Deep Multi-modal Graph Clustering via Graph Transformer Network

Qianqian Wang[1,3], **Haiming Xu**[1*], **Zihao Zhang**[1], **Wei Feng**[2], **Quanxue Gao**[1]

[1]School of Telecommunications Engineering, Xidian University, Xi'an, China
[2]School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China
[3] Anhui Provincial Key Laboratory of Multimodal Cognitive Computation, Anhui University
qqwang@xidian.edu.cn, 24011211044@stu.xidian.edu.cn, 22011211091@stu.xidian.edu.cn,
weifeng.ft@xjtu.edu.cn, qxgao@xidian.edu.cn

Abstract

Current deep multi-modal graph clustering methods primarily rely on Graph Neural Network (GNN) to fully exploit attribute features and graph structures, including message propagation and low-dimensional feature embedding. However, these methods lack further exploration of graph structural information, such as the relationship between nodes and shortest paths. Additionally, they may not sufficiently mine complementary information among multi-modal graph data. To address these issues, we propose a novel **D**eep **M**ulti-modal **G**raph **C**lustering via **G**raph **T**ransformer **N**etwork method, called DMGC-GTN. This method thoroughly dissects and utilizes graph structural information, applying graph smoothing to node features and incorporating various forms of embeddings into the transformer architecture. This achieves a unified embedding of graph structure and multi-modal feature attributes, fully exploiting the complementary information within multi-modal graph data. Extensive experiments demonstrate the effectiveness of our algorithm.

Introduction

Clustering is a prominent machine learning technique employed to categorize similar data samples into distinct groups based on their intrinsic features. As an unsupervised learning method, it significantly reduces the costs associated with data annotation (MacQueen et al. 1967; Johnson 1967; Gong et al. 2022) , thereby establishing itself as a key focus in contemporary research. Traditional clustering methods, such as (Xu et al. 2021; Xie et al. 2020), demonstrate efficacy in clustering Euclidean-structured data through deep learning and co-training strategies. However, their performance diminishes when addressing non-Euclidean structured data, such as social networks and citation networks. Due to the complex topological structures of these data, it is necessary to study more advanced techniques.

Graph Neural Networks (GNNs) (Scarselli et al. 2008), as a typical approach for non-Euclidean space, are particularly well-suited for handling graph-structured data. It obtains node embeddings by performing convolution operations on graphs to capture both the local structural features and abundant node attribute information. This capability enables GNNs to achieve promising clustering performance on non-Euclidean structured data, which inspires numerous GNN variants developed (Xu et al. 2018; Hamilton, Ying, and Leskovec 2017; Ying 2018). For example, the Graph Autoencoder (GAE) (Kipf and Welling 2016b) combines an autoencoder framework with Graph Convolutional Networks (GCNs) (Kipf and Welling 2016a) to capture both local structural features and intricate node associations. However, the aforementioned methods adopt global shared weights for neighbor information of a node, resulting in poor generalization ability. To overcome the weakness, Graph Attention Autoencoder (GATE) (Wang et al. 2019b) introduces an attention mechanism that assigns varying weights to neighbors to capture their contribution more accurately.

Despite their advancements, these methods are constrained to single-modal analysis, focusing on node attributes and graph construction from a single perspective, which results in inaccurate data description (Wang et al. 2022; Yu et al. 2024). To tackle this problem, several multi-modal graph clustering methods are proposed, which can integrate multiple perspectives and provide a more comprehensive understanding of node relationships. For instance, Cheng et al. (2021) developed the Multi-Modal Attribute Graph Convolutional Network (MAGCN) by employing a dual-path encoder to learn both graph embedding features and modality-consistent information. Similarly, Li et al. (2020) introduced Co-GCN that enhances model representation through the adaptive weighting of the Laplace matrix. However, these methods focus more on modality-consistent representation, which leads to the loss of modality-specific features and may not fully exploit the complementary information inherent across modalities. Additionally, they do not exploit the more complex graph structural information, such as shortest paths and node distances, leading to insufficient prior information extracted from the graph.

To mitigate these challenges, we propose a novel Deep Multi-modal Graph Clustering method via Graph Transformer Network, named DMGC-GTN, which effectively explores complementary information while ensuring consistency. Our model integrates graph smoothing over nodes and multi-dimensional embedding extraction to obtain thorough prior graph information. Innovatively, we combine multi-modal graph embeddings with the attention mechanism in

*Corresponding author

the Transformer. This approach captures the complex relationships between multi-modal nodes and edges. The main contributions of our work are summarized as follows:

- We introduce a novel deep multi-modal graph clustering method, which constructs multi-modal graph transformer network to adaptively integrate features from diverse modalities. Therefore, it effectively maintains the inherent connectivity of the graph structure.
- Our method introduces more complex graph structural information, such as shortest paths and node distances, making it capable of extracting more abundant features from the multi-modal graph.
- Extensive experiments on multiple multi-modal graph datasets demonstrate its superior performance.

Related Work

Multi-Modal Graph Clustering

Multi-modal graph clustering leverages diverse perspectives to enhance graph clustering accuracy and robustness. For instance, Li et al. (2015) proposed a bipartite graph-based approach to combine heterogeneous features, thus increasing the efficiency of spectral analysis. In contrast, Nie et al. (2017) introduced a self-weighted approach that dynamically learns the importance of each modality through Laplace rank constraints, thus eliminating the need for additional hyperparameters. Building on these foundations, Zhu et al. (2018) integrated the affinity matrix and clustering into one step to eliminate the noise and redundancy of two-step strategies. Wang et al. (2019) further developed graph-based MVC (GMC) that enables mutual reinforcement between the modality-specific graph and the unified graph and achieves one-step clustering via a rank constraint. Nonetheless, these methods often rely on hand-crafted features and might not capture complex patterns and structures, particularly when processing complex graph data.

