SGCN: Structure and Similarity-Driven Graph Convolutional Network for Semi-Supervised Classification

WenQiang Guo[1], YongLong Hu[2], YongYan Hou[3], BoFeng Xue[4]
School of Electronic Information and Artificial Intelligence, Shaanxi University of Science and Technology, Xi'an, China[1,2,4]
School of Electrical and Control Engineering, Shaanxi University of Science and Technology, Xi'an, China[3]

***Abstract*—Traditional Graph Convolutional Networks (GCNs) primarily utilize graph structural information for information aggregation, often neglecting node attribute information. This approach can distort node similarity, resulting in ineffective node feature representations and reduced performance in semi-supervised node classification tasks. To address these issues, this study introduces a similarity measure based on the Minkowski distance to better capture the proximity of node features. Building on this, SGCN, a novel graph convolutional network, is proposed, which integrates this similarity information with conventional graph structural information. To validate the effectiveness of SGCN in learning node feature representations, two classification models based on SGCN are introduced: SGCN-GCN and SGCN-SGCN. The performance of these models is evaluated on semi-supervised node classification tasks using three benchmark datasets: Cora, Citeseer, and Pubmed. Experimental results demonstrate that the proposed models significantly outperform the standard GCN model in terms of classification accuracy, highlighting the superiority of SGCN in node feature representation learning. Additionally, the impact of different distance metrics and fusion factors on the models' classification capabilities is investigated, offering deeper insights into their performance characteristics. The code and datasets are available at https://github.com/YONGLONGHU/SGCN.git.**

***Keywords*—Graph convolutional networks; semi-supervised node classification; Minkowski distance; similarity information**

I. Introduction

In recent years, Convolutional Neural Networks (CNNs) have achieved rapid advancements in fields such as image recognition and natural language processing, primarily due to their ability to conveniently perform convolutional operations on structured data like images and texts, which exhibit regular patterns [1]. Graph data, on the other hand, is unstructured with irregular connections between nodes, rendering traditional CNN convolution operations difficult to directly apply [2]. Nevertheless, the research and application of graph data hold extensive and profound significance. For instance, by studying knowledge graphs, one can leverage the entity and relationship information within existing knowledge graphs to predict novel facts [3]. The investigation of brain functional networks turns on the diagnosis of brain disorders such as autism and depression [4]. Moreover, the study of molecular networks helps to a deeper understanding of protein functions [5].

In deep learning research, the annotation of vast amounts of data is often required, yet the process of data annotation is labor-intensive, resource-consuming, and time-consuming, especially for graph data such as those found in social networks and biological networks, where the cost of labeling each node or edge is prohibitively high [6]. Semi-supervised learning addresses this challenge by leveraging a small amount of labeled data alongside a large quantity of unlabeled data to learn more powerful models [7]. Learning from graph-structured data primarily involves three tasks: node classification, graph classification, and link prediction [8]. Among these, node classification treats each node as a data sample, utilizing both the information from labeled data and the graph's structural information to predict the classes of unlabeled nodes. This represents a typical semi-supervised learning problem.

Graph Convolutional Networks have achieved great success in the field of semi-supervised node classification for graph data by propagating and aggregating information from neighboring nodes through the graph structure to learn and represent target nodes. GCNs utilize the adjacency matrix to obtain aggregation weights from neighbors for graph convolution [9]. Unlike traditional approaches, Graph Attention Networks (GATs) employ graph attention modules to learn discriminative aggregation weights for neighbor nodes, enabling graph convolution [10]. Given that a node can have numerous neighbors, aggregating information from all neighbors is inefficient. Therefore, GraphSAGE performs graph convolution by sampling a fixed number of neighbors for each node and aggregating their information [11]. However, all these methods fail to fully exploit the original attribute information between nodes. Furthermore, literature [12] theoretically and empirically demonstrates that GCNs tend to disrupt node similarity in the original feature space during information aggregation, reducing the effectiveness of learned node representations and subsequently impacting downstream tasks. Consequently, a new graph convolutional model is designed based on cosine similarity and a self-supervised module to preserve the similarity of original nodes. Nevertheless, cosine similarity describes the mathematical closeness in direction between vectors [13], which is not suitable for characterizing feature proximity between nodes. Additionally, Incorporating a self-supervised module into a graph convolutional network requires the delicate design of pretext tasks based on specific problems [14], which is not conducive to the generalization of the model. The degree of proximity between node features should be more aptly described using distance, and improving graph convolution operations is more conducive to the generalization of the model. Therefore, we propose a similarity measure based on the Minkowski distance [15]

between node features and design a novel graph convolutional network named SGCN that integrates structural information with the similarity information, building upon classic graph convolutional network architectures.

The goal of this paper is to construct a novel similarity measure based on the Minkowski distance and integrate this similarity information with the traditional GCN structural information to design a new graph convolutional network. Essentially, two challenges are addressed. First, how to construct a similarity measure using the feature distances between nodes to describe the proximity between nodes in terms of their features? In this paper, a linear similarity measure based on the Minkowski distance is proposed, which better captures the proximity between node features. Second, how to utilize the constructed similarity measure information to design a new graph convolutional network that can better learn node representations? This paper integrates similarity information with the structural information used in the classical GCN to design a novel graph convolutional network that combines both structural and similarity information to learn better node representations. The contributions of this work can be summarized as follows:

- A linear similarity measure function has been developed in this study, utilizing three specific forms of Minkowski distance: Manhattan distance, Euclidean distance, and Chebyshev distance. This measure enhances the description of similarity between node features.
- By combining the similarity information from the linear similarity measure with the structural information used in conventional GCNs, this study introduces SGCN, a novel graph convolutional network that improves node feature representation.
- Using the SGCN and traditional GCN frameworks, this study constructs two semi-supervised node classification models: SGCN-GCN and SGCN-SGCN. These models are evaluated on the Cora, Citeseer, and PubMed datasets to validate the effectiveness of SGCN and assess the performance of the models with different distance metrics and fusion ratios of structural and similarity information.

The structure of this paper is organized as follows: Section II reviews related work on graph convolutional networks. Section III explores the theoretical foundations of classical graph convolutional networks and semi-supervised classification. Section IV introduces this study proposed model, SGCN, which incorporates similarity information. Section V describes the experimental methods used to evaluate SGCN's advancements. Section VI presents the discussion of the study. Finally, Section VII summarizes the paper and highlights the contributions of this work.

II. Related Works

Graph convolution, as an extension of convolutional operations in structured data to graph-structured data, can be broadly divided into two major classes: spectral-based methods and spatial-based methods [16]. Spectral GCNs leverage Fourier transform to transform signals from the original space to the frequency domain, where multiplication is carried out to address the challenge of defining convolutions on graph structures [17]. However, directly computing graph convolutions is difficult and computationally intensive. Defferrard [18] overcame this by substituting Chebyshev polynomials for graph convolutions, eliminating the need for Laplacian eigendecomposition and reducing complexity. Kipf [19] further simplified graph convolutions by stacking layers of first-order Chebyshev polynomial filters and modifying the propagation matrix, resulting in the GCN model.

