ATTRIBUTED GRAPH CLUSTERING VIA MODULARITY AIDED COARSENING

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph clustering is a widely used technique for partitioning graphs, community detection, and other tasks. Recent graph clustering algorithms depend on combinations of the features and adjacency matrix, or solely on the adjacency matrix. However, in order to achieve high-quality clustering, it is necessary to consider all these components. In this paper, we propose a novel unsupervised learning framework that incorporates modularity with graph coarsening techniques and important graph regularization terms that improve the clustering performance. Furthermore, we also take into account Dirichlet energies for smoothness of signals, spectral similarity, and coarsening reconstructional error. The proposed framework is solved efficiently by leveraging block majorization-minimization, $\log\det$ of the Laplacian, smoothness and modularity, and is readily integrable with deep learning architectures such as GCNs and VGAEs in the form of losses. Extensive theoretical analysis and experiments with benchmark datasets elucidate the proposed framework's efficacy in graph clustering over existing state-of-the-art methods on both attributed and non-attributed graphs.

1 INTRODUCTION

Graph clustering partitions the nodes of a graph into groups based on their specific attributes and interaction patterns learnt from the adjacency matrix and the features. The major applications of graph clustering are social network analysis (Tang et al., 2008), genetics and bio-medicine (Cheng & Ma, 2022) (Buterez et al., 2021), knowledge graphs (Hamaguchi et al., 2017), and computer vision (Mondal et al., 2021) (Caron et al., 2018), etc. The recent surge in demand for unsupervised graph learning techniques because of a surge in data collection has made it infeasible to label the majority of datasets manually. Researchers are bound to resort to unsupervised methods of analysis.

Cut-based and Modularity metrics are the two categories of measures used to determine the quality of partitioning of a graph. Methods optimizing cut-based metrics are usually based on the fact that the Fiedler vector (the eigenvector of the second smallest eigenvalue of the Laplacian) produces a graph cut minimal in the number of edges (Fiedler, 1973; Newman, 2006). However, Leskovec et al. (2008) show that this does not translate well for real-world graphs as the partitions formed do not need to balanced in size (nodes) and cluster quality decreases with an increase in cluster size. Some methods try to normalize these partitions, such as Ratio Cut (Wei & Cheng, 1989) which considers the number of nodes and Normalized Cut (Shi & Malik, 2000) which considers total edge volume of the partition. MinCutPool (Bianchi et al., 2020) performs graph pooling using normalized cuts as a regularizer. A major disadvantage of cut-based metrics is that they assume each node gets mapped to one cluster, which is not consistent with real graph data.

Modularity has been popularized because it measures how different the edge density of a graph is compared to a random graph of the same degree sequence. This can also be viewed as a statistical approach incorporating a null model. A significant edge that modularity offers is that its definition can be extended to directed graphs with overlapping clusters (Nicosia et al., 2009), making it suitable for complex networks. In this paper, we study graph clustering through the lens of modularity. However, it too has drawbacks on small clusters such as its resolution limit (Fortunato & Barthélemy, 2007).

Graph coarsening condenses a large graph into a smaller one while preserving its properties. Feature Graph Coarsening (FGC) (Kumar et al., 2023) is the first optimization framework that takes into account node features instead of just the adjacency. More about this is discussed in Section 2. Recent Graph Neural Network (GNN)-based approaches utilize both node features and graph topology. Most state-of-the-art clustering methods use Variational Graph Auto-Encoder (VGAE)-based backbone because they offer superior performance and have flexibility to use different task-specific encoders/decoders. The existing GNN-free approaches, on the other hand, take only graph topology into account. This is disadvantageous because many real-word graphs have important/meaningful node features and satisfy important homophily and smoothness assumptions, i.e., if two nodes are connected, they should be similar.

Motivation. We aim to utilize the feature graph coarsening framework (which does not perform well on clustering) for clustering. With this in mind, we add a modularity maximization term because it is a good measure for community detection. This is discussed in 2. This comprises the Q-FGC method and greatly improves clustering performance. We also utilize various GNN architectures and integrate our loss function, resulting in Q-GCN, Q-VGAE, and Q-GMM-VGAE and outperform other methods. An important future work can entail utilizing contrastive learning in our framework to get the best of both worlds.

Key Contributions:

- We introduce a new approach to clustering by introducing the first optimization-based framework for attributed graph clustering via coarsening using modularity. Our approach is highly efficient and theoretically convergent. We also reflect on how our formulation overcomes the disadvantages of existing methods.
- We show that the proposed clustering objective can be easily integrated with GNN-based architectures. This allows us to leverage the message-passing mechanism of GNNs to further enhance the performance of our method.
- We conduct extensive experimental validation on real-world and synthetic datasets, showing that our method outperforms existing state-of-the-art methods.