Deep Multi-Modal Graph Clustering

Deep multi-modal graph clustering captures nonlinear data features and reveals underlying node connections. This is particularly significant for graph-based clustering, where the rich structural and attribute information inherent in graphs plays a crucial role. For example, Cheng et al. (2021) designed MAGCN with a two-pathway encoder structure, effectively capturing graph embedding features while learning modality-consistency information. Xia et al. (2021) introduced the Self-Supervised Graph Convolutional Network for Multi-Modal Clustering (SGCMC). Their method improves clustering performance by employing Euler transformations to enhance node attributes and using $\ell_{1,2}$-paradigm-constrained block diagonal representations. This self-supervised approach effectively refines node representations. Similarly, Fan et al. (2020) proposed the One2Multi Graph Autoencoder for Multi-Modal Graph Clustering (O2MAC), which learns node embeddings from a single comprehensive graph modality and reconstructs multiple modalitys. This method enhances clustering outcomes by capturing various graph perspectives. Despite these advances, challenges remain in fully utilizing complex graph structures. For example, existing methods often struggle to capture the intricate path and distance relationships in the graph. Our approach addresses these limitations by integrating node smoothing in the graph structure and extracting multi-dimensional embeddings from each graph dimension.

Symbol	Description
Graph Structures	
$\mathbf{G} = (\mathbf{V}, \mathbf{E})$	Graph with vertex set $\mathbf{V}$ and edge set $\mathbf{E}$
$\mathbf{A}, \mathbf{D}, \mathbf{L}$	Adjacency/degree/Laplacian matrix
$\hat{\mathbf{A}}, \hat{\mathbf{D}}, \hat{\mathbf{L}}$	Augmented adjacency/degree/Laplacian matrix
$\bar{\mathbf{A}}$	Row-normalized adjacency matrix
$\mathbf{S}$	PageRank-based intimacy matrix
$\tilde{\mathbf{S}}$	Reconstructed similarity matrix
Node Features and Embeddings	
$\boldsymbol{\mathcal{X}}$	Multi-modal node attribute tensor
$\mathbf{X}^m$	Node attribute matrix of m-th modality
$\hat{\mathbf{X}}^m$	Smoothed feature matrix of m-th modality
$\tilde{\mathbf{X}}$	Reconstructed feature matrix
$\boldsymbol{\mathcal{S}}_i$	Top-t intimacy sequence for node i
$\boldsymbol{\mathcal{Y}}_i^m$	Integrated node embeddings
$\mathbf{Y}_i^m$	Transformer input representation
$\mathbf{Z}^m$	Node representations of m-th modality
$\mathbf{Z}$	Final fused node representations
Clustering Components	
$\mathbf{C} = \{\mathbf{c}_j\}$	Set of cluster centers
$\mathbf{Q}, \mathbf{P}$	Predicted and target cluster distributions

Table 1: Main Notations Used in This Paper

Deep Multi-modal Graph Clustering via Graph Transformer Network

This section begins by defining the symbols and clarifying the problem statement. It then introduces the model components, including graph Laplacian smoothing filtering, graph decomposition embedding, and the design of loss functions

Problem Formalization

Consider a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ represents *Vertices* and $\mathbf{E}$ represents *Edges*, and the topological structure of the graph $\mathbf{G}$ is typically represented by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. $\mathbf{D} \in \mathbb{R}^{n \times n}$ denotes the degree matrix corresponding to $\mathbf{A}$, and the Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Owing to the nodes in graph $\mathbf{G}$ exhibiting feature attributes from different modalities, these attributes collectively form a multi-modal node attribute tensor, denoted as $\boldsymbol{\mathcal{X}} = \{\mathbf{X}^m \in \mathbb{R}^{n \times d_m}\}_{m=1}^M$, where $\mathbf{X}^m$ represents the node attribute matrix for the m-th modality with n nodes and d_m features, and $\mathbf{x}_i^m$ represents $\mathbf{X}^m$'s i-th row. Through the propagation and aggregation of information between multi-modal data $\boldsymbol{\mathcal{X}}$ and graph structure $\mathbf{A}$, the model aims to obtain a low-dimensional latent embedding $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\} \in \mathbb{R}^{n \times d}$ suitable for clustering tasks that partition the nodes into k clusters with cluster centers $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k\}$.

Data preprocessing

Graph Laplacian Smoothing The smoothness of a graph signal implies that adjacent nodes within the graph display similar feature representations, often stemming from the

