Neural Networks 158 (2023) 188–196

Contents lists available at ScienceDirect

Neural Networks

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

Neural Networks Letter

A unified deep semi-supervised graph learning scheme based on nodes re-weighting and manifold regularization

Fadi Dornaika [a,b,c,*], Jingjun Bi [b], Chongsheng Zhang [a]

[a] *Henan University, Henan Key Lab of Big Data Analysis and Processing, Kaifeng, China*
[b] *University of the Basque Country, UPV/EHU, San Sebastian, Spain*
[c] *IKERBASQUE, Basque Foundation for Science, Bilbao, Spain*

A R T I C L E I N F O

Article history:
Received 27 July 2022
Received in revised form 1 October 2022
Accepted 10 November 2022
Available online 19 November 2022

Keywords:
Semi-supervised learning
Deep Graph Neural Networks
Graph Convolutional Networks
Graph construction
Graph regularization

A B S T R A C T

In recent years, semi-supervised learning on graphs has gained importance in many fields and applications. The goal is to use both partially labeled data (labeled examples) and a large amount of unlabeled data to build more effective predictive models. Deep Graph Neural Networks (GNNs) are very useful in both unsupervised and semi-supervised learning problems. As a special class of GNNs, Graph Convolutional Networks (GCNs) aim to obtain data representation through graph-based node smoothing and layer-wise neural network transformations. However, GCNs have some weaknesses when applied to semi-supervised graph learning: (1) it ignores the manifold structure implicitly encoded by the graph; (2) it uses a fixed neighborhood graph and focuses only on the convolution of a graph, but pays little attention to graph construction; (3) it rarely considers the problem of topological imbalance.

To overcome the above shortcomings, in this paper, we propose a novel semi-supervised learning method called Re-weight Nodes and Graph Learning Convolutional Network with Manifold Regularization (ReNode-GLCNMR). Our proposed method simultaneously integrates graph learning and graph convolution into a unified network architecture, which also enforces label smoothing through an unsupervised loss term. At the same time, it addresses the problem of imbalance in graph topology by adaptively reweighting the influence of labeled nodes based on their distances to the class boundaries. Experiments on 8 benchmark datasets show that ReNode-GLCNMR significantly outperforms the state-of-the-art semi-supervised GNN methods.[1]

1. Introduction

In recent years, graph learning has demonstrated its importance in many practical domains and applications. This is partially due to the fact that a large amount of real-world data presents natural graph relationships. For instance, social networks reflect connections between people and can be used to predict/recommend items that are of interest to specific users. Such graph relationships can also be derived from unstructured data such as text collections and images. However, data annotation is often costly and time-demanding. To remedy this issue, people can resort to explicit graphs to represent the pairwise relationships between nodes/samples from the partially labeled data, then compute the similarities between the nodes/attributes to infer the labels of the unannotated samples, which is referred as semi-supervised learning on graphs (El Traboulsi, Dornaika, & Assoum, 2015). To put it simply, semi-supervised learning on graphs aims to utilize graph theory on the partially labeled data to infer the labels of the vast amount of unlabeled samples.

With the rise and development of deep learning, Deep Graph Neural Networks (GNNs) have been proposed, which have made breakthroughs in various machine learning tasks (Peng et al., 2021; Thiede, Zhou, & Kondor, 2021; Tolstikhin et al., 2021; Wu et al., 2021). They have been adapted to tackle unsupervised and semi-supervised learning problems as well, such as GGNN (Li, Tarlow, Brockschmidt, & Zemel, 2015) and SSE (Dai, Kozareva, Dai, Smola, & Song, 2018). Graph Convolutional Networks (GCNs) are a special class of GNNs, which encode the graph structure with a neural network model and perform linear and nonlinear neural network transformation at each layer. Well-established GCN methods include Spectral CNN (Bruna, Zaremba, Szlam, & LeCun, 2013), ChebNet (Defferrard, Bresson, & Vandergheynst, 2016), CayleyNet (Levie, Monti, Bresson, & Bronstein, 2018), GCN (Kipf & Welling, 2016), AGCN (Li, Wang, Zhu, & Huang,

* Corresponding author at: University of the Basque Country, UPV/EHU, San Sebastian, Spain.
E-mail addresses: fadi.dornaika@ehu.eus (F. Dornaika), bi.jingjun@outlook.com (J. Bi), cszhang@ieee.org (C. Zhang).

[1] The code is available at https://github.com/BiJingjun/ReNode-GLCNMR

https://doi.org/10.1016/j.neunet.2022.11.017

2018), DGCN (Zhuang & Ma, 2018), Diffusion CNN (Atwood & Towsley, 2016), scCDG (Wang, Zhao, Su, & Zheng, 2021), scGAC (Zhang, Gao, Zhao, Zheng, & Liu, 2022) and DAEMKL (Zhou et al., 2021). The model in scCDG has a graph autoencoder that uses a Graph Convolution Network (GCN) to address the challenge of large datasets and excessive noise in single-cell clustering. scGAC is a graph autoencoder-based consensus-driven model for efficient use of single-cell data and exploration of inter-cell heterogeneity. DAEMKL is a deep learning method for identifying microRNA-disease associations (MDAs) using deep autoencoders with multiple kernel learning.

However, when adapting GCNs to semi-supervised graph learning, researchers may face the following problems/challenges: (1) GCN models only focus on the fitness between ground-truth and predicted labels, but ignores the manifold structure implicitly encoded by the graph, which is a useful cue in semi-supervised learning (Kejani, Dornaika, & Talebi, 2020) because it encodes both local and global structures of the data and forces regularization of the model's output according to these structures. (2) GCNs often use a fixed graph, which is not always optimal for semi-supervised learning problems. This is because graphs in real world (especially for some non-graph data) can be noisy and may have spurious connectivity relationships. Such inaccurate graph structure has a negative impact on GCN message passing at each layer. (3) The two-stage framework of the GCN model cannot fully exploit the correlation between graph construction and graph convolution learning, and may lead to sub-optimal solutions. Indeed, the focus is often on graph convolution, while little attention is paid to graph construction (Jiang, Zhang, Lin, Tang, & Luo, 2019). (4) Graph data built upon partially labeled data commonly present imbalanced graph representation issue, due to the asymmetric topological properties of the labeled nodes, i.e., the labeled nodes are not equal with respect to their structural role in the graph (topological imbalance) (Chen et al., 2021).

Over the last years, researchers have proposed a few improved models to address one of the above-mentioned problems, such as GCNMR (Kejani et al., 2020), GLCN (Jiang et al., 2019), RGLN (Tang, Gao, & Hu, 2021), and ReNode (Chen et al., 2021). However, to the best of our knowledge, none of them can solve all the challenges in a unified manner.

To this end, in this work, we propose a novel model for semi-supervised learning problems that can overcome all the above shortcomings. This method is hereafter referred to as Re-weight Nodes and Graph Learning-Convolutional Network with Manifold Regularization (ReNode-GLCNMR for short), which has the following unique characteristics:

(i) It learns an optimal graph representation for semi-supervised learning to serve GCN layers by simultaneously integrating graph learning and graph convolution into a unified network architecture. In our proposal, the constructed graph is constrained by both the input and output data, while the GLCN method (Jiang et al., 2019) constrains the graph only by the input data.

(ii) It measures the degree of imbalance of the graph topology with an influence conflict detection based metric, and adaptively reweights the influence of the labeled nodes based on their relative distances to the class boundaries.

(iii) It enforces the smoothness of the predicted labels of the entire samples using an unsupervised loss term.

We carry out extensive experiments on 8 benchmark datasets, and the results show that our proposal significantly outperforms the state-of-the-art GCNs based semi-supervised learning approaches, especially on image datasets.