Spatial GCNs, on the other hand, define convolutions based on the spatial relationships between nodes. Starting from a node and its neighboring set, spatial GCNs first aggregate information and then combine the aggregated results to update node information [20].The prevalent framework for spatial GCNs is the Message Passing Neural Network (MPNN) [21]. MPNN first apply an aggregation function to each node and its neighbors to capture local structural information. Subsequently, the aggregated results are combined with node features through an update function to obtain new node representations. Battaglia [22] extended MNPP by proposing the Graph Network (GN) architecture to facilitate deep models in learning about entities, relations, and rules. Beyond defining node attributes, the GN architecture introduces edge and global attributes, enabling comprehensive learning of the interrelated properties of these three attributes on graph structures through well-designed update and aggregation functions.

In recent years, numerous researchers have improved GCNs by domain selection or incorporating attention mechanisms. On the one hand, differing from traditional networks like GraphSAGE that directly sample neighbor nodes, Chen [23] introduced FastGCN, which utilizes a novel sampling method for graph convolutional networks. This method interprets graph convolutions as integral transformations of embedding functions under a probabilistic measure and employs Monte Carlo methods to consistently estimate this integral. Zou [24] proposed an innovative hierarchical importance sampling approach that first selects neighbor nodes of central nodes to construct a bipartite graph structure, upon which the importance probability of each node is calculated. Subsequently, a certain number of nodes are probabilistically sampled based on these probabilities.On the other hand, differing from GAT, the Gated Attention Network (GaAN) introduces a self-attention mechanism that computes additional attention scores for each attention head, enriching the application scenarios of graph attention mechanisms and enhancing network performance [25]. Beyond applying graph attention mechanisms to the spatial dimension, GeniePath [26] proposes an LSTM-like gating mechanism that effectively controls information flow between graph convolutional layers, improving the efficiency and performance of graph convolutional networks. Zhang [27] developed a self-attention graph neural network for hypergraphs to process and learn different types of hypergraph information.

While these improvements have enhanced the performance of graph convolutional networks models for specific tasks, the refined models tend to be more complex, with weaker generalization capabilities and limited scalability. In contrast to these efforts, the approach of this work focuses on improvements from a mathematical modeling perspective, achieving

effective graph node feature representations without the need for additional supervised modules or increasing the number of model parameters.

III. Preliminary Study

This work aims to enhance the performance of graph convolutional networks in learning node feature representations by integrating similarity information, and subsequently to improve the accuracy of semi-supervised node classification using the refined GCN model. To achieve this, the relevant definitions of graph data and the fundamental concept of semi-supervised node classification are first introduced in this section. Following that, the classical GCN and the traditional semi-supervised node classification models constructed using GCN are presented.

A. Semi-supervised Node Classification

Let $G = (V, E, X)$ [28] be an undirected graph, where $V = \{v_i | i = 1, 2, \ldots, N\}$ denotes the set of N nodes, $N = L + U$. L and U denote the number of labeled and unlabeled nodes in the set of N nodes, $L \ll U$. $E = \{e_i | i = 1, 2, \ldots, M\}$ denotes the set of M edges. $X = [X_L, X_U] = [x_1, \ldots, x_N]^T \in \mathbb{R}^{N \times F}$ is the feature matrix of all nodes, where X_L denotes the feature matrix of labeled nodes and X_U denotes the feature matrix of unlabeled nodes, each node v_i corresponding to a feature vector $x_i \in \mathbb{R}^F$. The adjacency matrix $A \in \{0, 1\}^{N \times N}$ stores information about the structure of a graph. If $A_{ij} = 1$, it indicates that there is an edge between node v_i and node v_j, otherwise $A_{ij} = 0$. Each node in the graph corresponds to a label. Assuming there are C different types of labels in the graph, for each node v_i, its label $y_i \in \{0, 1, \ldots, C-1\}$. $Y_L \in \mathbb{R}^{L \times C}$ represents the label matrix of the labeled nodes. The objective of semi-supervised learning on graphs is to learn a neural network model $f(\cdot)$ from these known information, which can then predict labels for all unlabeled nodes.

For semi-supervised node learning, the traditional approach uses a graph Laplace regularization term in the loss function based on the principle that connected nodes are likely to share the same labels so that the label information is propagated over the graph [19] as shown in Eq. (1).

$$\begin{cases} \mathcal{L} = \mathcal{L}_0 + \lambda \mathcal{L}_{\text{reg}} \\ \mathcal{L}_{\text{reg}} = \sum_{i,j} A_{ij} \| f(X_i) - f(X_j) \|^2 = f(X)^\top \Delta f(X) \end{cases}, \tag{1}$$

where, $\mathcal{L}$ is the total loss function, the $\mathcal{L}_0$ denotes the supervised loss of labeled nodes, the $\mathcal{L}_{\text{reg}}$ denotes the unsupervised loss of unlabeled nodes, the λ is the weight factor. A_{ij} is the value i-th row and j-th col of adjacency matrix A. $f(\cdot)$ is the learned neural network model, X_i and X_j is the i-th and j-th feature vector of feature matrix. $\Delta = D - A$ represents the non-normalized Laplacian regularization term for the undirected graph G, $D_{ii} = \hat{\sum}_j A_{ij}$ denotes the degree matrix.

B. Graph Convolutional Networks

The fundamental idea of Graph Convolutional Networks lies in leveraging the structural information of graphs to learn new feature representations for each node through information propagation and aggregation. Given a graph G, a multi-layer GCN for semi-supervised node classification follows a hierarchical propagation rule [19] as shown in Eq. (2).

$$H^{(l+1)} = \sigma\left(D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}} H^{(l)} W^{(l)} \right), \tag{2}$$

where $\tilde{A} = A + I_N$ represents the adjacency matrix of the undirected graph G augmented with self-connections, A is the original adjacency matrix of G, and $I_N \in \mathbb{R}^{N \times N}$ is the identity matrix. D denotes the degree matrix of graph G. $\hat{A} = D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}}$ is the normalized adjacency matrix. $H^{(l)}$ and $W^{(l)}$ respectively denote the input feature matrix and shared trainable parameter matrix at the l-th layer of the model.

For each node, the rule for the hierarchical propagation of information [19] can be shown in Eq. (3).

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}_i} \frac{1}{\sqrt{d_i d_j}} h_j^{(l)} W^{(l)} \right), \tag{3}$$

where, N_i represents the set of neighbor nodes of node i, d_i and d_j are the degrees of nodes i and j respectively, $h_j^{(l)}$ denotes the feature vector of node j at the l-th layer, $W^{(l)}$ is the shared trainable parameter matrix at the l-th layer of the model, and σ is the activation function.