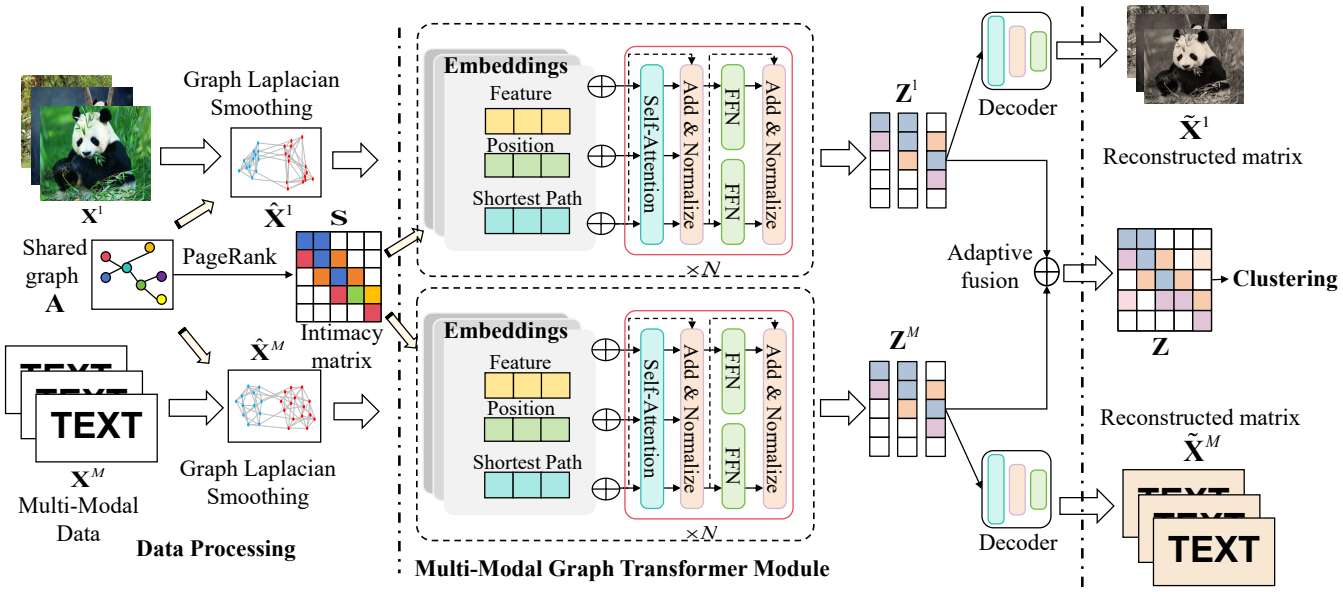


Figure 1: This model begins with graph smoothing on the input feature matrix $\mathbf{X}^m$ to yield the smoothed matrix $\hat{\mathbf{X}}^m$. It utilizes three types of embeddings: Feature Embeddings for node feature learning, Position Embeddings for positional information, and Shortest Path Embeddings for capturing distances and local structures. These embeddings are combined to form an integrated feature representation. This representation is then processed through several Transformer layers, which use self-attention mechanisms to refine node representations. The final feature matrix is normalized to obtain low-dimensional embeddings $\mathbf{Z}^m$, which are used for clustering to distinguish different node groups.

message-passing process of GNNs. This feature propagation process can be expressed as:

$$\hat{\mathbf{X}} = \mathbf{H}\mathbf{X}, \tag{1}$$

where $\hat{\mathbf{X}}$ denotes the filtered attribute feature matrix, $\mathbf{H}$ is the filter, which is computed as:

$$\begin{aligned}\mathbf{H} &= \hat{\mathbf{D}}^{-\frac{1}{2}}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-\frac{1}{2}} \\ &= \hat{\mathbf{D}}^{-\frac{1}{2}}(\hat{\mathbf{D}} - g\mathbf{L})\hat{\mathbf{D}}^{-\frac{1}{2}} \\ &= \mathbf{I} - g\hat{\mathbf{L}}_s,\end{aligned} \tag{2}$$

where $\mathbf{I}$ denotes the identity matrix, and $\hat{\mathbf{D}}$ denotes the degree matrix derived from the adjacency matrix $\hat{\mathbf{A}}$ with $\hat{\mathbf{A}} = \mathbf{I} + \mathbf{A}$. $\hat{\mathbf{L}}_s$ signifies the Laplacian matrix after symmetric normalization. Notice that g influences the efficacy of the smoothing filter, with $g = 1$ corresponding to classic Graph Convolutional Networks.

The expansion of the neighborhood range of nodes with increasing order will result in more pronounced smoothing effects. To capture neighborhood relationships within a certain distance, it is recommended to set up k-th stacking and consider the node properties of each modality, namely

$$\hat{\mathbf{X}}^m = \mathbf{H}^k\mathbf{X}^m. \tag{3}$$

Intimacy Inference The intimacy between nodes can be effectively inferred from the graph structure. Various methods exist to quantify this intimacy, among which the PageRank algorithm serves as a notable approach, as defined below:

$$\mathbf{S} = (1-\alpha)\cdot\left(\mathbf{I} - \alpha\cdot\bar{\mathbf{A}}\right)^{-1}, \tag{4}$$

where α is the damping factor, typically set to 0.85. $\bar{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$ denotes the row-normalized adjacency matrix. By leveraging random walks, the matrix $\mathbf{S}$ encodes the intimate relationships between nodes within the graph.

Graph-guided Node Embedding

Through the message-passing mechanism, graph structures can smooth the node attribute matrix. However, certain aspects of the information on the graph structure, such as the intimacy and relative distances between nodes, remain underutilized.

Feature Embeddings Given a node $\mathbf{X}_i^m$, we select the top t nodes with the closest intimacy from $\mathbf{S}$ and construct a sequence $\boldsymbol{\mathcal{S}}_i \in \mathbb{R}^{1\times(t+1)\times d_m}$. Consequently, the sequence with the central node x_i^m is encoded to yield the following embedding:

$$\boldsymbol{\mathcal{F}}_i^m = f^m(\boldsymbol{\mathcal{S}}_i, \phi) \in \mathbb{R}^{1\times(t+1)\times d}, \tag{5}$$

where $f^m(\cdot,\phi)$ represents the function mapping to embeddings of m-th modality, and ϕ represents learnable parameters of the mapping function.

Position Embeddings The central node $\mathbf{x}_i^m$ retains relative position information with respect to the internal nodes $\mathbf{x}_j^m$ within the sequence $\boldsymbol{\mathcal{S}}_i$. To capture this relationship, we define a relative position function $\mathrm{Pos}(\mathbf{x}_j^m)$, which assigns position values based on the intimacy between the central node $\mathbf{x}_i^m$ and each internal node $\mathbf{x}_j^m$. Specifically, the function assigns the central node $\mathbf{x}_i^m$ a position value of 0, i.e.,

$\mathrm{Pos}(\mathbf{x}_i^m) = 0$, and ensures that nodes with higher intimacy values relative to $\mathbf{x}_i^m$ are assigned position values closer to 0. This design reflects the closeness of nodes in the graph structure. The embedding of the relative position is then computed as:

$$\boldsymbol{\mathcal{G}}_i^m = f^m(\mathrm{Pos}(\boldsymbol{\mathcal{S}}_i), \epsilon) \in \mathbb{R}^{1\times(t+1)\times d}, \tag{6}$$

where $f^m(\cdot, \epsilon)$ maps the relative position sequence to embeddings within the desired representation space, and ϵ represents learnable parameters of the mapping function.