The remainder of this paper is organized as follows. In Section 2, we give a brief overview of some related semi-supervised methods. In Section 3, we present the ReNode-GLCNMR model in detail. Then, Section 4 reports the experimental results of our proposed ReNode-GLCNMR methods. Finally, Section 5 concludes the work.

2. Related work

In this section we will introduce the main notations. We will then briefly introduce GCN-based semi-supervised learning (Kipf & Welling, 2016) and some improved models based on GCNs.

2.1. Notations

In this article, matrices are shown in bold uppercase letters and vectors are shown in bold lowercase letters. We also emphasize that the two terms "data sample" and "node" mean the same thing. We define $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \ldots; \mathbf{x}_n] \in \mathbf{R}^{n\times d}$ as the input matrix representing n data samples (row vectors with d features), the graph similarity matrix $\mathbf{A}$ or $\mathbf{S} \in \mathbf{R}^{n\times n}$ denotes the pairwise relationships between the data $\mathbf{X}$, $\mathbf{D}$ is a diagonal matrix whose elements are the row or column sums of $\mathbf{A}$, $\mathbf{I}$ denotes the identity matrix, and $\mathbf{L}$ is the normalized Laplacian matrix given by $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}$. In the following, $\mathbf{A}$ will denote a fixed, precomputed graph and $\mathbf{S}$ will denote an adaptive, constructed graph. $\mathbf{W}^{(k)} \in \mathbf{R}^{d_k\times d_{k+1}}$ is a learnable layer-specific weight matrix in the kth layer in GCN, and $d_0 = d$,

$\mathbf{P}$ denotes the Personalized PageRank matrix, T_i is the Totoro value of node i (Chen et al., 2021), c denotes the number of classes, $\mathbf{Y}$ is the ground truth labels matrix of the labeled samples, and $\mathbf{Z} \in \mathbf{R}^{n\times c}$ is the soft label prediction for all n data samples $\mathbf{X}$, where each row $\mathbf{Z}_i$ is the soft label prediction for the ith node.

2.2. Graph Convolutional Networks (GCNs)

Let $\mathbf{A} \in \mathbf{R}^{n\times n}$ be a fixed graph representing the pairwise relations between the data $\mathbf{X}$. GCN (Kipf & Welling, 2016) with K layers performs layer-wise propagation in hidden layers as follows,

$$\mathbf{X}^{(k+1)} = \sigma(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{X}^{(k)}\mathbf{W}^{(k)}) \tag{1}$$

Here $k = 0, 1, \ldots, K-1$. $\mathbf{D}$ is a diagonal matrix whose elements are the row or column sums of $\mathbf{A}$. $\mathbf{W}^{(k)} \in \mathbf{R}^{d_k\times d_{k+1}}$, $d_0 = d$ is a learnable layer-specific weight matrix. $\sigma(\cdot)$ denotes a non-linear activation function, here it is $ReLU(\cdot) = max(0, \cdot)$. $\mathbf{X}^{(k+1)} \in \mathbf{R}^{n\times d_{k+1}}$ denotes the output of the activations in the kth layer, and $\mathbf{X}^{(0)} = \mathbf{X}$. The last layer of the GCN (Kipf & Welling, 2016) is

$$\mathbf{Z} = softmax(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{X}^{(K-1)}\mathbf{W}^{(K-1)}) \tag{2}$$

where c is the number of classes, $\mathbf{Z} \in \mathbf{R}^{n\times c}$ is the matrix of labels representing the label prediction for $\mathbf{X}$, $\mathbf{Z}_i$ denotes the soft label prediction for the ith node, $\left\{\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(K-1)}\right\}$ is the optimal weight matrix trained by minimizing the following cross-entropy loss function:

$$\mathcal{L}_{Semi-GCN} = -\sum_{i\in L}\sum_{j=1}^{c} Y_{ij}\, ln\, Z_{ij} \tag{3}$$

where L denotes the set of labeled nodes and $\mathbf{Y}$ is the ground truth labeling matrix of the labeled samples.

2.3. Improved models based on GCNs

Graph Attention Network (GAT) The method presented in Velikovi et al. (2017) learns graphs and prediction tasks simultaneously. It does not explicitly generate graphs, but uses an attention-based similarity measure to learn the weight of the relationship between each node and its neighbors.

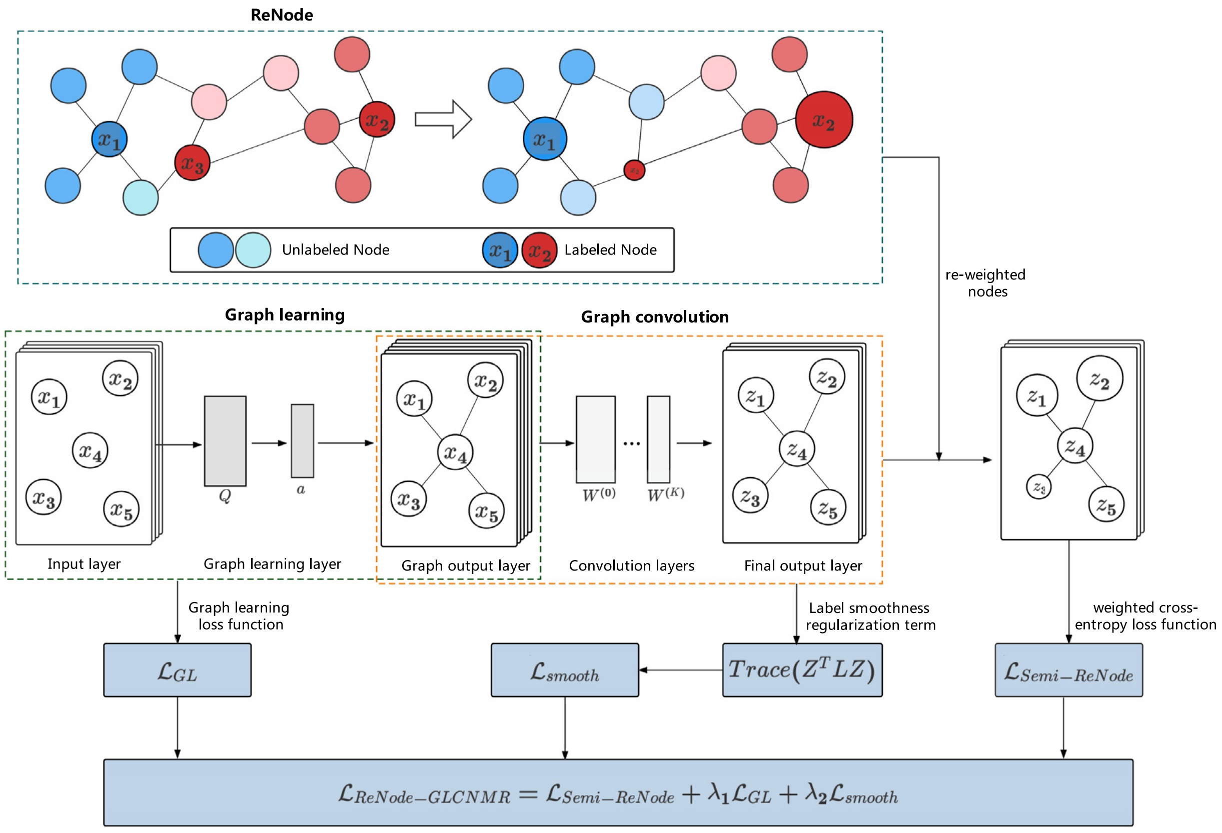

ReNode
Unlabeled Node
Labeled Node
re-weighted nodes
Graph learning
Graph convolution
Input layer
Graph learning layer
Graph output layer
Convolution layers
Final output layer
Graph learning loss function
Label smoothness regularization term
weighted cross-entropy loss function
$\mathcal{L}_{GL}$
$\mathcal{L}_{smooth}$
$Trace(Z^T LZ)$
$\mathcal{L}_{Semi-ReNode}$
$\mathcal{L}_{ReNode-GLCNMR} = \mathcal{L}_{Semi-ReNode} + \lambda_1 \mathcal{L}_{GL} + \lambda_2 \mathcal{L}_{smooth}$

Fig. 1. The framework of our proposed ReNode-GLCNMR method.