Utilizing the aforementioned classical two-layer graph convolutional network to construct a graph neural network model for semi-supervised node classification in graphs has become a popular approach in recent years. The model [19] can be expressed as shown in Eq. (4).

$$f(\hat{A}, X; \theta) = softmax(\hat{A}\text{ReLU}(\hat{A}XW^{(0)})W^{(1)}), \tag{4}$$

where, $\theta = \{W^{(0)}, W^{(1)}\}$ represents the parameter matrices that are optimized through gradient descent to minimize the cross-entropy loss function. $\hat{A} = D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}}$ is the normalized adjacency matrix, $ReLU(.)$ and $softmax(.)$ are the activation functions applied after the first and second layers of the network, respectively. The output of the model is denoted as $H \in R^{N \times C}$, where N is the number of nodes and C is the number of labels for the graph nodes. Each row of H matrix contains the scores for each possible label for a given node.

Utilizing the aforementioned classical two-layer graph convolutional network model $f(X, A)$ for semi-supervised node classification is currently the mainstream approach. This method trains labeled nodes through a supervised loss function $\mathcal{L}_0$ and adjusts parameters via a gradient descent strategy, enabling the model to learn feature representations of both labeled and unlabeled nodes simultaneously, thereby achieving a satisfactory semi-supervised node classification performance. However, during the message passing process, the classification performance is hindered by the insufficient utilization of attribute information between nodes and the fact that convolution can disrupt the original similarity among nodes. This

work is motivated by this observation and aims to address these issues.

IV. Proposed SGCN Architecture

Traditional graph convolutions, in the process of message passing, not only neglect the attribute information between nodes but also disrupt the similarity information among them. Therefore, this work proposes the construction of a similarity measure based on the Minkowski distance and the integration of this similarity information with the structural information commonly used in traditional graph convolutional networks to design a novel graph convolutional network, termed SGCN. In the following sections, the graph convolutional network that integrates similarity information is first introduced, followed by an elaboration on the similarity measure constructed using the Minkowski distance, and finally, the construction of a graph convolutional neural network model based on the proposed SGCN for semi-supervised node classification tasks is presented.

A. Graph Convolutional Network with Integrated Similarity Information

To integrate similarity information for better learning of node embeddings, this study propose a novel graph convolutional network, SGCN, with the following layered message propagation rule as shown in Eq. (5):

$$H^{(l+1)} = \sigma\left((\lambda D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}} + (1-\lambda)\oplus S)H^{(l)}W^{(l)}\right), \quad (5)$$

where, $\lambda \in [0,1]$ is the fusion factor and represents the fusion proportion for the structural information in the original graph convolutional network. $\hat{A} = D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}$ is the normalized adjacency matrix corresponding to the original graph convolutional network, where $D^{-\frac{1}{2}}$ on both sides normalizes the matrix $\tilde{A}$, $\hat{A}_{ij}$ denotes the weight of the feature vector that node i aggregates from its neighbor node j, ranging from 0 to 1. $1-\lambda$ serves as the fusion proportion for the similarity information, also taking values between 0 and 1. S is a similarity matrix constructed based on the linear similarity measure function designed in this paper using the Minkowski distance. $\oplus$ is a unary operator that normalizes the S matrix. $\hat{S} = \oplus S$ represents the normalized similarity matrix, where $\hat{S}_{ij}$ indicates the weight of the feature vector that node i aggregates from its neighbor node j, ranging from 0 to 1.

$H^{(l)}$ and $W^{(l)}$ are the feature matrix and the trainable parameter matrix at the l-th layer, while $H^{(l+1)}$ is the updated feature matrix after the l-th layer.

To aggregate the feature information of neighbor nodes for obtaining a better embedding representation, the original graph convolutional network solely computes the aggregation weights $\hat{A} = D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}$ through structural information without fully utilizing the similarity information between nodes. Based on the principle that nodes with closer features are more likely to belong to the same class, this work integrate the similarity information between nodes, $\hat{S} = \oplus S$, into the aggregation weight calculation to obtain better node feature representations. Both parts of the aggregation weights derived from structural information and similarity information are processed through normalization operations. Finally, the two pieces of information are linearly combined using fusion factors λ and $1-\lambda$ to ensure that the final aggregation weights range from 0 to 1. A smaller value indicates less information aggregated from that neighbor, while a larger value indicates more information aggregated. The proposed method differs from other graph convolutional network models that add self-supervised modules to achieve better classification performance. By integrating similarity information into the hierarchical message propagation mechanism, the approach demonstrates enhanced generalization capabilities and a more streamlined model architecture with reduced parameter count, as opposed to methods that integrate supervised modules.

The hierarchical message propagation rule for each specific node is defined as shown in Eq. (6):

$$h_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}_i}(\frac{\lambda}{\sqrt{d_i d_j}} + (1-\lambda)\oplus Sim(h_i,h_j))h_j^{(l)}W^{(l)}\right), \quad (6)$$

where N_i denotes the set of neighbors of node i. d_i, d_j represent the degrees of nodes i and j respectively. The function $Sim(,.,)$ is a linear similarity measure constructed in this paper, which calculates the similarity between nodes i and j to form the similarity matrix S. Regarding the unary normalization operator $\oplus$, it maintains the similarity value when $Sim(h_i,h_j)$ equals 1, and halves the similarity value calculated by $Sim(h_i,h_j)$ when it does not equals to 1. Specifically, it is defined as shown in Eq. (7):

$$\oplus Sim(h_i,h_j) = \begin{cases} 1, Sim(h_i,h_j) = 1 \\ \frac{Sim(h_i,h_j)}{2}, Sim(h_i,h_j) \neq 1 \end{cases}. \quad (7)$$

In the original graph convolutional network, if a graph node has no neighbors, it can only aggregate all of its own information, meaning the aggregation weight is 1. However, if this node is not an isolated node, the aggregation weight for any neighbor would be less than or equal to 1/2. In the proposed approach, if the similarity between two nodes is 1, it indicates that the graph node is an isolated node, and thus the aggregation weight is set to 1. If the similarity is not 1, similarly to the original graph convolutional network, it is necessary to map the similarity to a value between 0 and 1/2. Here, the classical normalization operator $\hat{S} = D^{-\frac{1}{2}}SD^{-\frac{1}{2}}$ used in GCN is not employed. Instead, the elements that are not equal to 1 are directly divided by 2 to achieve the desired result. This method is supported by extensive experiments, which demonstrate that this straightforward operator design yields superior performance.

B. Linear Similarity Measure based on Minkowski Distance

The Minkowski distance is a method used to measure the distance between two points in a multidimensional space, playing a significant role in quantifying the similarity between sample points. In the task of semi-supervised node classification, nodes with closer features are more likely to belong

to the same class. Therefore, in this study, the Minkowski distance is employed instead of cosine similarity to construct a similarity measure that characterizes the proximity between node features.