Shortest Path Embeddings In addition to the intimacy between nodes, the shortest paths between every pair of nodes on the graph provide valuable structural information. Specifically, the shortest path captures the most direct connection between two nodes in the graph, which can enhance the representation of their relationships. For nodes $\mathbf{x}_i^m$ and $\mathbf{x}_j^m$, the shortest path between them is formally defined as $\mathrm{Sp}(\mathbf{x}_i^m, \mathbf{x}_j^m)$, representing the shortest path distance between $\mathbf{x}_i^m$ and $\mathbf{x}_j^m$. To leverage this information, we compute a shortest path embedding as:

$$\boldsymbol{\mathcal{H}}_i^m = f^m(\mathrm{Sp}(\boldsymbol{\mathcal{S}}_i), \gamma) \in \mathbb{R}^{1\times(t+1)\times d}, \tag{7}$$

where $f^m(\cdot, \gamma)$ is a learnable mapping function that encodes the shortest path relationships into the desired embedding space, with γ representing trainable parameters.

Through the aforementioned graph smoothing filters and embedding techniques, we further exploit the graph structure information. For each node $\mathbf{x}_i^m$, we integrate the embeddings as follows:

$$\boldsymbol{\mathcal{Y}}_i^m = \boldsymbol{\mathcal{F}}_i^m + \boldsymbol{\mathcal{G}}_i^m + \boldsymbol{\mathcal{H}}_i^m \in \mathbb{R}^{1\times(t+1)\times d}. \tag{8}$$

Multi-Modal Graph Transformer Module

Transformer Encoder The encoder adopts a standard Transformer architecture. To describe its operation, a two-dimensional slice is extracted from the embedding tensor $\boldsymbol{\mathcal{Y}}_i^m$ as:

$$\mathbf{Y}_i^m = \boldsymbol{\mathcal{Y}}_i^m[0, :, :] \in \mathbb{R}^{(t+1)\times d}, \tag{9}$$

where $\mathbf{Y}_i^m$ represents the embedding for node $\mathbf{x}_i^m$, obtained by slicing along the first dimension of $\boldsymbol{\mathcal{Y}}_i^m$.

This slice $\mathbf{Y}_i^m$ is linearly projected into query, key, and value matrices, denoted as $\mathbf{U}_i^m$, $\mathbf{K}_i^m$, and $\mathbf{V}_i^m$. The attention mechanism is computed as:

$$\mathrm{Attention}(\mathbf{U}_i^m, \mathbf{K}_i^m, \mathbf{V}_i^m) = \mathrm{softmax}\left(\frac{\mathbf{U}_i^m(\mathbf{K}_i^m)^\top}{\sqrt{d_K}}\right)\mathbf{V}_i^m, \tag{10}$$

where the scaling factor $\frac{1}{\sqrt{d_K}}$ ensures numerical stability and softmax function normalizes these values into a probability distribution.

Through L stacked layers, each consisting of multi-head attention, feed-forward networks (FFN), and residual connections, the encoder processes $\mathbf{Y}_i^m$, producing the output feature for node i, denoted as $\mathbf{Z}_i^m \in \mathbb{R}^{(t+1)\times d}$. The features from all n nodes are concatenated to form the aggregated tensor:

$$\boldsymbol{\mathcal{Z}}^m = [\mathbf{Z}_1^m; \mathbf{Z}_2^m; \cdots; \mathbf{Z}_n^m] \in \mathbb{R}^{n\times(t+1)\times d}. \tag{11}$$

To prepare for clustering, average pooling is applied over the temporal dimension to obtain the final node representations:

$$\mathbf{Z}^m = \frac{1}{t+1}\sum_{t=1}^{t} \boldsymbol{\mathcal{Z}}^m[:, t, :] \in \mathbb{R}^{n\times d}. \tag{12}$$

For M different modalities, we fuse their features through weighted summation:

$$\mathbf{Z} = \sum_{m=1}^{M} \alpha_m \mathbf{Z}^m, \tag{13}$$

where the weights α_m are adaptively learned to fuse the modality features into the final representation $\mathbf{Z} \in \mathbb{R}^{n\times d}$.

Decoder The decoder is responsible for reconstructing both the input feature matrix and the graph structure from the embeddings, ensuring that the learned representations effectively capture both local and global information. To achieve this, we design two complementary reconstruction tasks:

1. *Feature Reconstruction*: The decoder uses a multilayer perception (MLP) to reconstruct the feature matrix $\tilde{\mathbf{X}}$ from the embeddings, minimizing the reconstruction error via the mean squared error (MSE) loss:

$$\mathcal{L}_r = \sum_{m=1}^{M} \left\| \hat{\mathbf{X}}^m - \tilde{\mathbf{X}}^m \right\|_F^2, \tag{14}$$

where $\hat{\mathbf{X}}^m$ represents the smoothed feature matrix of the m-th modality obtained through Graph Laplacian Smoothing, and $\tilde{\mathbf{X}}^m$ is the corresponding reconstructed feature matrix.

2. *Graph Structure Reconstruction*: To retain the graph's structural information, the decoder aligns the predicted similarity matrix $\tilde{\mathbf{S}}$, derived from embeddings $\mathbf{Z}$ using cosine similarity, with the intimacy matrix $\mathbf{S}$ by minimizing the MSE loss:

$$\mathcal{L}_g = \left\| \mathbf{S} - \tilde{\mathbf{S}} \right\|_F^2, \tag{15}$$

where $\mathbf{S}$ is the intimacy matrix computed by PageRank, and $\tilde{\mathbf{S}}_{ij} = \cos(\mathbf{z}_i, \mathbf{z}_j)$, and $\mathbf{z}_i$ is the i-th row of $\mathbf{Z}$.