GCNs with Manifold Regularization (GCNMR) In Kejani et al. (2020), the authors proposed GCNMR. GCNMR has another unsupervised item than GCN, which enforces the smoothness of the predicted labels of the entire samples, thus deriving a more powerful semi-supervised learning. GCNMR retains the advantages of classical GCN yet can improve it without increasing the time complexity. It performs feature propagation in each layer and ensures that the predicted labels satisfy the manifold regularization.

Graph Learning-Convolutional Network (GLCN) Since GCNs use a fixed graph, which is not always optimal for semi-supervised learning problems, and a two-stage framework in the GCN model cannot fully exploit the correlation between graph construction and graph convolution learning and may lead to a weak suboptimal solution, GLCN aims to learn an optimal graph representation by simultaneously integrating graph learning and graph convolution into a unified network architecture. GLCN includes both the labeled and predicted labels so that useful 'weakly' supervised information can be provided to refine (or learn) the graph construction and facilitate the graph convolution operation to estimate the unknown labels.

Topology-Imbalance Node Representation Learning (TINL) The imbalance considered in the existing studies originates from the unequal number of labeled examples with different classes (quantitative imbalance). In Chen et al. (2021), the authors argued that graph data reveal a unique source of imbalance in the asymmetric topological properties of the labeled nodes, i.e., the labeled nodes are not equally important with respect to their structural role in the graph (topological imbalance).

Robust Residual Graph Learning Networks via Similarity (RGLN) In Tang et al. (2021), the authors proposed a paradigm for learning residual graphs to infer the connectivity of edges and weights in graphs. It is presented as metric distance learning under the assumption of low rank and similarity-preserving regularization. It learns the underlying graph based on a similarity-preserving mapping on graphs that keeps similar nodes close and pushes away dissimilar nodes.

From the above literature, we see that existing methods only focus one particular limitation and none of them has can overcome all the weaknesses in a uniform manner.

3. Proposed approach: ReNode-GLCNMR

In this work, we propose the ReNode-GLCNMR method for semi-supervised graph learning, which aims to overcome all the above-mentioned shortcomings in a unified network architecture. The framework of the proposed approach is depicted in Fig. 1. Inspired by that of GLCN (Jiang et al., 2019), our framework includes a graph learning layer and multiple GCN layers. Moreover, our proposal devises three types of loss terms: (i) weighted cross entropy loss (relevant labeled nodes have large weights), (ii) graph learning loss, and (iii) label smoothing loss.

The first layer of the architecture is used to learn an optimal adaptive graph representation $\mathbf{S}$, as described in Eq. (4) (Jiang et al., 2019). This adaptive graph is used in subsequent graph convolution blocks and label smoothing.

Given an input data matrix $\mathbf{X} = [\mathbf{x}_1;\ \mathbf{x}_2; \ldots ; \mathbf{x}_n] \in \mathbf{R}^{n\times d}$, we aim to estimate a non-negative graph $S_{ij} = g(\mathbf{x}_i,\ \mathbf{x}_j)$ representing the pairwise similarity between data $\mathbf{x}_i$ and $\mathbf{x}_j$. It implements $g(\mathbf{x}_i, \mathbf{x}_j)$ over a single-layer neural network parameterized by the weight vector $\mathbf{a} = (a_1,\ a_2, \ldots , a_p)^T \in \mathbf{R}^{p\times 1}$. Instead of explicitly estimating the n^2 elements of the graph matrix $\mathbf{S}$, we only need to learn the vector of weight $\mathbf{a}$ that completely defines the graph $\mathbf{S}$ using the following:

$$S_{ij} = g\left(\tilde{\mathbf{x}}_i,\ \tilde{\mathbf{x}}_j\right) = \frac{\exp(ReLU\left(\mathbf{a}^T\left|\tilde{\mathbf{x}}_i -\ \tilde{\mathbf{x}}_j\right|\right))}{\sum_{j=1}^{n}\exp(ReLU\left(\mathbf{a}^T\left|\tilde{\mathbf{x}}_i -\ \tilde{\mathbf{x}}_j\right|\right))}\quad \forall\, i, j = 1, \ldots , n \tag{4}$$

where $\tilde{\mathbf{x}}_i$ is a low-dimensional representation of $\mathbf{x}_i$. We have $\tilde{\mathbf{x}}_i = \mathbf{x}_i\,\mathbf{Q}$ where $\mathbf{Q} \in \mathbf{R}^{d\times p}$ and $p < d$. The linear transformation $\mathbf{Q}$ is also a learnable matrix, which is included in the graph learning layer (Fig. 1). We use the projection matrix $\mathbf{Q}$ to perform graph learning in a low-dimensional subspace, as more robustness and efficiency can be gained. Eq. (4) guarantees the non-negativity of the graph matrix $\mathbf{S}$ (i.e., $S_{ij} \geq 0$) as well as that the sum of each row in $\mathbf{S}$ is equal to one, and the optimal weight vector $\mathbf{a}$ is determined by minimizing the following loss function:

$$\mathcal{L}_{GL}(\mathbf{Q}, \mathbf{a}) = \sum_{i,j=1}^{n} \left\|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\right\|_2^2 S_{ij} + \lambda\, \|\mathbf{S}\|_F^2 \tag{5}$$

The layer-wise propagation in Eq. (1) becomes:

$$\mathbf{X}^{(k+1)} = \sigma(\mathbf{D}_s^{-\frac{1}{2}}\mathbf{S}\mathbf{D}_s^{-\frac{1}{2}}\mathbf{X}^{(k)}\mathbf{W}^{(k)}) \tag{6}$$

The final perceptron layer in Eq. (2) becomes as follows:

$$\mathbf{Z} = softmax(\mathbf{D}_s^{-\frac{1}{2}}\mathbf{S}\mathbf{D}_s^{-\frac{1}{2}}\mathbf{X}^{(K-1)}\mathbf{W}^{(K-1)}) \tag{7}$$

We also re-weight the labeled nodes as described in Chen et al. (2021). In this work, the labeled nodes are weighted according to their location from the boundaries of classes. This is achieved by calculating the Totoro score. In order to measure the node influence distribution with every labeled node, Personalized PageRank matrix $\mathbf{P}$ is calculated as:

$$\mathbf{P} = \alpha\,(\mathbf{I} - (1-\alpha)\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})^{-1} \tag{8}$$

where $\mathbf{I}$ denotes the identity matrix and $\alpha \in (0, 1]$ denotes the probability of restarting the random walk. Here we use the initial precomputed graph $\mathbf{A}$, where the graph matrix $\mathbf{A}$ denotes an initial graph associated with the dataset. In citation network datasets, $\mathbf{A}$ is the citation links provided by a binary adjacency matrix. In image datasets, it can be constructed using any graph construction technique, such as the KNN graph.