In Euclidean space, for two sample points $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$, their Minkowski distance [29] $d(x, y)$ is defined as shown in Eq. (8):

$$d(x,y) = (\sum_{i=1}^{n} \mid x_i - y_i \mid^p)^{\frac{1}{p}}, \tag{8}$$

where p is a positive real-valued parameter that controls the sensitivity of the metric distance.When $p = 1$, it corresponds to the Manhattan distance as shown in Eq. (9):

$$d(x,y) = \sum_{i=1}^{n} \mid x_i - y_i \mid . \tag{9}$$

When $p = 2$, it becomes the Euclidean distance as shown in Eq. (10):

$$d(x,y) = (\sum_{i=1}^{n} \mid x_i - y_i \mid^2)^{\frac{1}{2}}. \tag{10}$$

As p approaches infinity, it converges to the Chebyshev distance as shown in Eq. (11):

$$d(x,y) = \max_i \mid x_i - y_i \mid . \tag{11}$$

Based on the Minkowski distance, for the features h_i, h_j of nodes i and j, this study construct the following linear similarity measure function [Eq. (12)]:

$$\mathrm{Sim(h_i, h_j)} = -\frac{\mathrm{d(h_i, h_j) - D_{min}}}{\mathrm{D_{max} - D_{min}}} + 1, \tag{12}$$

where D is a distance matrix composed of Minkowski distances between different nodes, $D_{\max} = \max \sum_{i,j} d_{ij}$ is the maximum value among all elements in the distance matrix, $D_{\min} = \min \sum_{i,j} d_{ij}$ is the minimum value among all elements in the distance matrix, and $d(h_i, h_j)$ represents the Minkowski distance between the features of node i and node j.

This study normalizes the distance $d(h_i, h_j)$ to a uniform scale by computing $d_{ij} = \frac{d(h_i,h_j)-D_{\min}}{D_{\max}-D_{\min}}$. A value of d_{ij} closer to 1 indicates that the features of nodes i and j are less similar, whereas a value closer to 0 signifies greater similarity. The closer the features, the higher the similarity, and vice versa. This suggests that the similarity measure function needs to be designed in such a way that it maps a smaller d_i to a larger similarity value, and a larger d_i to a smaller similarity value. It requires the function to be monotonically decreasing and have its range within $[0, 1]$ in the interval $[0, 1]$. Based on this principle, various similarity measure functions were designed, such as $Sim(d_{ij}) = \cos\left(\frac{\pi}{2} d_{ij}\right)$, $Sim(d_{ij}) = 1 - \sin\left(\frac{\pi}{2} d_{ij}\right)$, and $Sim(d_{ij}) = \text{sigmoid}(-x) + \frac{1}{2}$. Nonetheless, the comprehensive experimental evaluation revealed that the linear similarity measure function, defined as $Sim(d_{ij}) = -d_{ij} + 1$, consistently outperformed the other methods, showcasing its superior performance.

C. SGCN for Semi-supervised Node Classification

To demonstrate the effectiveness of the proposed SGCN in learning graph node feature representations, this work follows the same approach as Kipf [19], employing a two-layer graph convolutional network for semi-supervised node classification. Two graph convolutional neural network models, SGCN-GCN and SGCN-SGCN, are constructed to tackle the semi-supervised node classification task. Here, GCN refers to the original classical graph convolutional network. The SGCN-GCN model is defined as shown in Eq. (13):

$$\begin{aligned} &f_{SGCN-GCN}(\hat{A}, X; \theta) = \\ &softmax(\hat{A}\mathrm{ReLU}((\lambda\hat{A} + (1-\lambda)\hat{S})XW^{(0)})W^{(1)}) \end{aligned}, \tag{13}$$

where $\mathrm{ReLU}((\lambda\hat{A} + (1-\lambda)\hat{S})XW^{(0)})$ represents the first layer of this work proposed SGCN. The second layer, $softmax(\hat{A}X^{(1)}W^{(1)})$, corresponds to the classical GCN network before improvement, where $\mathrm{X^{(1)}} = \mathrm{ReLU}((\lambda\hat{A} + (1-\lambda)\hat{S})XW^{(0)})$ is the graph node feature matrix output by the first convolutional layer. The schematic diagram of the constructed model is shown in Fig. 1, where the first layer is the proposed SGCN, and the second layer is the traditional GCN.

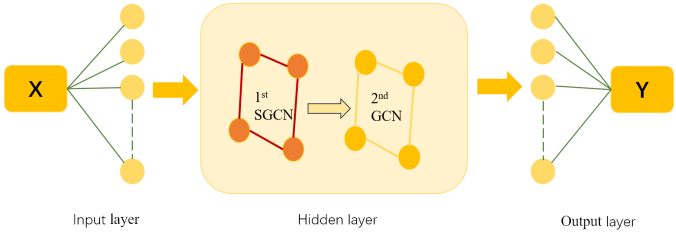

Fig. 1. Schematic diagram of the SGCN-GCN model.

The SGCN-SGCN model is formulated as follows [Eq. (14)]:

$$\begin{aligned} &f_{SGCN-SGCN}(\hat{A}, X; \theta) = \\ &softmax((\lambda\hat{A} + (1-\lambda)\hat{S}^{(1)}) \\ &\mathrm{ReLU}((\lambda\hat{A} + (1-\lambda)\hat{S}^{(0)})XW^{(0)})W^{(1)}) \end{aligned}, \tag{14}$$

where, $\mathrm{ReLU}((\lambda\hat{A} + (1-\lambda)\hat{S}^{(0)})XW^{(0)})$ represents the first layer of the proposed SGCN. $softmax((\lambda\hat{A} + (1-\lambda)\hat{S}^{(1)})X^{(1)}W^{(1)})$ also represents the second layer of the proposed SGCN. $\mathrm{X^{(1)}} = \mathrm{ReLU}((\lambda\hat{A} + (1-\lambda)\hat{S}^{(0)})XW^{(0)})$ represents the graph node feature matrix output by the first convolutional layer's processing. A larger fusion factor λ indicates that structural information has a greater weight in the information propagation between nodes, while the weight

of attribute information is relatively smaller. Like many parameters in deep learning models, W is a parameter matrix that needs to be trained using the gradient descent algorithm. If the model is extended to L layers and the expanded SGCN-SGCN model is used for semi-supervised node classification, the process is shown in Algorithm 1.

Algorithm 1 Expanded SGCN-SGCN for Semi-Supervised Node Classification

Require: Graph $(\mathbf{V}, \mathbf{E}, \mathbf{X})$, Adjacency matrix $\mathbf{A}$, Initial labels $\mathbf{Y}^{(0)}$, Number of SGCN layers $\mathbf{L}$, Fusion factor λ
Ensure: Predicted labels $\mathbf{Y}_U$
Initialize node embeddings $\mathbf{H}^{(0)} = \mathbf{X}$
Initialize transformation matrices $\mathbf{W}^{(0)}$, $\mathbf{W}^{(1)}$, ... ,$\mathbf{W}^{(L-1)}$
Compute $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$,$\mathbf{D}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$
Compute $\hat{\mathbf{A}} = \mathbf{D}^{-1/2}\tilde{\mathbf{A}}\mathbf{D}^{-1/2}$
for $l = 1$ to L **do**
    Calculate similarity matrix $\mathbf{S}^{(l-1)}$ using Equation (12)
    $\hat{\mathbf{S}}^{(l-1)} = \oplus\mathbf{S}^{(l-1)}$
    $\mathbf{Z}^{(l)} = \sigma(\lambda\hat{\mathbf{A}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l-1)} + (1-\lambda)\hat{\mathbf{S}}^{(l-1)})$
    $\mathbf{H}^{(l)} = \mathbf{Z}^{(l)}$
end for
Train the model using labeled nodes $\mathbf{L}$:
    $\mathcal{L}_0 = \sum_{i\in\mathcal{L}} \text{loss}(\mathbf{H}_i^{(L)}, \mathbf{Y}_i^{(0)})$
Predict labels for unlabeled nodes $\mathbf{U}$:
    $\mathbf{Y}_U = \text{softmax}(\mathbf{H}_U^{(L)})$
Return $\mathbf{Y}_U$