We show a new approach to clustering, as most methods don't take into account various important terms such as the Dirichlet energy (smoothness) of the coarsened graph, modularity maximization, a term to ensure the coarsened graph is connected, and a regularizer for balanced mapping of nodes to clusters, preventing collapse. Our method can also directly observe the relationship between clusters, and also represent each cluster. This is quite important for performing first-hand analysis on large unlabelled datasets.

Novelty This may seem like an incremental work because our modification to the FGC objective is one term. However, here are some points to consider:

- A significant increase in performance from baseline models FGC (**+153%**) and DMoN (**+40%**), which shows that our contribution is to **allow potentially any coarsening method to work in a clustering setting**, as there are *theoretical benefits* in coarsening literature that have not been studied in clustering yet.
- Comprehensive analyses with theoretical guarantees, including supporting proofs of convexity, proofs of KKT optimality, proofs of convergence, ablation studies on the behavior of the loss terms and how it differs from FGC, recovery on a DC-SBM, comparison of runtime and complexities, and also comparison of modularity to make it empirically and theoretically comprehensive.
- Experiments that are not limited to a few small datasets but range from benchmark, to non-attributed, to very large datasets, even though our method does not specialize for them.

2 Related Works

The focus is on three types of existing graph clustering approaches: (1) Graph coarsening methods, (2) VGAE-based clustering (Section 3), and (3) Spectral modularity maximization techniques.

Graph Coarsening and Pooling Methods. DiffPool (Ying et al., 2018)learns soft cluster assignments at each layer of the GNN and two extra losses, entropy to penalize the soft assignments and a link prediction based loss. SAGPool (Lee et al., 2019) calculates attention scores and node embeddings to determine the nodes that need to be preserved or removed. Top-k (Gao & Ji, 2019) also works by sparsifying the graph with the learned weights. MinCutPool (Bianchi et al., 2020) formulates a differentiable relaxation of spectral clustering via pooling. However, Tsitsulin et al. (2023) show that it's orthogonal regularization dominates over the clustering objective and the objective is not optimized. Some disadvantages of these methods are stability and computational complexity in the case of SAGPool and DiffPool and convergence in MinCutPool.

Modularity Optimization. In theory, a higher value of modularity (**Q**) is associated with better quality clusters. However, maximizing modularity over all partitions of a graph is **NP-hard** (Brandes et al., 2008). Various heuristic algorithms have been established that solve this problem including sampling, simulated annealing (sim, 2005) (Guimerà & Amaral, 2005), mathematical programming (Agarwal, G. & Kempe, D., 2008), and greedy agglomerative algorithms (Newman, 2004; Blondel et al., 2008a). The usage of these algorithms has plummeted over the years because they rely solely on the topological information of graphs and ignore node features. Another major drawback is that they require intensive computation and are thus impractical for large-scale networks. The Louvain (Blondel et al., 2008b) and Leiden algorithms improve that. With the development of GNNs, this also improved drastically; however, modularity maximization using GNNs still needs to be studied. To the best of our knowledge, Deep Modularity Network (DMoN) (Tsitsulin et al., 2023) and Modularity-Aware GAE (Salha-Galvan et al., 2022) are the only deep learning architectures to use modularity in training and are crucial baselines for our method. DMoN's clustering objective optimizes only for modularity (with a collapse regularization to prevent all nodes being assigned one cluster) but does not consider smoothness of signals and offers no theoretical guarantees about convergence. Modularity-Aware GAEs and VGAEs use a prior membership matrix using Louvain algorithm and optimize for modularity using RBF kernel as a same-community assignment proxy.

Deep Graph Clustering. Previous literature can be classified based on contrastive and non-contrastive methods. On the non-contrastive side, Pan et al. (2018) proposed ARGA and ARVGA, enforcing the latent representations to align to a prior using adversarial learning. By utilizing an attention-based graph encoder and a clustering alignment loss, Wang et al. (2019) propose DAEGC. Subsequently, SDCN (Bo et al., 2020) propose to learn a "structure-aware" representation by their delivery operator. Liu et al. (2022) design the DCRN model to alleviate representation collapse by a propagation regularization term minimizing the JSD between the latent and its product with normalized A. Contrastive methods include AGE (Cui et al., 2020) which builds a training set by adaptively selecting node pairs that are highly similar or dissimilar after filtering out high-frequency noises using Laplacian smoothing. Zhao et al. (2021) propose GDCL to correct the sampling bias by choosing negative samples based on the clustering label. Liu et al. (2023a) propose SCGC, which uses two MLPs to get augmented node features, and then find a cross-view similarity matrix to contrast against A.