Clustering Module

An ideal cluster distribution should ensure that samples within the same cluster are closer, while those in different clusters are farther apart. To achieve this goal, we construct an ideal target distribution $\mathbf{P}$ to guide the current cluster distribution $\mathbf{Q}$, thereby enhancing the clustering performance.

The predicted cluster distribution $\mathbf{Q}$, representing the likelihood of node i belonging to cluster j, is defined as:

$$\mathbf{q}_{ij} = \frac{\left(1 + \|\mathbf{z}_i - \mathbf{c}_j\|^2/\beta\right)^{-\frac{\beta+1}{2}}}{\sum_{j'}\left(1 + \|\mathbf{z}_i - \mathbf{c}_{j'}\|^2/\beta\right)^{-\frac{\beta+1}{2}}}, \tag{16}$$

where $\mathbf{z}_i$ is the i-th row of $\mathbf{Z}$, $\mathbf{c}_j$ is the centroid of cluster j, $\mathbf{q}_{ij}$ is the i-th row and the j-th column of $\mathbf{Q}$ and $\beta > 0$ controls the softness of the cluster assignments.

To refine the cluster assignments, the target distribution $\mathbf{P}$ is computed by normalizing $\mathbf{Q}$ with respect to cluster confidence and frequency. This is formulated as:

$$\mathbf{p}_{ij} = \frac{\mathbf{q}_{ij}^2 / f_j}{\sum_{j'} \mathbf{q}_{ij'}^2 / f_{j'}}, \tag{17}$$

where $\mathbf{p}_{ij}$ is the i-th row and the j-th column of $\mathbf{P}$, f_j denotes the frequency of cluster j. Specifically, the target distribution $\mathbf{P}$ is designed to emphasize high confidence assignments, making the separation between clusters more distinct.

The clustering loss, defined as the Kullback-Leibler (KL) divergence between $\mathbf{P}$ and $\mathbf{Q}$, is expressed as:

$$\mathcal{L}_{pq} = \sum_i \sum_j \mathbf{p}_{ij} \log \frac{\mathbf{p}_{ij}}{\mathbf{q}_{ij}}. \tag{18}$$

By minimizing $\mathcal{L}_{pq}$, the predicted cluster distribution $\mathbf{Q}$ is progressively aligned with the ideal target distribution $\mathbf{P}$, leading to improved cluster compactness and separation.

Joint Optimization

The total loss function of the proposed DMGC-GTN is eventually formulated as:

$$\mathcal{L} = \mathcal{L}_r + \lambda_1 \mathcal{L}_g + \lambda_2 \mathcal{L}_{pq}, \tag{19}$$

where λ_1 and λ_2 are hyperparameters. The optimization is two-staged: first, $\mathcal{L}_r$ and $\mathcal{L}_g$ are optimized for robust feature extraction; second, $\mathcal{L}_{pq}$ refines these features for improved clustering performance.

Experimental Analysis

Experimental Setting

Metrics and Databases: We evaluated our approach on four datasets: AMAP, WIKI, Citeseer, and Cora. These datasets vary in size and structure, with Cora and Citeseer containing citation networks, WIKI featuring co-occurrence relationships, and AMAP representing product networks. To enrich graph representation, we generated an additional attribute modality using the Fast Fourier Transform (FFT). Table 2 summarizes the dataset statistics.

Database	Dimension	Classes	Nodes	Edges
AMAP (Liu et al. 2022)	745	8	7650	119081
WIKI (Yang et al. 2015)	4973	17	2,405	8261
CITESEER (Sen et al. 2008)	3,327	6	3,327	4,732
CORA (Sen et al. 2008)	1,433	7	2,708	5,429

Table 2: Dataset Statistics

Implementation Details

In this study, all experiments were performed on a Windows server equipped with an NVIDIA GeForce RTX 4090 graphics card with driver version 552.41 and CUDA version 12.4. The experimental environment was based on the Python platform using the PyTorch framework (Python version 3.10.13) and the MATLAB experimental simulation software to realize the proposed method and its comparison with existing data clustering methods.

Comparison of Algorithms

To evaluate the performance of our proposed method, we compared it with several state-of-the-art clustering algorithms, categorized as follows:

- **Node Attributes Only:** K-Means (MacQueen et al. 1967).
- **Both Node Attributes and Graph Structure:** GAE (Graph Auto-Encoder) (Kipf and Welling 2016b), VGAE (Variational Graph Auto-Encoder) (Kipf and Welling 2016b), DAEGC (Deep Attentional Embedding Graph Clustering) (Wang et al. 2019a), GATE (Graph Attention Auto-Encoders) (Salehi and Davulcu 2019), SDCN (Structural Deep Clustering Network) (Bo et al. 2020), EGAE (Embedding Graph Auto-Encoder) (Zhang et al. 2022), NACL (Neighbor-Aware Clustering) (Shen et al. 2023).
- **Node Attributes from Multiple modalitys:** DCCA (Deep Canonical Correlation Analysis) (Andrew et al. 2013), DCCAE (Deep Canonically Correlated Autoencoders) (Wang et al. 2015).
- **Node Attributes from Multiple modalitys and Graph Structure:** CO-GCN (Co-Graph Convolutional Network) (Li, Li, and Wang 2020), SGCMC (Self-Supervised Graph Convolutional Network for Multi-Modal Clustering) (Xia et al. 2021), CMGEC (Consistent Multiple Graph Embedding for Multi-Modal Clustering) (Wang et al. 2021), SGFormer(SGFormer: Simplifying and Empowering Transformers for Large-Graph Representations)(Wu et al. 2023)

Experimental Results

The experimental results of our multi-modal graph clustering method on Cora, Citeseer, WIKI and AMAP datasets show its excellent performance. Overall, our method outperforms the comparison algorithms in ACC, NMI, and ARI metrics on all datasets, which demonstrates the consistency and generalizability of our method. Specifically, our method not only achieves performance improvement on a single dataset but also demonstrates strong clustering ability on different types of graph-structured data.