This work construct two graph convolutional models for semi-supervised node classification by combining the proposed SGCN with the traditional GCN. In addition to utilizing graph structural information to obtain neighbor aggregation weights, SGCN also takes into account the attribute information of similarity between nodes. The similarity measure constructed using the Minkowski distance differs from cosine similarity in that it more precisely captures the closeness between node features, focusing on the similarity of feature content rather than spatial proximity. It is straightforward to envision that, in the context of semi-supervised node classification, nodes with similar features are more likely to belong to the same category. From this perspective, the rationale behind the approach becomes evident.

D. Analysis of Algorithm Complexity

Assuming L is the number of graph convolutional layers, N is the number of nodes, $\|A\|_0$ is the number of non-zero elements in the adjacency matrix A, and F is the number of node features. The time complexity of training a traditional GCN model for semi-supervised classification is $O(L \parallel A \parallel_0 F+LNF^2)$ and the space complexity is $O(LNF+LF^2)$ [30]. For the L-layer SGCN this study designed, the time complexity required for training is $O(2L \parallel A \parallel_0 F + LNF^2)$, and the space complexity is $O(2LNF + LF^2)$. It is evident that the proposed scheme can learn better node feature representations while maintaining the same order of magnitude of time and space complexity as the classical GCN. Taking the Cora dataset as an example and keeping consistent with the experimental setup detailed in Section 5, this work utilized a two-layer GCN architecture to perform semi-supervised node classification over 300 epochs. The training time for the GCN model was 8.21 seconds, while the proposed enhanced SGCN model took 9.08 seconds. The slight increase in time consumption of the proposed scheme is mainly attributed to the calculation of the similarity matrix.

V. Experiment and Result Analysis

To validate the effectiveness of the proposed SGCN in graph node representation learning, this study leverages the two graph convolutional neural network models outlined in the preceding section to evaluate their performance on semi-supervised node classification tasks. In this section, the experimental datasets, benchmark models, and experimental parameter settings are first introduced. Then, the experimental results of different models are analyzed. Finally, the impact of different metric distances, namely Manhattan distance, Euclidean distance, and Chebyshev distance, as well as varying fusion factors, on the performance of the two constructed semi-supervised node classification models is explored.

A. Datasets and Baseline Methods

This work evaluate the performance of the two proposed models on three benchmark citation network datasets: Cora, Citeseer, and PubMed [31]. These citation network datasets are structured as graphs with papers represented as nodes and citations between papers as edges. The specific details of these datasets are presented in Table I. Taking the Cora dataset as an example, it contains 2708 graph nodes and 5429 edges, with 7 domain categories represented by 7-dimensional one-hot vectors. Each paper is described by a 1433-dimensional one-hot feature vector, where each dimension takes a value of 0 or 1, indicating whether the corresponding word appears in the paper. The label rate in the dataset is 5.2%, meaning that 94.8% of the data is unlabeled.

TABLE I. The Details of Citation Network Datasets

Dataset	Nodes	Edges	Classes	Features	Labeled rate
Cora	2708	5429	7	1433	5.2%
Citeseer	3327	4732	6	3703	3.6%
Pubmed	19717	44338	3	500	0.3%

To validate the effectiveness and advancement of the proposed models for semi-supervised node classification tasks, this study compare them with the following benchmark methods: Multi-Layer Perceptron (MLP) [32], Label Propagation (LP) [33], Semi-supervised Embedding (SemiEmb) [34], Manifold Regularization (ManiReg) [35], Iterative Classification Algorithm (ICA) [36], DeepWalk [37], Planetoid [38], GCN [19], GraphSAGE [11] and GAT [10].

B. Implementation Details

Experimental setup was equipped with an 11th Gen Intel® Core™ i7-1165G7 processor, operating at 2.80 GHz and accompanied by 16GB of RAM. this work developed the SGCN-GCN and SGCN-SGCN models utilizing Python 3.9.10, PyTorch 2.2.1, and torch-geometric 2.5.3, all within the Windows 10 environment. The models were evaluated on three distinct datasets, employing uniform hyperparameter configurations: an L2 regularization factor of 5×10^{-4}, a hidden layer size of 128

units, the Adam optimizer, 300 training epochs, and a learning rate set to 0.01. The reported experimental outcomes reflect the mean classification accuracy across 10 distinct model parameter initializations, ensuring the reliability and robustness of these findings.

C. Experimental Results and Analysis of Different Models

Table II reports the average accuracy of different semi-supervised node classification models, with the highest and second-highest accuracies highlighted in bold. In addition to the experimental results on classification accuracy for MLP, GCN, GraphSAGE, GAT, and the two models proposed in this study, the results for all other models are derived from their original publications. As can be seen from the table, the two proposed models demonstrate superior performance compared to other models on all three datasets. This is attributed to the proposed SGCN, which considers not only the structural information of the graph but also the similarity information between nodes during information aggregation. Unlike the cosine similarity in mathematical space, a linear similarity measure using the Minkowski distance was designed, which better describes the proximity relationship between node features.

TABLE II. CLASSIFICATION ACCURACY (%) OF VARIOUS MODELS

Model	Cora	Citeseer	Pubmed
MLP	57.6	58.9	72.9
ManiReg	59.5	60.1	70.7
SemiEmb	59.0	59.6	71.1
LP	68.0	45.3	63.0
DeepWalk	67.2	43.2	65.3
ICA	75.1	69.1	73.9
Planetoid	75.7	64.7	73.9
GCN	80.6	68.7	78.7
GraphSAGE	80.0	62.4	76.0
GAT	**81.3**	68.5	77.2
SGCN-GCN	**81.5**	**69.5**	**79.5**
SGCN-SGCN	**81.3**	**68.9**	**79.9**

In contrast, other methods have limitations: MLP only uses node attribute information without considering the graph's structural information; ManiReg relies heavily on the data structure within local neighborhoods and can easily overlook global information; SemiEmb has high requirements for labels; DeepWalk cannot model attribute information; Planetoid suffers from information loss in the graph structure during random sampling; GCN, GraphSAGE, and GAT adopt a neighborhood aggregation scheme to improve performance by mixing the features of nodes and their neighbors, but they do not fully utilize the attribute information of the nodes themselves during the aggregation process. The SimP-GCN model proposed in [12] achieves the preservation of node similarity and the adaptive integration of structural and similarity information through the K-Nearest Neighbors (KNN) algorithm, a Node Similarity Preserving Aggregation module, and a Self-Supervised Learning module. In comparison, the scheme proposed in this study is more straightforward and efficient, has a reduced number of parameters, and exhibits stronger generalization capabilities. Since the code provided for constructing the SimP-GCN model in the original text did not utilize the same torch-geometric package as the one used in this study for building graph neural networks, significant errors occurred when attempting to reproduce the results. Consequently, the relevant experimental results for this model are not reported in this work.