3 BACKGROUND

In this section, we introduce the concepts that play a foundational role in the formulation of our method.

Notations. Let $G = \{V, E, A, X\}$ be a graph with node set $V = \{v_1, v_2, ..., v_p\}$ ($|V| = p$), edge set $E \subset V \times V\}(|E| = e)$, weight (adjacency) matrix A and node feature matrix $X \in \mathbb{R}^{p\times n}$. Also, let $\mathbf{d} \in \mathbb{Z}_+^p$ be the degree vector. Then, the graph Laplacian is $\Theta = \text{diag}(\mathbf{d}) - A$ and the set of all valid Laplacian matrices is defined as: $S_\Theta = \{\Theta \in \mathbb{R}^{p\times p} | \Theta_{ij} = \Theta ji \leq 0 \text{ for } i \neq j, \Theta_{ii} = \sum_{j=1}^{p} \Theta_{ij}\}$

3.1 GRAPH COARSENING

Graph coarsening is a classical method used in large-scale machine learning to construct a smaller graph G_c from the original graph $G = \{V, E, A, X\}$ while preserving properties of G. The commonly used measures of similarity are hyperbolic error (Bravo Hermsdorff & Gunderson, 2019), reconstruction error, ϵ- similarity (Loukas, 2019), and spectral similarity. For the coarsened graph G_c, we denote the new vertex set as $\tilde{V}(|\tilde{V}| = k)$ features as $\tilde{X} \in \mathbb{R}^{k\times n}$ and the Laplacian Θ_C. We define $C \in [0, 1]^{p\times k}$ to be the *soft* cluster assignment matrix (i.e. each non-zero entry of matrix C i.e., C_{ij} indicates probability of i-th node of G mapped to j-th cluster or supernode of G_c). The coarsening matrix C plays the role of cluster assignment matrix in our study, i.e. k represents the number of clusters. Moreover, the Laplacian and feature matrix of the coarsened graph and original graph together satisfy the following properties:

$$X = C\tilde{X}, \quad \Theta_C = C^T \Theta C \text{ where, } C \in \mathcal{S}_c \tag{1}$$

$$\mathcal{S}_c = \{C \in \mathbb{R}_+^{p\times k} | \langle C_i, C_j \rangle = 0 \ \forall \, i \neq j, \langle C_l, C_l \rangle = d_i, ||C_i||_0 \geq 1 \text{ and } ||[C^T]_i||_0 = 1\} \tag{2}$$

Kumar et al. (2023) proposed the first optimization-based framework FGC to incorporate both graph topology using the Laplacian matrix and node features. Their method preserves important qualities of the original graph G in the coarsened graph G_c.

3.2 Variational Graph Auto Encoders (VGAEs)

VGAEs (Kipf & Welling, 2016a) are an increasingly popular class of GNNs that leverage variational inference techniques for learning latent representations for graphs in unsupervised settings. A typical architecture involves using GCN-based encoders to transform a high-dimensional graph to a low-dimensional space, followed by a decoder to reconstruct the adjacency matrix. Due to its ability to attain competitive performance on a multitude of tasks, including node classification and link prediction, it is the preferred backbone for contemporary graph-based architectures.

Many attempts have been made to use VGAEs with k-means on latent embeddings, but it has been unsuitable for clustering. This is primarily because embedded manifolds obtained from VGAEs are curved and must be flattened before any clustering algorithms using Euclidean distance are applied. Refer to supplementary material K for an explanation. VGAEs only use a single Gaussian prior for the latent space, whereas clustering requires the integration of meta-priors. Additionally, the inner-product decoder fails to capture locality and cluster information in the formed edges. Several clustering-oriented variants of VGAEs (Mrabah et al., 2022) (Hui et al., 2020) have been developed that overcome most of these challenges. GMM-VGAE (Hui et al., 2020) is one such architecture and is relevant to our paper. GMM-VGAE partitions the latent space using a Gaussian Mixture Model and assigns a separate prior for each cluster to better model complex data distributions instead of the single prior in VGAEs. Despite the improvement in performance, it's inner-product decoder still suffers from the same problems.