T_i is the Totoro value of the node i. It measures the topological proximity of node i to the center of the class to which it belongs. It is calculated as follows:

$$T_i = \mathbb{E}_{x\sim \mathbf{P}_i}:[\sum_{j\in[1,c],j\neq \mathbf{y}_i} \frac{1}{|C_j|}\sum_{k\in C_j} P_{k,x}] \tag{9}$$

where $\mathbf{y}_i$ denotes the ground-truth label of node i, c is the number of classes, $(C_1, C_2, \ldots, C_c)$ is the training sets for different classes. Since the relevance of a labeled node i is indicated by a small Totoro value, the weight w_i can be calculated using the following cosine wave based mapping:

$$w_i = w_{min} + \frac{1}{2}\,(w_{max} - w_{min})\left(1 + \cos\left(\frac{Rank\,(T_i)}{|L|}\pi\right)\right),\ i \in L \tag{10}$$

where w_{min} and w_{max} are the lower and upper bound of the weight correction factor, and rank(T_i) denotes the ranking order of T_i in the labeled set of samples. With the re-weighted labeled nodes, we can get a new cross-entropy loss function based on Eq. (3). This new cross-entropy loss function $\mathcal{L}_{Semi-ReNode}$ is calculated by:

$$\mathcal{L}_{Semi-ReNode} = -\frac{1}{|L|}\sum_{i\in L} w_i \sum_{j=1}^{c} Y_{ij}\, lnZ_{ij} \tag{11}$$

where $\mathbf{Z}$ is defined in Eq. (7). Our third loss (unsupervised term) enforces the smoothness of the predicted labels of the entire data (Kejani et al., 2020) using the constructed graph. The label regularization term $\mathcal{L}_{smooth}$ is:

$$\mathcal{L}_{smooth} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\|\mathbf{Z}_i - \mathbf{Z}_j\|^2 S_{ij} = Trace(\mathbf{Z}^T\mathbf{L}\mathbf{Z}) \tag{12}$$

where $Trace(\cdot)$ denotes the trace of the matrix, and the normalized Laplacian matrix $\mathbf{L}$ is $\mathbf{L} = \mathbf{I} - \mathbf{D}_s^{-\frac{1}{2}}\mathbf{S}\,\mathbf{D}_s^{\frac{1}{2}}$. We emphasize that $\mathcal{L}_{smooth}$ depends on all unknowns, namely the transformation $\mathbf{Q}$, the graph parametrization $\mathbf{a}$, and the linear transformations of the GCN blocks $\mathbf{W}^{(0)}, \ldots, \mathbf{W}^{(K-1)}$. Thus, minimizing this value leads to a certain optimality of both the graph structure and the output representation. In our method, the constructed graph is constrained by both the input and output data, while the GLCN method constrains the graph only by the input data.

Our proposed model ReNode-GLCNMR is trained by minimizing the following global loss function $\mathcal{L}_{ReNode-GLCNMR}$

$$\mathcal{L}_{ReNode-GLCNMR} = \mathcal{L}_{Semi-ReNode} + \lambda_1\,\mathcal{L}_{GL} + \lambda_2\,\mathcal{L}_{smooth} \tag{13}$$

where $\mathcal{L}_{Semi-ReNode}$, $\mathcal{L}_{GL}$ and $\mathcal{L}_{smooth}$ are defined in Eqs. (11), (5) and (12), respectively. λ_1 and λ_2 are two hyperparameters. Learning the linear transformation $\mathbf{Q}$, the graph parametrization vector $\mathbf{a}$, and the transformations $\mathbf{W}^{(k)}$ are carried out by minimizing the global loss (13).

Specially, if we set labeled node weight $w_i = 1$, we have cross-entropy loss function $\mathcal{L}_{Semi}$ without ReNode:

$$\mathcal{L}_{Semi} = -\sum_{i\in L}\sum_{j=1}^{c} Y_{ij}\, ln\, Z_{ij} \tag{14}$$

where $\mathbf{Z}$ is defined in Eq. (7). The global loss function without ReNode is $\mathcal{L}_{GLCNMR}$:

$$\mathcal{L}_{GLCNMR} = \mathcal{L}_{Semi} + \lambda_1\,\mathcal{L}_{GL} + \lambda_2\,\mathcal{L}_{smooth} \tag{15}$$

4. Performance evaluation

4.1. Datasets

To test the effectiveness of our proposed method ReNode-GLCNMR on semi-supervised learning problems, we perform experiments on eight benchmark datasets, including three widely used Plantoid Paper Citation Graphs (Citeseer, Cora, and Pubmed Sen et al., 2008), a co-authorship graph (Coauthor CS Shchur, Mumme, Bojchevski, & Gunnemann, 2018) based on the Microsoft Academic Graph, and four image datasets (CIFAR10 Krizhevsky & Hinton, 2009, SVHN Netzer, Wang, Coates, Bissacco, & Ng, 2011, MNIST,[2] and Scene15 Jiang, Lin, & Davis, 2013; Lazebnik, Schmid, & Ponce, 2006). Each image in CIFAR10 or SVHN is a 32 × 32 RGB image, each image in MNIST consists of handwritten digits from ‘0’ to ‘9’, and Scene15 has 4485 scene images. The number of images in CIFAR10, SVHN and MNIST datasets is large, but we select only 1000 in each class and get a total of 10,000 images from each dataset. More details about these datasets can be found in Table 1.

4.2. Experimental setup

In this section, we compare our models GLCNMR and ReNode-GLCNMR with four baselines GCN (Kipf & Welling, 2016),[3] GLCN (Jiang et al., 2019),[4] ReNode-GCN (TINL) (Chen et al., 2021),[5] and

[2] http://yann.lecun.com/exdb/mnist/.
[3] https://github.com/tkipf/gcn.
[4] https://github.com/jiangboahu/GLCN-tf.
[5] https://github.com/victorchen96/ReNode.

Table 1
Datasets used in our experiments.

	Dataset	# Nodes	# Labeled	# Edges	# Features	# Classes
Citation dataset	CiteSeer	3 327	120	4 732	3703	6
	CORA	2 708	140	5 429	1433	7
	PubMed	19 717	60	44 338	500	3
Co-authorship dataset	Coauthor CS	18 333	300	81 894	6805	15
Image dataset	CIFAR10	10 000	1000	–	2048	10
	SVHN	10 000	1000	–	2048	10
	MNIST	10 000	1000	–	784	10
	Scene15	4 485	750	–	3000	15

GCNMR (Kejani et al., 2020),[6] which are related to our model. We also compare our models with three other Graph Neural Network methods, including Graph Attention Networks (GAT) (Velikovi et al., 2017),[7] DeepWalk (Perozzi, Al-Rfou, & Skiena, 2014),[8] and RGLN (Tang et al., 2021).[9] The code of GCN, GLCN, ReNode-GCN (TINL), RGLN and GAT used in our experiments is provided by the authors, and we use the default parameters in their code. To ensure the fairness of our experimental results, our GLCNMR and ReNode-GLCNMR models use the same parameters as GLCN (Jiang et al., 2019). More specifically, the number of layers is set to two, the number of features in the dimension reduction module is set to 70 ($p = 70$), and the number of features in the hidden layers is set to 30 ($d_1 = 30$). This configuration is used in these models GCN, GLCN, GCNMR, GLCNMR, and ReNode-GLCNMR.

For the CIFAR10, SVHN and MNIST image datasets, we randomly select 1000 images (100 images per class) for training and 1000 images (100 images per class) for validation. The remaining 8000 images are used as test samples. For the Scene15 image dataset, we randomly select 750 images (50 images per class) for training and use 500 images for validation, the remaining images are for testing. To reduce the effect of random selection, we run all our experiments with ten different splits of training, validation, and testing, and report the average results and the associated standard deviation. To fairly compare the models, we first randomly split all datasets into ten different splits, then fix the ten splits and run the eight competing models with these fixed ten splits.

For the three citation graphs and the Coauthor CS, we follow the evaluation protocol of studies (Chen et al., 2021; Yang, Cohen, & Salakhudinov, 2016) and randomly select 20 examples per class for training and 30 examples in each class for validation; all remaining examples are for testing. As with the image datasets, the results are averaged over ten different splits.