The experimental results of the proposed SGCN-GCN and SGCN-SGCN compared to the classical GCN are presented in Fig. 2 below. Specifically, the SGCN-GCN achieves improvements of 0.9%, 0.8%, and 0.8% in accuracy over the traditional GCN on the three datasets, respectively, while SGCN-SGCN achieves improvements of 0.7%, 0.2%, and 1.2%, respectively.

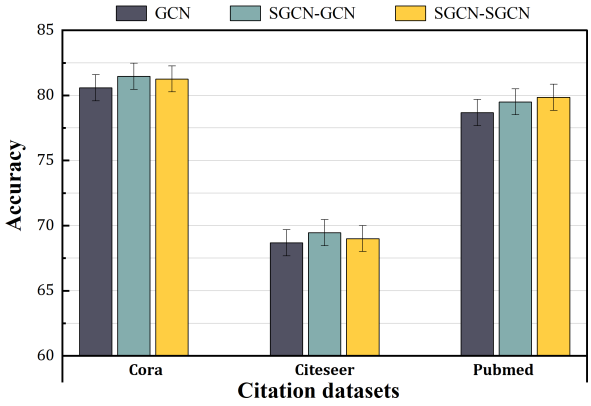

Fig. 2. Comparison chart with classical GCN classification accuracy.

D. Experimental Results and Analysis at Different Distances

This work report the classification accuracy of SGCN-GCN and SGCN-SGCN under different metric distances on the Cora, Citeseer, and Pubmed datasets in Fig. 3 and Fig. 4, respectively. The horizontal axis represents the tested datasets, while the vertical axis represents the classification accuracy, which is the average of ten experimental results.

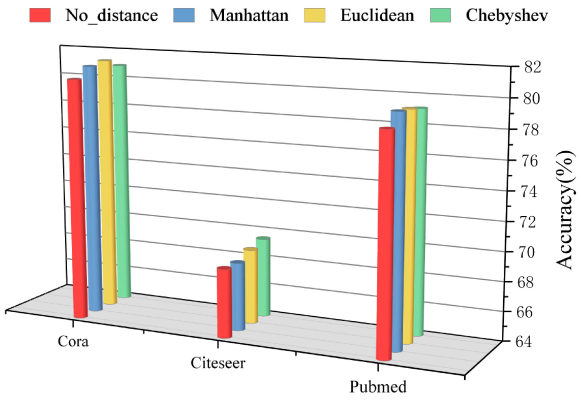

Fig. 3. Comparison of accuracy of SGCN-GCN at different metric distances.

The analysis of the figures reveals that both models perform best using Manhattan distance and Euclidean distance on the Cora and Pubmed datasets, while the Chebyshev distance yields the best results on the Citeseer dataset. This can be primarily attributed to the fact that the node feature dimensions in the Cora and Pubmed networks are relatively low (1433 and

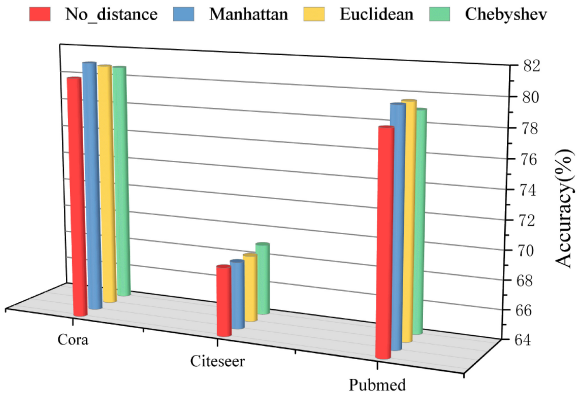

Fig. 4. Comparison of accuracy of SGCN-SGCN at different metric distances.

500, respectively), whereas the Citeseer network has a higher dimension (3703). For nodes with lower-dimensional features, which tend to be denser, using Manhattan and Euclidean distances to calculate similarity can fully utilize each feature value. In contrast, for nodes with higher-dimensional features, which are relatively sparse, using the Chebyshev distance to calculate similarity can better avoid interference from invalid feature values.

E. Experimental Results and Analysis of Different Fusion Factors

This work reports the variation in classification accuracy of SGCN-GCN across various fusion factors on the Cora, Citeseer, and Pubmed datasets, with Fig. 5, Fig. 6, and Fig. 7 depicting the performance under Manhattan distance, Euclidean distance, and Chebyshev distance, respectively. The horizontal axis represents the fusion factor, while the vertical axis indicates the classification accuracy, which is the average of ten experimental results.

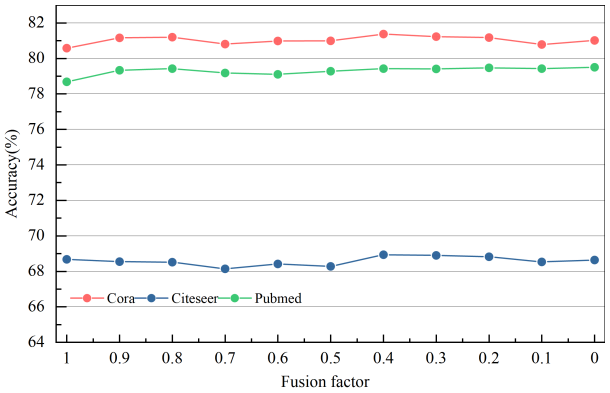

Fig. 5. SGCN-GCN classification accuracy variation map at Manhattan distance.

The analysis of the figures reveals that the SGCN-GCN model achieves better classification performance on the three datasets when the similarity information allocated to the first

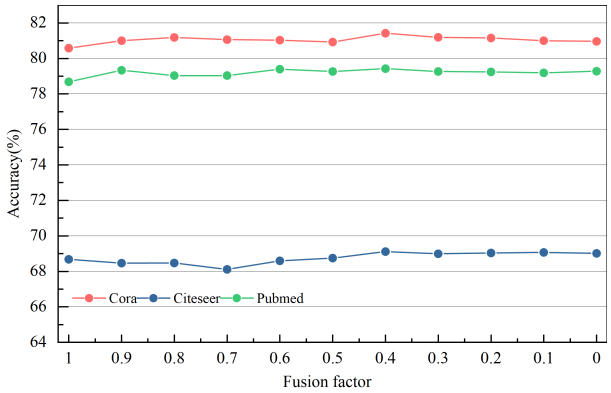

Fig. 6. SGCN-GCN classification accuracy variation map at Euclidean distance.