When comparing our Deep Multi-modal Graph Clustering Algorithm with existing methods on the AMAP dataset, our model demonstrates clear superiority. Specifically, our model achieves a clustering accuracy of 0.768, surpassing the CMGEC model's accuracy of 0.751 by approximately 2.3%. For normalized mutual information (NMI), our model scores 0.680, improving by about 3.5% and 4.3% over DAEGC and EGAE, respectively. In terms of the adjusted Rand Index (ARI), our model scores 0.586, outperforming VGAE and GAE by 5.4% and 9.7%, respectively. These improvements highlight our model's enhanced ability to accurately categorize data, capture true-predicted label relationships, and preserve data connections. The superior performance is attributed to our model's innovative approach to

Method	AMAP			WIKI			Citeseer			Cora		
Evaluation	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-means (MacQueen et al. 1967)	0.280	0.329	0.059	0.439	0.384	0.152	0.544	0.312	0.285	0.500	0.317	0.239
GAE (Kipf and Welling 2016b)	0.741	0.649	0.534	0.332	0.320	0.151	0.380	0.174	0.141	0.530	0.397	0.293
VGAE (Kipf and Welling 2016b)	0.757	0.646	0.556	0.493	0.461	0.299	0.392	0.163	0.101	0.592	0.408	0.347
DAEGC (Wang et al. 2019a)	0.762	0.657	0.584	0.521	0.432	0.337	0.672	0.397	0.410	0.704	0.528	0.496
GATE (Salehi and Davulcu 2019)	0.696	0.611	0.473	0.482	0.343	0.188	0.616	0.401	0.381	0.658	0.527	0.451
SDCN (Bo et al. 2020)	0.543	0.457	0.324	0.432	0.396	0.227	0.663	0.390	0.406	0.495	0.275	0.215
EGAE (Zhang et al. 2022)	0.762	0.652	0.573	0.515	0.480	0.331	0.674	0.412	0.432	0.724	0.540	0.472
NACL (Shen et al. 2023)	0.679	0.647	0.479	0.508	0.454	0.349	0.616	0.376	0.339	0.523	0.326	0.383
DCCA (Andrew et al. 2013)	0.382	0.285	0.147	0.358	0.366	0.170	0.450	0.221	0.204	0.436	0.214	0.160
DCCAE (Wang et al. 2015)	0.475	0.332	0.235	0.321	0.343	0.156	0.503	0.240	0.211	0.472	0.289	0.221
CO-GCN (Li, Li, and Wang 2020)	0.682	0.622	0.511	0.515	0.450	0.330	0.655	0.432	0.423	0.735	**0.567**	0.512
SGCMC (Xia et al. 2021)	0.734	0.642	0.571	0.478	0.436	0.275	0.612	0.336	0.366	0.658	0.502	0.382
CMGEC (Wang et al. 2021)	0.751	0.639	0.565	0.557	0.496	0.316	**0.728**	0.403	0.443	0.737	0.504	0.438
SGFormer (Wu et al. 2023)	0.670	0.588	0.478	0.447	0.406	0.254	0.533	0.257	0.249	0.591	0.436	0.346
DMGC-GTN	**0.786**	**0.684**	**0.599**	**0.576**	**0.514**	**0.408**	0.699	**0.444**	**0.463**	**0.763**	0.566	**0.532**

Table 3: Performance comparison of different methods on four datasets

$\mathcal{L}_r$	$\mathcal{L}_g$	$\mathcal{L}_{pq}$	**ACC**	**NMI**	**ARI**
✓	×	×	0.768	0.681	0.586
×	✓	✓	0.663	0.540	0.438
✓	×	✓	0.763	0.659	0.565
✓	✓	×	0.772	0.671	0.575
✓	✓	✓	0.786	0.684	0.599

Table 4: Ablation Study Results

Embedding Type	**ACC**	**NMI**	**ARI**
Position Embeddings (Pos)	0.145	0.003	0.001
Shortest Path Embeddings (SP)	0.170	0.013	0.005
Pos + SP	0.172	0.013	0.005
Feature Embeddings (FE)	0.652	0.547	0.439
FE + SP	0.659	0.550	0.440
FE + Pos	0.758	0.668	0.575
FE + Pos + SP	0.786	0.684	0.599

Table 5: Performance of various embedding strategies: Position Embeddings (Pos), Shortest Path Embeddings (SP), and Feature Embeddings (FE), along with their combinations.

decomposing and utilizing graph structure information, integrating node intimacy, shortest paths, graph smoothing, and diverse embeddings through a transformer architecture.

Ablation Experimental Analysis

Ablation experiments on the loss function reveal the effects of reconstruction loss, graph structure loss, and KL divergence loss on model performance. Using only KL divergence loss maintains feature fidelity but lacks structural information; using only graph structure and KL divergence loss retains structural information but lacks feature fidelity. Combining these loss functions significantly improves clustering performance by learning from multiple perspectives.

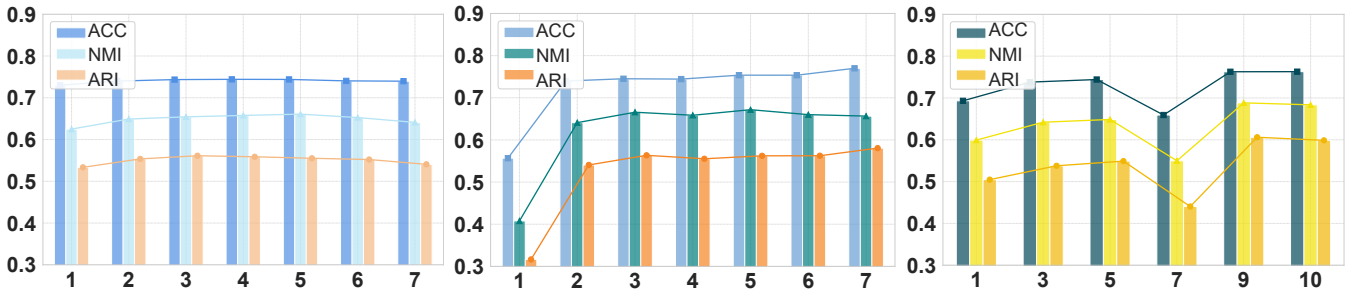
Ablation experiments on feature embedding, position embedding, and shortest path embedding show their effects on model performance. Feature embeddings directly reflect node information and contribute the most to performance. Position and shortest path embeddings provide additional information but are less effective alone. Combining these embedding layers allows the model to understand nodes and graph structures from multiple perspectives, improving clustering performance.