In Shchur et al. (2018), it was shown that different splits of the citation datasets Citeseer, Cora and Pubmed can lead to a completely different result and ranking. This study recommended the use the same split for fair comparison. Following this recommendation, in a group of experiments, we also used with the same split used by GLCN (Jiang et al., 2019) for the Citeseer and Cora datasets. We used the maximum of 5000 epochs and set an early stop at 100.

Since Jiang et al. (2019) trains GLCN for a maximum of 5000 epochs, we also reported experiments with the maximum of 5000 epochs and an early stop set of 100 on image datasets. Note that the DeepWalk method is different from other GCN-based models and does not depend on epochs like other models.

For more details on the parameters setting of our experiment, see Appendix. Table 7 shows the parameters applicable to the 8 datasets in our ReNode-GLCNMR.

6 https://github.com/BiJingjun/GCNMR.
7 https://github.com/PetarV-/GAT.git.
8 https://github.com/DSE-MSU/DeepRobust.git.
9 https://github.com/ashawkey/JLGCN.git.

4.3. Experimental results and method comparison

Tables 2 and 3 show the mean and standard deviation of accuracy (ACC), weighted-F1 (W-F1), and macro-F1 (M-F1) for the four image datasets used. These tables summarize the performance of semi-supervised classification by nine models. We experimented with both the maximum of 200 epochs and 5000 epochs, and all results were averaged over ten splits. The best results are shown in bold and the second best underlined. Since the DeepWalk method is different from other GCN-based models and does not depend on epochs as other models do, it appears only in Table 3. We can conclude that the proposed ReNode-GLCNMR model outperforms the other competing methods on image datasets for the three metrics used regardless of accuracy, Weighted-F1, and Macro-F1 in many configurations.

Table 4 shows the results for the three citation graphs and one co-authorship graph (Coauthor CS). We used a maximum of 200 epochs and averaged all results over ten splits. We can see that the accuracy obtained by the proposed model is good. The obtained Weighted-F1 and Macro-F1 are slightly worse. We emphasize that these dataset not only the classes are imbalanced but also the amount of labeled data used for these four datasets can be very small, which can benefit some models on the expense of other models.

Table 5 shows the results of the eight competing models using the same split as GLCN (Jiang et al., 2019) on the Citeseer and Cora datasets and using the maximum of 5000 epochs. In this split, our proposed models GLCNMR and ReNode-GLCNMR outperform the other competing models including the GLCN model. The obtained accuracy in Table 5 is higher than that in Table 4. the reason is twofold. First, the splits used in the two tables are not exactly the same. Second, they have different number of nodes for validation and testing. Take Citeseer for example, Table 5 uses the same split as Jiang et al. (2019), it has 500 nodes for validation. Table 4 uses the split that has 30 nodes per class for validation and the nodes fall into six classes, so it has 180 nodes for validation.

4.4. Ablation study

The proposed method ReNode-GLCNMR integrates the weight w_i of the labeled nodes and a smoothing term $Trace(\mathbf{Z}^T\mathbf{LZ})$. We conducted an ablation study aiming to quantify the effect of each component. Table 6 shows the results of the ablation study. The same split is used as in Table 5 for the Citeseer and Cora datasets, and the same split as in Tables 2 and 3 for the Scene15 image dataset. The maximum number of epochs is set to 200 for Citeseer and Scene15 and 5000 for Cora. From this ablation study, we can see that the ReNode weight w_i can improve the Weighted-F1 or Macro-F1, but cannot achieve better accuracy. If we use the ReNode weight w_i and the smoothing term $Trace(\mathbf{Z}^T\mathbf{LZ})$ simultaneously, we can get a better overall result for the three metrics accuracy, Weighted-F1 and Macro-F1.

Table 2
Averaged accuracy (%), weighted F1 (%) and macro F1 (%) on the SVHN, CIFAR10, MNIST and Scene15 datasets. The maximum number of epochs is set to 200.

	Method (Year)	MNIST	CIFAR10	SVHN	Scene15
	GCN (2016)	86.89 ± 0.4	60.86 ± 0.8	60.12 ± 1.9	85.43 ± 1.0
	ReNode-GCN (2021)	83.39 ± 0.6	56.91 ± 0.9	71.11 ± 0.8	80.11 ± 10.2
	GAT (2017)	86.60 ± 1.0	55.88 ± 0.9	67.80 ± 1.3	12.64 ± 6.7
	DeepWalk (2014)	–	–	–	–
ACC	RGLN (2021)	89.12 ± 0.7	24.98 ± 3.5	28.03 ± 3.6	10.46 ± 0.3
	GCNMR (2020)	87.54 ± 0.7	61.09 ± 0.7	73.06 ± 0.6	85.47 ± 0.9
	GLCN (2019)	92.85 ± 0.2	61.12 ± 0.7	74.97 ± 0.5	91.77 ± 0.4
	GLCNMR (Ours)	92.85 ± 0.2	61.12 ± 0.7	74.97 ± 0.5	92.95 ± 0.4
	ReNode-GLCNMR (Ours)	**93.01** ± **0.2**	**62.16** ± **0.6**	**75.54** ± **0.5**	**93.21** ± **0.4**
	GCN (2016)	77.37 ± 0.4	48.89 ± 0.7	46.68 ± 1.3	64.18 ± 1.7
	ReNode-GCN (2021)	68.63 ± 0.5	42.96 ± 0.7	57.03 ± 4	58.87 ± 10.2
	GAT (2017)	86.42 ± 1.1	54.49 ± 0.9	67.12 ± 1.5	11.50 ± 6.9
	DeepWalk (2014)	–	–	–	–
W_F1	RGLN (2021)	89.13 ± 0.7	16.94 ± 4.8	20.07 ± 5.2	3.52 ± 1.1
	GCNMR (2020)	79.00 ± 0.4	50.66 ± 0.9	65.53 ± 1.6	64.48 ± 1.1
	GLCN (2019)	90.94 ± 0.3	56.40 ± 0.8	71.02 ± 0.6	89.01 ± 1.0
	GLCNMR (Ours)	90.94 ± 0.3	56.40 ± 0.8	71.02 ± 0.6	89.87 ± 0.6
	ReNode-GLCNMR (Ours)	**91.25** ± **0.2**	**57.8** ± **0.5**	**71.77** ± **0.7**	**90.08** ± **0.8**
	GCN (2016)	77.37 ± 0.4	48.89 ± 0.7	46.68 ± 1.3	64.00 ± 1.6
	ReNode-GCN (2021)	68.63 ± 0.5	42.96 ± 0.7	57.03 ± 0.4	58.80 ± 10.0
	GAT (2017)	86.42 ± 1.1	54.49 ± 0.9	67.12 ± 1.5	11.47 ± 6.9
	DeepWalk (2014)	–	–	–	–
M_F1	RGLN (2021)	89.13 ± 0.7	16.94 ± 4.8	20.07 ± 5.2	2.88 ± 1.1
	GCNMR (2020)	79.00 ± 0.4	50.66 ± 0.9	65.53 ± 1.6	64.14 ± 1.1
	GLCN (2019)	90.94 ± 0.3	56.40 ± 0.8	71.02 ± 0.6	88.66 ± 1.0
	GLCNMR (Ours)	90.94 ± 0.3	56.40 ± 0.8	71.02 ± 0.6	89.81 ± 0.5
	ReNode-GLCNMR (Ours)	**91.25** ± **0.2**	**57.80** ± **0.5**	**71.77** ± **0.7**	**89.98** ± **0.5**

Table 3
Averaged accuracy (%), weighted F1 (%) and macro F1 (%) on the SVHN, CIFAR10, MNIST and Scene15 datasets. The maximum number of epochs is set to 5000.