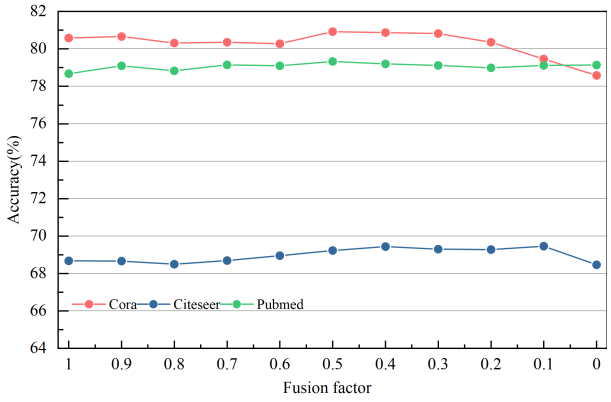

Fig. 7. SGCN-GCN classification accuracy variation map at Chebyshev distance.

layer of the graph convolutional network carries a larger weight. This is because only the first layer of the network utilizes similarity information for aggregation. Specifically, the model performs well when the fusion factor λ ranges from 0.1 to 0.5. When $\lambda = 1$ and $\lambda = 0$, the model solely relies on graph structural information (original GCN) and similarity information, respectively, for information aggregation. Notably, this study observes that the model using similarity information computed by Manhattan and Euclidean distances achieves better classification performance than the classical GCN when $\lambda = 0$ (i.e. using only similarity information). Irrespective of the value of the fusion factor λ, the model that integrates similarity information computed by Manhattan and Euclidean distances consistently demonstrates excellent classification results on the Cora and PubMed datasets. This verifies the validity of the approach to integrate similarity information into the original GCN framework.

This work reports the variation in classification accuracy of SGCN-SGCN across various fusion factors on the Cora, Citeseer, and Pubmed datasets, with Fig. 8, Fig. 9, and Fig. 10 depicting the performance under Manhattan distance, Euclidean distance, and Chebyshev distance, respectively. The horizontal axis represents the fusion factor, while the vertical axis indicates the classification accuracy, which is the average of ten experimental results.

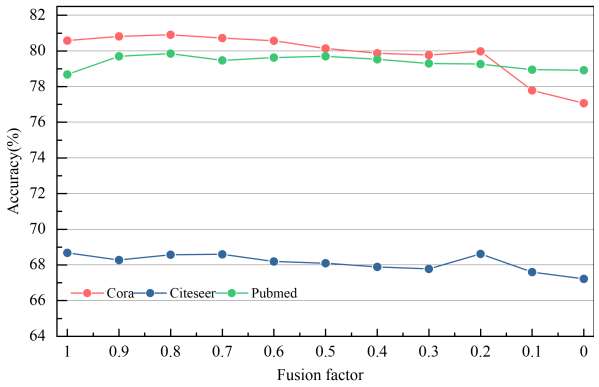

Fig. 8. SGCN-SGCN classification accuracy variation map at Manhattan distance.

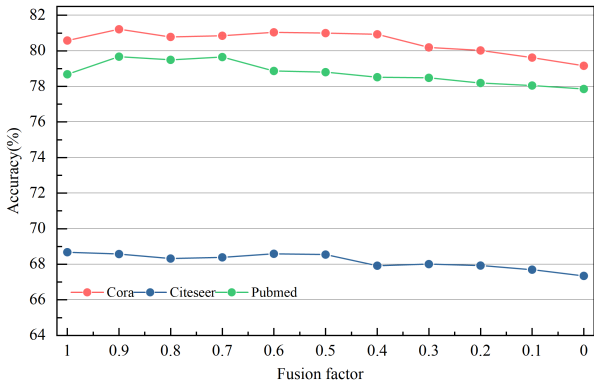

Fig. 9. SGCN-SGCN classification accuracy variation map at Euclidean distance.

The analysis of the figures indicates that the SGCN-SGCN model, which leverages similarity information for information aggregation in both layers of the network, achieves superior classification performance on the Cora and Pubmed datasets when each layer of the graph convolutional network is assigned a lower weight to the similarity information. Specifically, the model performs well when the fusion factor λ is greater than or equal to 0.6. When $\lambda = 1$ and $\lambda = 0$, the model relies solely on graph structural information (original GCN) and similarity information, respectively, for information aggregation. However, the model's performance on the Citeseer dataset is suboptimal, likely due to the sparsity of the feature vectors caused by its higher node feature dimension (3703).

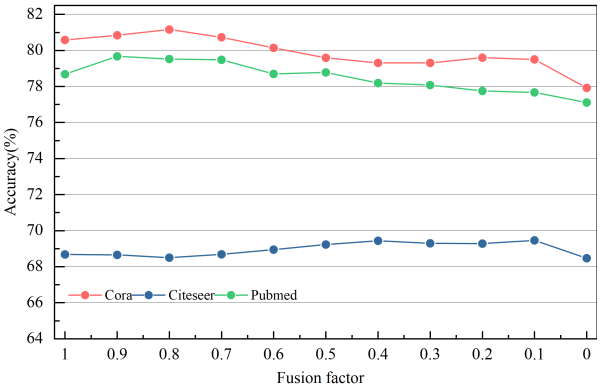

Fig. 10. SGCN-SGCN classification accuracy variation map at Chebyshev distance.

VI. Discussion

This study introduces SGCN, a novel graph convolutional network that integrates a linear similarity measure based on Minkowski distance with traditional graph structural information, addressing the limitations of conventional GCNs. Traditional GCNs typically overlook node attribute information and tend to disrupt the similarity information between nodes during information aggregation. Methods like GraphSAGE and GAT improve neighborhood aggregation but fail to fully capture node feature similarity and require the addition of supervised modules with more parameters and black-box characteristics.

Experimental results demonstrate that SGCN performs best when similarity information is weighted more heavily in the first layer, with the fusion factor between 0.1 and 0.5. This indicates that, while structural information plays a dominant role in the information propagation and aggregation of graph nodes, attribute information also provides a complementary, supportive function. Among the distance metrics tested, Manhattan and Euclidean distances consistently yield better results, suggesting they are more effective at capturing feature proximity in node classification tasks.

Future work may focus on extending SGCN to more specific graph data, such as functional brain networks, and implementing adaptive fusion factor adjustments to learn better node embedding representations, aiming to improve the recognition accuracy of brain diseases such as Alzheimer's and autism.

VII. Conclusion

Traditional Graph Convolutional Networks (GCNs) primarily rely on graph structure for node aggregation, often overlooking node attribute information. This approach can reduce node similarity and impede effective node feature learning, particularly in semi-supervised classification tasks. To address these issues, this study proposes SGCN, a novel Graph Convolutional Network. SGCN introduces a linear similarity measure based on Minkowski distance between node features, enhancing the description of similarity. By integrating this measure with traditional graph structure, SGCN improves node feature representation. Experimental validation on datasets such as Cora, Citeseer, and PubMed demonstrates that SGCN models (SGCN-GCN and SGCN-SGCN) significantly outperform traditional GCNs in node feature learning. The impact of metric distances and fusion factors on performance is also analyzed, providing insights for model optimization.