Parameter Analysis

Regularization Parameters λ_1 and λ_2 In multi-modal graph clustering, the regularization parameters λ_1 and λ_2 are crucial for model performance, with their values ranging from 10^{-2} to 10^2. Experimental results show that the model performance is optimized when λ_1 and λ_2 are set to 0.1 and 1, respectively. Conversely, when λ_1 and λ_2 are set to 0.01 and 0.1, the model performance is poor. This is likely because small values for λ_1 and λ_2 cause the reconstruction loss to dominate the overall loss function, leading to an imbalance that diminishes the impact of graph structure and KL divergence losses. Consequently, the model loses graph structure information, generates unbalanced embeddings, overfits local features, and fails to generalize well.

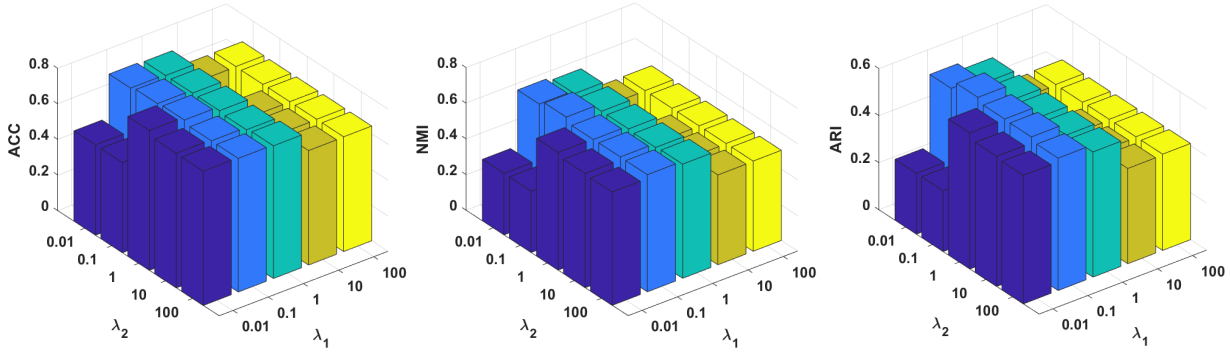
Impact of Hyperparameter t The number of neighbor nodes t significantly affects the performance of graph convolutional networks. Smaller t values capture local structural features, while larger t values capture more extensive node relationships. However, excessively high t values can introduce noise, degrading model performance. Therefore, an optimal t value balances local and global structural features, improving clustering accuracy and robustness.

Influence of the Number of Network Layers The number of network layers has a significant impact on the clustering performance. Increasing the number of layers can improve the model's ability to capture complex relationships and improve feature representation, resulting in better clustering results. However, too many layers can lead to overfitting, increased training difficulty, and potential noise trap-



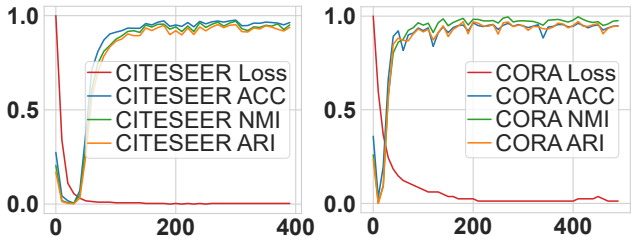
(a) t (b) GTN layer (c) Graph smoothing layer

Figure 2: Impact of key hyperparameters on clustering performance: (a) Temperature parameter t, (b) Transformer layers L, and (c) Graph smoothing iterations K.



(a) ACC (b) NMI (c) ARI

Figure 3: Visualization of the change in ACC, NMI, and ARI metrics as parameters λ_1 and λ_2 are adjusted, demonstrating the impact of geometric relationship consistency and probability distribution consistency on clustering performance.



(a) Citeseer Metrics (b) Cora Metrics

Figure 4: These plots depict the evolution of Training Loss, ACC, NMI, and ARI for the Cora and Citeseer datasets over the course of multiple training epochs.

ping. An optimal number of layers balances feature learning and model complexity, which is important for achieving clustering with excellent results.

Impact of Graph Smoothing Graph smoothing (e.g., Laplace smoothing) improves clustering performance by increasing the consistency of node features and reducing noise. Moderate smoothing helps to create more consistent node features, leading to better clustering. However, excessive smoothing may blur the features of individual nodes and reduce the clustering effect. Therefore, it is crucial to find a balance of smoothing to improve feature consistency while maintaining node specificity.

Conclusions

In this paper, we propose a novel deep multi-modal graph clustering algorithm, termed DMGC-GTN, which significantly enhances clustering performance by thoroughly dissecting graph structural information and utilizing inter-modal relationships. The proposed algorithm uniquely integrates graph Laplacian smoothing filters with attention mechanisms to effectively capture complex node relationships across varying scales, thereby addressing the challenges posed by diverse and heterogeneous data. Leveraging a transformer architecture, DMGC-GTN excels in feature extraction via multiple embedding techniques, enabling it to jointly exploit both the graph structure and the multi-modal feature attributes. This dual-pronged approach facilitates the generation of a unified embedding space, effectively harnessing the complementary information inherent in multi-modal data. Extensive experimental results validate the robustness and effectiveness of the proposed algorithm, demonstrating its superiority over several state-of-the-art methods in terms of clustering accuracy and interpretability.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 62176203 and Grant 62102306, the Fundamental Research Funds for the Central Universities, the Natural Science Basic Research Program of Shaanxi Province (Grant 2023-JC-YB-534), and the Science and Technology Project of Xi'an (Grant 2022JH-JSYF-0009), Initiative Postdocs Supporting Program (Grant BX20190262), Open Project of Anhui Provincial Key Laboratory of Multimodal Cognitive Computation, Anhui University (No. MMC202416), and the Xidian Innovation Fund (Project NoYISJ24017)

References

Andrew, G.; Arora, R.; Bilmes, J.; and Livescu, K. 2013. Deep canonical correlation analysis. In *International conference on machine learning*, 1247–1255. PMLR.