	Method (Year)	MNIST	CIFAR10	SVHN	Scene15
	GCN (2016)	91.66 ± 0.2	64.31 ± 0.9	78.50 ± 0.3	87.01 ± 2.4
	ReNode-GCN (2021)	90.96 ± 0.2	65.17 ± 0.7	**78.79** ± **0.5**	86.87 ± 3.0
	GAT (2017)	92.61 ± 0.2	60.69 ± 1.1	75.86 ± 0.5	86.93 ± 5.2
	DeepWalk (2014)	90.28 ± 1.4	58.93 ± 0.4	75.92 ± 0.3	**96.77** ± **0.1**
ACC	RGLN (2021)	89.63 ± 0.7	37.79 ± 3.1	40.06 ± 5.8	10.93 ± 0.4
	GCNMR (2020)	91.90 ± 0.2	64.37 ± 0.7	78.17 ± 0.4	87.89 ± 0.5
	GLCN (2019)	**93.97** ± **0.2**	65.89 ± 0.4	78.49 ± 0.3	95.78 ± 0.2
	GLCNMR (Ours)	**93.97** ± **0.2**	65.89 ± 0.4	78.49 ± 0.3	95.78 ± 0.2
	ReNode-GLCNMR (Ours)	93.92 ± 0.5	**66.18** ± **0.4**	78.68 ± 0.4	95.86 ± 0.2
	GCN (2016)	84.49 ± 0.4	50.68 ± 0.7	68.77 ± 0.8	67.97 ± 4.4
	ReNode-GCN (2021)	81.08 ± 0.4	51.97 ± 0.7	68.99 ± 0.5	68.00 ± 5.0
	GAT (2017)	92.59 ± 0.2	59.64 ± 1.3	75.61 ± 0.6	86.93 ± 5.1
	DeepWalk (2014)	90.28 ± 1.4	58.58 ± 0.3	75.81 ± 0.3	**96.77** ± **0.1**
W_F1	RGLN (2021)	89.62 ± 0.7	36.26 ± 3.0	39.11 ± 6.8	4.18 ± 0.7
	GCNMR (2020)	84.73 ± 0.4	50.71 ± 0.8	67.91 ± 0.6	67.65 ± 0.9
	GLCN (2019)	92.71 ± 0.1	61.68 ± 0.7	75.86 ± 0.2	94.73 ± 0.3
	GLCNMR (Ours)	92.71 ± 0.1	61.68 ± 0.7	75.86 ± 0.2	94.73 ± 0.3
	ReNode-GLCNMR (Ours)	**92.72** ± **0.3**	**61.81** ± **0.5**	**76.16** ± **0.3**	94.82 ± 0.3
	GCN (2016)	84.49 ± 0.4	50.68 ± 0.7	68.77 ± 0.8	67.67 ± 4.4
	ReNode-GCN (2021)	81.08 ± 0.4	51.97 ± 0.7	68.99 ± 0.5	67.71 ± 4.9
	GAT (2017)	92.59 ± 0.2	59.64 ± 1.3	75.61 ± 0.6	86.77 ± 5.0
	DeepWalk (2014)	90.28 ± 1.4	58.58 ± 0.3	75.81 ± 0.3	**96.65** ± **0.2**
M_F1	RGLN (2021)	89.62 ± 0.7	36.26 ± 3.0	39.11 ± 6.8	3.67 ± 0.7
	GCNMR (2020)	84.73 ± 0.4	50.71 ± 0.8	67.91 ± 0.6	67.06 ± 1.0
	GLCN (2019)	92.71 ± 0.1	61.68 ± 0.7	75.86 ± 0.2	94.56 ± 0.3
	GLCNMR (Ours)	92.71 ± 0.1	61.68 ± 0.7	75.86 ± 0.2	94.56 ± 0.3
	ReNode-GLCNMR (Ours)	**92.72** ± **0.3**	**61.81** ± **0.5**	**76.16** ± **0.3**	94.64 ± 0.3

4.5. Sensitivity to parameters

Similar to many semi-supervised methods that use balancing parameters, we opted for a grid search over a fixed range of values. Our proposed model has two independent balancing parameters λ_1 and λ_2. For λ_1 we use the set of these values {0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 20, 30, 40, 50, 100, 200, 400, 600} and for λ_2 the following set {0.000001, 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 20, 30, 40, 50, 100, 200, 300, 400, 500, 1000, 10 000, 100 000, 1 000 000}. A validation procedure was performed to select the best combination of parameters. The last two columns in Table 7 summarizes the best values of λ_1 and λ_2 for each dataset used. The same table shows the value used for other parameters α, w_{min}, and w_{max} needed for estimating the weight of the labeled samples.

We also investigated the influence of different values of the parameters λ_1 and λ_2 on the results of our ReNode-GLCNMR. Fig. 2 shows the accuracy of our ReNode-GLCNMR on the Cora

Table 4
Comparison results on the Citeseer, Cora, Pubmed and Coauthor CS datasets.

	Method (Year)	Citeseer	Cora	Pubmed	Coauthor CS
ACC	GCN (2016)	68.22 ± 1.5	79.12 ± 1.3	75.84 ± 1.7	90.99 ± 0.4
	ReNode-GCN (2021)	66.37 ± 2.5	76.84 ± 1.1	75.84 ± 1.7	90.73 ± 0.6
	GAT (2017)	68.41 ± 1.4	77.71 ± 3.3	73.45 ± 2.0	90.30 ± 0.6
	DeepWalk (2014)	49.66 ± 1.3	70.50 ± 0.8	72.62 ± 1.5	83.78 ± 0.7
	RGLN (2021)	67.02 ± 1.9	79.02 ± 1.3	**76.23 ± 1.8**	88.99 ± 0.3
	GCNMR (2020)	68.45 ± 1.6	79.14 ± 1.1	75.55 ± 2.1	90.84 ± 0.7
	GLCN (2019)	67.95 ± 1.7	79.01 ± 1.4	74.84 ± 2.1	91.39 ± 0.5
	GLCNMR (Ours)	68.37 ± 1.1	79.07 ± 1.4	74.91 ± 2.0	91.43 ± 0.5
	ReNode-GLCNMR (Ours)	**70.27 ± 1.2**	**79.19 ± 1.0**	75.11 ± 2.0	**91.45 ± 0.5**
W_F1	GCN (2016)	58.25 ± 1.5	70.89 ± 1.2	71.48 ± 1.9	84.81 ± 0.5
	ReNode-GCN (2021)	50.58 ± 1.8	62.13 ± 1.5	71.55 ± 1.9	84.35 ± 0.7
	GAT (2017)	**68.73 ± 1.0**	78.11 ± 3.1	72.93 ± 2.3	**90.30 ± 0.6**
	DeepWalk (2014)	50.60 ± 1.1	70.71 ± 0.8	72.65 ± 1.5	83.94 ± 0.7
	RGLN (2021)	67.06 ± 1.5	**79.18 ± 1.2**	**76.20 ± 1.9**	89.12 ± 0.3
	GCNMR (2020)	60.62 ± 1.9	71.54 ± 0.6	70.79 ± 2.4	84.54 ± 0.7
	GLCN (2019)	59.34 ± 1.3	71.55 ± 0.7	72.50 ± 2.2	89.04 ± 0.5
	GLCNMR (Ours)	59.88 ± 1.4	71.59 ± 1.3	72.70 ± 1.9	89.10 ± 0.5
	ReNode-GLCNMR (Ours)	61.03 ± 0.6	71.04 ± 0.5	72.87 ± 2.0	88.92 ± 0.5
M_F1	GCN (2016)	54.43 ± 1.3	69.33 ± 1.3	71.30 ± 1.8	80.03 ± 0.4
	ReNode-GCN (2021)	46.61 ± 1.6	60.70 ± 0.9	71.38 ± 1.8	79.39 ± 0.5
	GAT (2017)	**64.67 ± 0.8**	76.79 ± 3.1	72.98 ± 2.1	**87.66 ± 0.6**
	DeepWalk (2014)	46.76 ± 1.1	69.56 ± 1.2	70.85 ± 1.9	78.79 ± 1.0
	RGLN (2021)	62.86 ± 1.5	**77.46 ± 1.4**	**76.02 ± 1.7**	85.19 ± 0.9
	GCNMR (2020)	56.66 ± 1.8	70.31 ± 0.8	70.64 ± 2.1	78.01 ± 1.0
	GLCN (2019)	55.60 ± 1.1	69.84 ± 0.6	72.23 ± 2.1	85.86 ± 0.5
	GLCNMR (Ours)	55.95 ± 1.2	70.14 ± 1.5	72.50 ± 1.7	86.11 ± 0.5
	ReNode-GLCNMR (Ours)	56.92 ± 0.8	69.20 ± 1.5	72.63 ± 1.9	85.69 ± 0.4