Acknowledgment

This project has received funding support from the Shaanxi Provincial Department of Science and Technology and the Xi'an Science and Technology Bureau, with project numbers 2024GX-YBXM-113 and 23GXFW0004, respectively.

REFERENCES

[1] X. Zhao, L. Wang, Y. Zhang, X. Han, M. Deveci, and M. Parmar, "A review of convolutional neural networks in computer vision," *Artificial Intelligence Review*, vol. 57, no. 4, p. 99, 2024.

[2] W. Ju, Z. Fang, Y. Gu, Z. Liu, Q. Long, Z. Qiao, Y. Qin, J. Shen, F. Sun, Z. Xiao *et al.*, "A comprehensive survey on deep graph representation learning," *Neural Networks*, vol. 173, p. 106207, 2024.

[3] Z. Chen, Y. Zhang, Y. Fang, Y. Geng, L. Guo, X. Chen, Q. Li, W. Zhang, J. Chen, Y. Zhu *et al.*, "Knowledge graphs meet multi-modal learning: A comprehensive survey," *arXiv preprint arXiv:2402.05391*, 2024.

[4] K. Zheng, S. Yu, and B. Chen, "Ci-gnn: A granger causality-inspired graph neural network for interpretable brain network-based psychiatric diagnosis," *Neural Networks*, vol. 172, p. 106147, 2024.

[5] W. Ju, Z. Liu, Y. Qin, B. Feng, C. Wang, Z. Guo, X. Luo, and M. Zhang, "Few-shot molecular property prediction via hierarchically structured learning on relation graphs," *Neural Networks*, vol. 163, pp. 122–131, 2023.

[6] Z. Song, X. Yang, Z. Xu, and I. King, "Graph-based semi-supervised learning: A comprehensive review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 11, pp. 8174–8194, 2022.

[7] Z. Kang, C. Peng, Q. Cheng, X. Liu, X. Peng, Z. Xu, and L. Tian, "Structured graph learning for clustering and semi-supervised classification," *Pattern Recognition*, vol. 110, p. 107627, 2021.

[8] W. Guo, B. Xue, Y. Hou, and Y. Hu, "Semi-supervised classification based on improved graph convolutional networks," *Journal of Shaanxi University of Science and Technology*, vol. 42, no. 05, pp. 191–197, 2024.

[9] Y. Zhang, Y. Zhang, D. Yan, Q. He, and Y. Yang, "Nie-gcn: Neighbor item embedding-aware graph convolutional network for recommendation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, no. 5, pp. 2810–2821, 2024.

[10] C. Ding, S. Sun, and J. Zhao, "Mst-gat: A multimodal spatial–temporal graph attention network for time series anomaly detection," *Information Fusion*, vol. 89, pp. 527–536, 2023.

[11] T. Liu, A. Jiang, J. Zhou, M. Li, and H. K. Kwan, "Graphsage-based dynamic spatial–temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 10, pp. 11 210–11 224, 2023.

[12] W. Jin, T. Derr, Y. Wang, Y. Ma, Z. Liu, and J. Tang, "Node similarity preserving graph convolutional networks," in *Proceedings of the 14th ACM international conference on web search and data mining*, 2021, pp. 148–156.

[13] J. Yin and S. Sun, "Incomplete multi-view clustering with cosine similarity," *Pattern Recognition*, vol. 123, p. 108371, 2022.

[14] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, H. Luo, and D. Tao, "A survey on self-supervised learning: Algorithms, applications, and future trends," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2024.

[15] K. Janani, S. Mohanrasu, A. Kashkynbayev, and R. Rakkiyappan, "Minkowski distance measure in fuzzy promethee for ensemble feature selection," *Mathematics and Computers in Simulation*, vol. 222, pp. 264–295, 2024.

[16] Z. Wu, X. Lin, Z. Lin, Z. Chen, Y. Bai, and S. Wang, "Interpretable graph convolutional network for multi-view semi-supervised learning," *IEEE Transactions on Multimedia*, vol. 25, pp. 8593–8606, 2023.

[17] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[18] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, vol. 29, 2016.

[19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[20] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*, 2018.

[21] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International conference on machine learning*. PMLR, 2017, pp. 1263–1272.

[22] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner *et al.*, "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[23] J. Chen, T. Ma, and C. Xiao, "Fastgcn: fast learning with graph convolutional networks via importance sampling," *arXiv preprint arXiv:1801.10247*, 2018.

[24] D. Zou, Z. Hu, Y. Wang, S. Jiang, Y. Sun, and Q. Gu, "Layer-dependent importance sampling for training deep and large graph convolutional networks," *Advances in neural information processing systems*, vol. 32, 2019.

[25] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "Gaan: Gated attention networks for learning on large and spatiotemporal graphs," *arXiv preprint arXiv:1803.07294*, 2018.

[26] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[27] R. Zhang, Y. Zou, and J. Ma, "Hyper-sagnn: a self-attention based graph neural network for hypergraphs," *arXiv preprint arXiv:1911.02613*, 2019.

[28] W. Ju, Z. Fang, Y. Gu, Z. Liu, Q. Long, Z. Qiao, Y. Qin, J. Shen, F. Sun, Z. Xiao *et al.*, "A comprehensive survey on deep graph representation learning," *Neural Networks*, vol. 173, p. 106207, 2024.

[29] E. Gwynne and J. Sung, "The minkowski content measure for the liouville quantum gravity metric," *The Annals of Probability*, vol. 52, no. 2, pp. 658–712, 2024.

[30] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 257–266.

[31] L. Zhang, R. Song, W. Tan, L. Ma, and W. Zhang, "Igcn: A provably informative gcn embedding for semi-supervised learning with extremely limited labels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–14, 2024.

[32] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems*, vol. 34, pp. 24 261–24 272, 2021.

[33] I. B. El Kouni, W. Karoui, and L. B. Romdhane, "Node importance based label propagation algorithm for overlapping community detection in networks," *Expert Systems with Applications*, vol. 162, p. 113020, 2020.

[34] Y. Liu, Q. Wang, X. Wang, F. Zhang, L. Geng, J. Wu, and Z. Xiao, "Community enhanced graph convolutional networks," *Pattern Recognition Letters*, vol. 138, pp. 462–468, 2020.

[35] W. Liu, S. Fu, Y. Zhou, Z.-J. Zha, and L. Nie, "Human activity recognition by manifold regularization based dynamic graph convolutional networks," *Neurocomputing*, vol. 444, pp. 217–225, 2021.

[36] A. Tharwat, "Independent component analysis: An introduction," *Applied Computing and Informatics*, vol. 17, no. 2, pp. 222–249, 2021.

[37] J. Guo, H. Wen, W. Huang, and C. Yang, "A collaborative filtering recommendation algorithm based on deepwalk and self-attention," *International Journal of Computational Science and Engineering*, vol. 26, no. 3, pp. 296–304, 2023.

[38] S. Ershkov, D. Leshchenko, and A. Rachinskaya, "Dynamics of a small planetoid in newtonian gravity field of lagrangian configuration of three primaries," *Archive of Applied Mechanics*, vol. 93, no. 10, pp. 4031–4040, 2023.