Bo, D.; Wang, X.; Shi, C.; Zhu, M.; Lu, E.; and Cui, P. 2020. Structural deep clustering network. In *Proceedings of the web conference 2020*, 1400–1410.

Cheng, J.; Wang, Q.; Tao, Z.; Xie, D.; and Gao, Q. 2021. Multi-view attribute graph convolution networks for clustering. In *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence*, 2973–2979.

Fan, S.; Wang, X.; Shi, C.; Lu, E.; Lin, K.; and Wang, B. 2020. One2multi graph autoencoder for multi-view graph clustering. In *proceedings of the web conference 2020*, 3070–3076.

Gong, X.; Yuan, D.; Bao, W.; and Luo, F. 2022. A unifying probabilistic framework for partially labeled data learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(7): 8036–8048.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Johnson, S. C. 1967. Hierarchical clustering schemes. *Psychometrika*, 32(3): 241–254.

Kipf, T. N.; and Welling, M. 2016a. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR*, abs/1609.02907.

Kipf, T. N.; and Welling, M. 2016b. Variational Graph Auto-Encoders. *CoRR*, abs/1611.07308.

Li, S.; Li, W.-T.; and Wang, W. 2020. Co-GCN for multi-view semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 4691–4698.

Li, Y.; Nie, F.; Huang, H.; and Huang, J. 2015. Large-scale multi-view spectral clustering via bipartite graph. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

Liu, Y.; Tu, W.; Zhou, S.; Liu, X.; Song, L.; Yang, X.; and Zhu, E. 2022. Deep graph clustering via dual correlation reduction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 7603–7611.

MacQueen, J.; et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, 281–297. Oakland, CA, USA.

Nie, F.; Li, J.; Li, X.; et al. 2017. Self-weighted multiview clustering with multiple graphs. In *IJCAI*, 2564–2570.

Salehi, A.; and Davulcu, H. 2019. Graph Attention Auto-Encoders. *CoRR*, abs/1905.10715.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1): 61–80.

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

Shen, X.; Sun, D.; Pan, S.; Zhou, X.; and Yang, L. T. 2023. Neighbor contrastive learning on learnable graph augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 9782–9791.

Wang, C.; Pan, S.; Hu, R.; Long, G.; Jiang, J.; and Zhang, C. 2019a. Attributed Graph Clustering: A Deep Attentional Embedding Approach. In Kraus, S., ed., *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, 3670–3676. ijcai.org.

Wang, H.; Yang, Y.; and Liu, B. 2019. GMC: Graph-based multi-view clustering. *IEEE Transactions on Knowledge and Data Engineering*, 32(6): 1116–1129.

Wang, W.; Arora, R.; Livescu, K.; and Bilmes, J. 2015. On deep multi-view representation learning. In *International conference on machine learning*, 1083–1092. PMLR.

Wang, X.; Hu, P.; Liu, P.; and Peng, D. 2022. Deep Semisupervised Class- and Correlation-Collapsed Cross-View Learning. *IEEE Transactions on Cybernetics*, 52(3): 1588–1601.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019b. Heterogeneous graph attention network. In *The World Wide Web Conference*, 2022–2032.

Wang, Y.; Chang, D.; Fu, Z.; and Zhao, Y. 2021. Consistent multiple graph embedding for multi-view clustering. *IEEE transactions on multimedia*, 25: 1008–1018.

Wu, Q.; Zhao, W.; Yang, C.; Zhang, H.; Nie, F.; Jiang, H.; Bian, Y.; and Yan, J. 2023. SGFormer: Simplifying and Empowering Transformers for Large-Graph Representations. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xia, W.; Wang, Q.; Gao, Q.; Zhang, X.; and Gao, X. 2021. Self-supervised graph convolutional network for multi-view clustering. *IEEE Transactions on Multimedia*, 24: 3182–3192.

Xie, Y.; Lin, B.; Qu, Y.; Li, C.; Zhang, W.; Ma, L.; Wen, Y.; and Tao, D. 2020. Joint deep multi-view learning for image clustering. *IEEE Transactions on Knowledge and Data Engineering*, 33(11): 3594–3606.

Xu, J.; Ren, Y.; Li, G.; Pan, L.; Zhu, C.; and Xu, Z. 2021. Deep embedded multi-view clustering with collaborative training. *Information Sciences*, 573: 279–290.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How Powerful are Graph Neural Networks? *CoRR*, abs/1810.00826.

Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; and Chang, E. Y. 2015. Network representation learning with rich text information. In *IJCAI*, volume 2015, 2111–2117.

Ying, Z. 2018. Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31(4800-4810): 2.

Yu, X.; Jiang, Y.; Chao, G.; and Chu, D. 2024. Deep Contrastive Multi-View Subspace Clustering With Representation and Cluster Interactive Learning. *IEEE Transactions on Knowledge and Data Engineering*.

Zhang, H.; Li, P.; Zhang, R.; and Li, X. 2022. Embedding graph auto-encoder for graph clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11): 9352–9362.

Zhu, X.; Zhang, S.; He, W.; Hu, R.; Lei, C.; and Zhu, P. 2018. One-step multi-view spectral clustering. *IEEE Transactions on Knowledge and Data Engineering*, 31(10): 2022–2034.