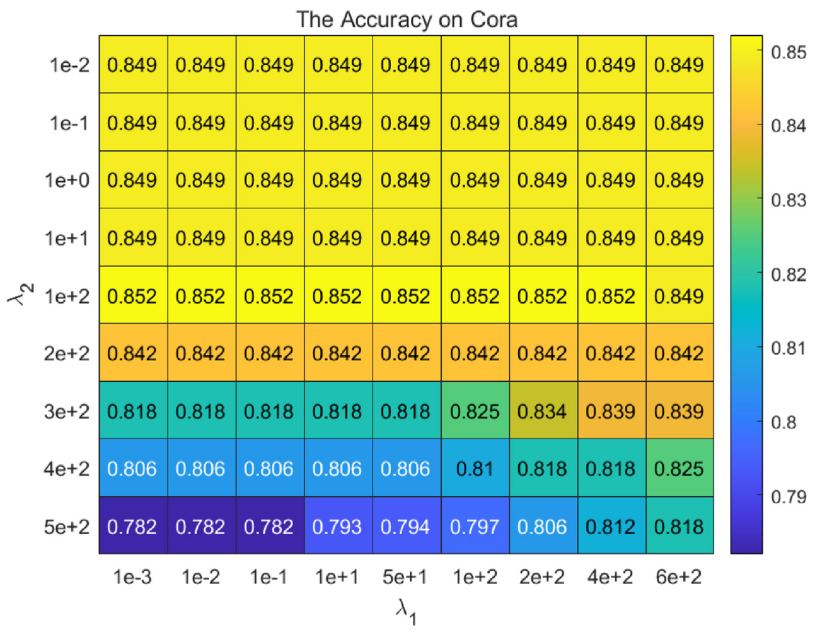


Fig. 2. The accuracy of ReNode-GLCNMR as a function of λ_1 and λ_2 on dataset Cora.

Table 5
Accuracy on datasets Citeseer and Cora use the split same as GLCN (Jiang et al., 2019).

Method	Citeseer	Cora
GCN	70.20	83.50
GAT	71.70	83.60
DeepWalk	52.00	76.90
RGLN	70.30	83.70
GCNMR	71.80	84.20
GLCN	71.90	85.20
GLCNMR (Ours)	74.00	**85.60**
ReNode-GLCNMR (Ours)	**74.20**	85.20

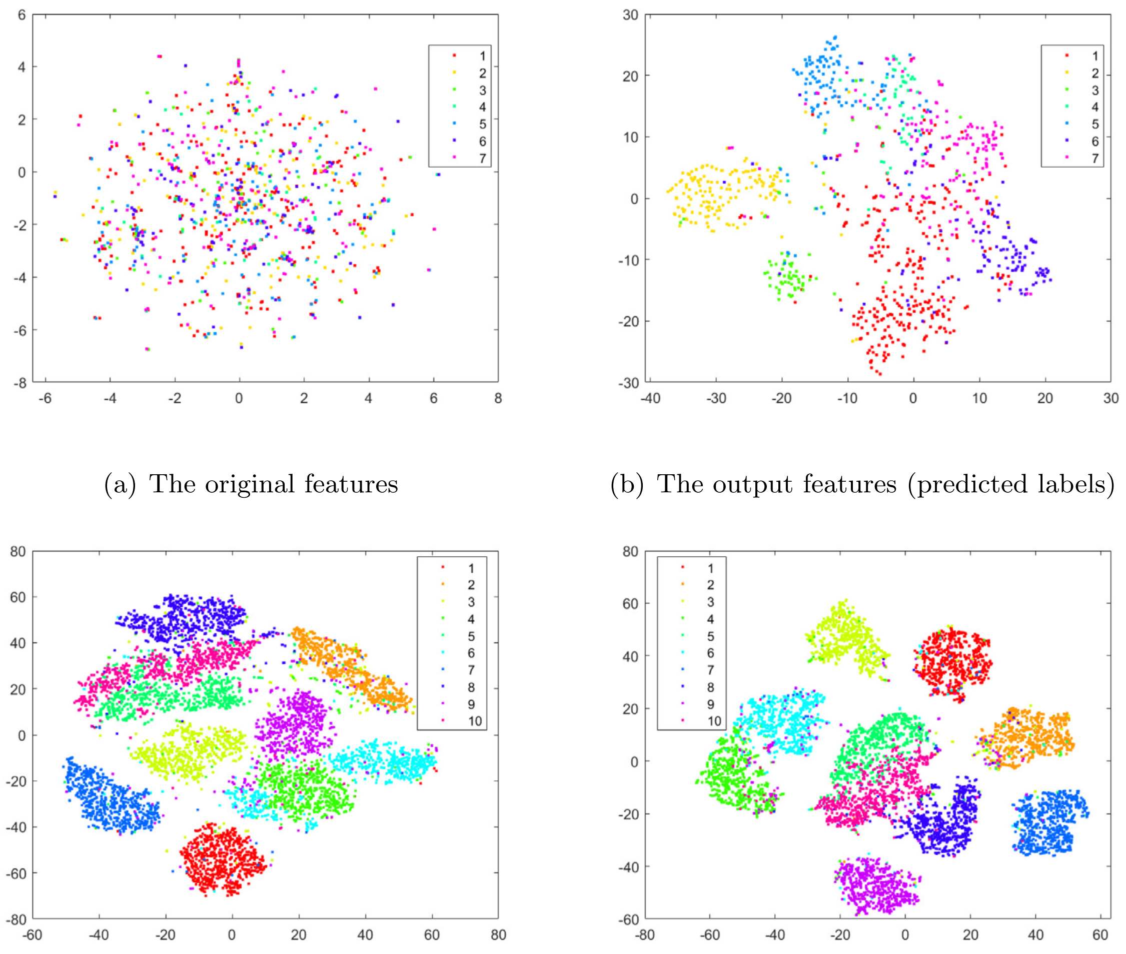
dataset with different λ_1 and λ_2. It can be seen that the accuracy first increases and then decreases with the increase of λ_2. In general, the accuracy is stable for a large domain of values of λ_1 and λ_2.

4.6. Visualization

Finally, we visualize the distribution of nodes according to ReNode-GLCNMR with the t-SNE projection. Fig. 3.a shows the t-SNE projection of 1000 nodes in the Cora datasets using the original features. Fig. 3.b is the distribution of the same nodes after ReNode-GLCNMR. Fig. 3.c illustrates the t-SNE projection of 8000 test images in the MNIST dataset using the original

Table 6
Ablation study.

