans. Image Process.*, vol. 24, no. 10, pp. 3203–3217, Oct. 2015.
[52] M. Yu, S. Zhang, L. Zhao, and G. Kuang, "Deep supervised t-SNE for SAR target recognition," in *Proc. 2nd Int. Conf. Frontiers Sensors Technol. (ICFST)*, Apr. 2017, pp. 265–269.
[53] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The Mahalanobis distance," *Chemometrics Intell. Lab. Syst.*, vol. 50, no. 1, pp. 1–18, 2000.
[54] P. Xie, "Diversity-promoting and large-scale machine learning for healthcare," Ph.D. dissertation, Univ. Pittsburgh Med. Center, Pittsburgh, PA, USA, 2018.
[55] B. Xie, Y. Liang, and L. Song, "Diversity leads to generalization in neural networks," 2016, *arXiv:1611.03131*.
[56] A. Kulesza and B. Taskar, "Determinantal point processes for machine learning," *Found. Trends Mach. Learn.*, vol. 5, nos. 2–3, pp. 123–286, Dec. 2012.
[57] D. Bernstein, "Matrix mathematics: Theory, facts, and formulas with application to linear systems theory [Book review; DS Berstein]," *IEEE Trans. Autom. Control*, vol. 52, no. 8, pp. 1539–1540, Aug. 2007.
[58] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.
[59] T. Berry and T. Sauer, "Consistent manifold representation for topological data analysis," *Found. Data Sci.*, vol. 1, no. 1, pp. 1–38, 2019.
[60] Z. Liu and M. Barahona, "Graph-based data clustering via multiscale community detection," *Appl. Netw. Sci.*, vol. 5, no. 1, pp. 1–20, Dec. 2020.
[61] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–17. [Online]. Available: https://openreview.net/forum?id=ryGs6iA5Km
[62] D. Bo, X. Wang, C. Shi, and H. Shen, "Beyond low-frequency information in graph convolutional networks," in *Proc. AAAI*. Palo Alto, CA, USA: AAAI Press, 2021, pp. 3950–3957.
[63] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.
[64] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 499–515.
[65] S. Zhang, K. Huang, J. Zhu, and Y. Liu, "Manifold adversarial training for supervised and semi-supervised learning," *Neural Netw.*, vol. 140, pp. 282–293, Aug. 2021.
[66] S. Zhang, Z. Qian, K. Huang, Q. Wang, R. Zhang, and X. Yi, "Towards better robust generalization with shift consistency regularization," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 12524–12534.
[67] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, Feb. 2017, pp. 731–739.
[68] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, 2008.
[69] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5425–5434.
[70] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 2623–2631.
[71] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *Proc. ICLR*, 2019, pp. 1–15.
[72] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," 2018, *arXiv:1811.05868*.

Jingwei Guo received the bachelor's degree (Hons) in applied mathematics from the University of Liverpool, Liverpool, U.K., in 2018, where he is currently pursuing the Ph.D. degree.

He was a Research Associate with Xi'an Jiaotong–Liverpool University, Suzhou, China, for one year. His research interests include developing new graph neural networks and applying the techniques in various domains.

Kaizhu Huang received the Ph.D. degree from The Chinese University of Hong Kong (CUHK), Hong Kong, in 2004.

He works on machine learning, neural information processing, and pattern recognition. He was with the Fujitsu Research Centre, CUHK, the University of Bristol, Bristol, U.K., the National Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing, China, and Xi'an Jiaotong–Liverpool University, Suzhou, from 2004 to 2022. He is currently a Professor of electrical and computer engineering with Duke Kunshan University, Suzhou, China.

Dr. Huang was a recipient of the 2011 Asia–Pacific Neural Network Society Young Researcher Award. He received the Best Paper or Book Award seven times and published extensively in the IEEE TRANSACTIONS and top conferences, including AAAI, ICDM, NeurIPS, ICML, ECML, and CVPR. He serves as an associated editor/advisory board member in a number of journals and book series. He was invited as a keynote speaker at more than 30 international conferences or workshops.

Xinping Yi received the Ph.D. degree in electronics and communications from Télécom Paris, Paris, France, in 2015.

He is currently a Lecturer (Assistant Professor) with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, U.K. He was a Research Associate with the Technische Universität Berlin, Berlin, Germany, from 2014 to 2017, a Research Assistant with EURECOM, Sophia Antipolis, France, from 2011 to 2014, and a Research Engineer with Huawei Technologies, Shenzhen, China, from 2009 to 2011. His current research interests include information theory, graph theory, and machine learning, and their applications in wireless communications and artificial intelligence.

Rui Zhang received the bachelor's degree (Hons) in telecommunication engineering from Jilin University, Changchun, China, in 2001, and the Ph.D. degree in computer science and mathematics from the University of Ulster, Belfast, U.K., in 2007.

After finishing her Ph.D. study, she was a Research Associate with the University of Bradford, Bradford, U.K., and the University of Bristol, Bristol, U.K., for five years. She joined Xi'an Jiaotong–Liverpool University, Suzhou, China, in 2012, where she currently holds the position of Associate Professor. Her research interests include machine learning, data mining, and statistical analysis.