	Variant	ReNode	Smoothness	Citeseer	Cora	Scene15
ACC	GLCN	×	×	73.20	85.20	91.77 ± 0.4
	GLCNMR	×	✓	72.40	**85.60**	92.95 ± 0.4
	ReNode-GLCN	✓	×	73.30	84.90	92.92 ± 0.5
	ReNode-GLCNMR (Ours)	✓	✓	**74.90**	85.20	**93.21 ± 0.4**
W_F1	GLCN	×	×	62.28	76.51	89.01 ± 1.0
	GLCNMR	×	✓	63.27	76.69	89.87 ± 0.6
	ReNode-GLCN	✓	×	**66.81**	**78.15**	**90.08 ± 0.8**
	ReNode-GLCNMR (Ours)	✓	✓	65.58	**78.15**	**90.08 ± 0.8**
M_F1	GLCN	×	×	59.34	74.35	88.66 ± 1.1
	GLCNMR	×	✓	60.50	73.51	89.81 ± 0.5
	ReNode-GLCN	✓	×	**63.14**	75.71	89.82 ± 0.9
	ReNode-GLCNMR (Ours)	✓	✓	62.49	**75.72**	**89.98 ± 0.5**


(a) The original features (b) The output features (predicted labels)

(c) The original features (d) The output features (predicted labels)

Fig. 3. (a) and (b): t-SNE visualization of (a) the original features and (b) output features by using ReNode-GLCNMR on the Cora dataset. **(c) and (d):** t-SNE visualization of (a) the original features and (b) output features by using ReNode-GLCNMR on the MNIST dataset.

features. Fig. 3.d is the distribution of the same nodes according to ReNode-GLCNMR. As you can be seen from the figures, the representation obtained by ReNode-GLCNMR contributed to the class discrimination.

5. Conclusion

In this work, we proposed the ReNode-GLCNMR approach for graph-based semi-supervised learning problems. ReNode-GLCNMR includes a graph learning layer, multiple convolutional layers, weight estimation of labeled nodes, and label smoothing. The results of our experiments on eight benchmarks show that ReNode-GLCNMR generally performs better than traditional GCNs on several semi-supervised learning problems. This was evident on the image datasets. These results tend to confirm that the proposed model, which simultaneously integrates graph construction and label regularization, leads to better performance of GCN architectures for semi-supervised learning problems. Another issue that arises from the proposed work is the need for a more

Table 7
Parameters applicable to the 8 datasets in our ReNode-GLCNMR.

	α	w_{min}	w_{max}	λ_1	λ_2
CiteSeer	0.15	0.5	1.5	0.01	40
CORA	0.15	0.5	2	0.01	0.01
PubMed	0.15	0.5	100	0.001	0.001
Coauthor CS	0.15	0.5	40	0.00001	0.0001
CIFAR10	0.15	0.5	30	0.01	0.1
SVHN	0.15	0.5	30	0.01	0.1
MNIST	0.15	0.5	30	0.01	0.1
Scene15	0.15	0.5	30	0.01	0.1

rigorous evaluation protocol for datasets representing citation networks.

Declaration of competing interest

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

Data availability

The data that support the findings of this study are available upon reasonable request.

Acknowledgments

This work is part of the grant PID2021-126701OB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe".

Appendix. More details of experiments

- PageRank teleport probability α: [0.05; 0.1; 0.15; 0.2];
- Dimension of graph learning hidden layer: 70;
- Dimension of graph convolution hidden layer: 30;
- Lower bound of the cosine annealing w_{min}: [0.25; 0.5; 0.75];
- Upper bound of the cosine annealing w_{max}: fron 1.25 to 200;
- Weight for L2 loss on embedding matrix: [1e–4; 5e–2];
- Initial Graph Learning Layer learning rate: [0.005];
- Initial Graph Convolution Layer learning rate: [0.005];
- Decay of the learning rate: [0.9];
- Dropout Probability: [0.6];
- λ_1: from 1e–6 to 1e+2;
- λ_2: from 1e–6 to 1e+6;

References

Atwood, J., & Towsley, D. (2016). Diffusion-convolutional neural networks. *Advances in Neural Information Processing Systems*, *29*.

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203.

Chen, D., Lin, Y., Zhao, G., Ren, X., Li, P., Zhou, J., et al. (2021). Topology-imbalance learning for semi-supervised node classification. *Advances in Neural Information Processing Systems*, *34*.

Dai, H., Kozareva, Z., Dai, B., Smola, A., & Song, L. (2018). Learning steady-states of iterative algorithms over graphs. In *International conference on machine learning* (pp. 1106–1114). PMLR.

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

El Traboulsi, Y., Dornaika, F., & Assoum, A. (2015). Kernel flexible manifold embedding for pattern classification. *Neurocomputing*, *167*, 517–527.

Jiang, Z., Lin, Z., & Davis, L. S. (2013). Label consistent K-SVD: Learning a discriminative dictionary for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2651–2664.

Jiang, B., Zhang, Z., Lin, D., Tang, J., & Luo, B. (2019). Semi-supervised learning with graph learning-convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11313–11320).

Kejani, M. T., Dornaika, F., & Talebi, H. (2020). Graph convolution networks with manifold regularization for semi-supervised learning. *Neural Networks*, *127*, 160–167.

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

Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases*, *1*(4).

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*.

Levie, R., Monti, F., Bresson, X., & Bronstein, M. M. (2018). Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, *67*(1), 97–109.

Li, Y., Tarlow, D., Brockschmidt, M., & Zemel, R. (2015). Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493.

Li, R., Wang, S., Zhu, F., & Huang, J. (2018). Adaptive graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 32*.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*.

Peng, H., Du, B., Liu, M., Liu, M., Ji, S., Wang, S., et al. (2021). Dynamic graph convolutional network for long-term traffic flow prediction with reinforcement learning. *Information Sciences*, *578*, 401–416.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). *DeepWalk: Online learning of social representations*. ACM.

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

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

Tang, J., Gao, X., & Hu, W. (2021). RGLN: Robust residual graph learning networks via similarity-preserving mapping on graphs. In *ICASSP 2021 - 2021 IEEE international conference on acoustics, speech and signal processing (ICASSP)*.

Thiede, E. H., Zhou, W., & Kondor, R. (2021). Autobahn: Automorphism-based graph neural nets. arxiv arXiv:2103.01710.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., et al. (2021). MLP-mixer: An all-MLP architecture for vision. arXiv arXiv:2105.01601v4.

Velikovi, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

Wang, H., Zhao, J., Su, Y., & Zheng, C. (2021). scCDG: A method based on DAE and GCN for scRNA-seq data analysis.. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, http://dx.doi.org/10.1109/TCBB.2021.3126641.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24.

Yang, Z., Cohen, W., & Salakhudinov, R. (2016). Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning* (pp. 40–48). PMLR.

Zhang, D., Gao, Y., Zhao, J., Zheng, C., & Liu, J. (2022). A new graph autoencoder-based consensus-guided model for scRNA-seq cell type detection. *IEEE Transactions on Neural Networks and Learning Systems*, http://dx.doi.org/10.1109/TNNLS.2022.3190289.

Zhou, F., Yin, M., Jiao, C., Zhao, J., Zheng, C., & Liu, J. (2021). Predicting miRNA-disease associations through deep autoencoder with multiple kernel learning.. *IEEE Transactions on Neural Networks and Learning Systems*, http://dx.doi.org/10.1109/TNNLS.2021.3129772.

Zhuang, C., & Ma, Q. (2018). Dual graph convolutional networks for graph-based semi-supervised classification. In *Proceedings of the 2018 world wide web conference* (pp. 499–508).