3.3 Spectral Modularity Maximization

Spectral Clustering is the most direct approach to graph clustering, where we minimize the volume of inter-cluster edges (i.e. the total number of edges in between clusters and not inside them). Modularity is the difference the number of edges between nodes in a cluster $\mathbf{C_i}$ and the expected number of such edges in a random graph with identical degree sequence. It is mathematically defined as

$$\mathcal{Q} = \frac{1}{2e}\sum_{i,j}\left[A_{ij} - \frac{d_i d_j}{2e}\right]\delta(c_i, c_j) \tag{3}$$

where $\delta(c_i, c_j)$ is the Kronecker delta. Maximizing this form of modularity is NP-hard (Brandes et al., 2008). However, we can approximate it with a spectral relaxation, which involves a modularity matrix B. The modularity matrix B and spectral modularity are defined as:

$$B = A - \frac{\mathbf{d}\mathbf{d}^T}{2e}, \quad \mathcal{Q} = \frac{1}{2e}Tr(C^T BC) \tag{4}$$

Similar to the Graph Laplacian matrix $\mathbf{\Theta}$, $\mathbf{B}$ is symmetric and is defined to have a row-sum and column-sum as 0, thereby making $\mathbf{1}$ as one of it's eigenvector and 0 as the corresponding eigenvalue. These spectral properties of the modularity matrix are also seen in the Laplacian, as noted in Newman (2006), which is of course a crucial element in spectral clustering. Also, modularity is maximized when $u_1^T s$ is maximized where u are eigenvectors of B and s is the community assignment vector, i.e., placing the majority of the summation in Q on the first (and largest) eigenvalue of B. Moreover, modularity is closely associated with community detection. These special spectral properties make $\mathbf{B}$ as the ideal choice for graph clustering.

4 Proposed Method

In this section, we present our proposed clustering framework that overcomes the aforementioned drawbacks of existing methods and achieves competitive performance on six benchmark datasets.

4.1 Modularity aided Feature Graph Coarsening

Following the success of FGC, we pose the clustering problem as a special case of graph coarsening, when the number of nodes in the coarsened graph is equal to the number of clusters. FGC performs excellently for coarsening ratios (k/p) in the range of 0.01-0.1, preserving the eigenvalue distribution of the original graph. However, for clustering purposes, we usually have p of the order of thousands, whereas k is usually less than 10, implying a coarsening ratio lower than 0.001. So empirically and experimentally (Table 2), it is evident that FGC alone is inadequate. We assume that the original graph is smooth considering that most real world graphs are smooth and homophilic. Note that this doesn't mean our method fails on synthetic graphs. We have studied this in the ablation study 5.5, and is able to recover the community structure completely. We introduce an optimization-based framework for attributed graph clustering via coarsening with modularity as follows:

$$\min_{\tilde{X},C} \mathcal{L}_{MAGC} = \text{tr}(\tilde{X}^T C^T \Theta C \tilde{X}) + \frac{\alpha}{2} \left\| C\tilde{X} - X \right\|_F^2 - \frac{\beta}{2e} \text{tr}(C^T B C) \tag{5}$$

$$- \gamma \log \det(C^T \Theta C + J) + \frac{\lambda}{2} \left\| C^T \right\|_{1,2}^2$$

$$\text{subject to } \mathcal{C} = \{C \in \mathbb{R}^{p\times k} | C \geq 0, \ \left\| C_i^T \right\|_2^2 \leq 1\} \ \forall i \ \text{ where, } J = \frac{1}{k}\mathbf{1}_{k\times k} \tag{6}$$

The significance of each term is term in the optimization objective is:

- $\text{tr}(\tilde{X}^T C^T \Theta C \tilde{X})$ is the Dirichlet energy of the coarsened graph (because $\Theta_C = C^T \Theta C$), a measure of the smoothness of it's signals. Graph-based modeling is based on the assumption that graph signal variations are smooth between connected nodes. This term ensures the smoothness property of the original graph is passed onto the coarsened one.
- $\left\| C\tilde{X} - X \right\|_F^2$ simply enforces the relaxation of the constraint $X = C\tilde{X}$, i.e., the large graph X gets coarsened to the the smaller $\tilde{X}$ with the cluster assignment/loading matrix C.
- $\text{tr}(C^T B C)$ is the modularity term for better cluster quality. The negative of this is added to the loss as we want to maximize modularity while minimizing the loss. This is discussed in 3.3
- -$\log \det(C^T \Theta C + J)$ is the log-determinant term to ensure the coarsened graph is connected. This works because it can be written as $-\sum_i \log \lambda_i$ where λ_i's are the eigenvalues of the Θ_C. By minimizing this, we are ensuring that minimal number of λ_i's are 0, since the number of connected components in a graph is equal to the multiplicity of 0 in it's laplacian eigenvalues.
- $\left\| C^T \right\|_{1,2}^2$ is an $\ell_{1,2}^2$ norm regularizer for a balanced mapping of nodes to clusters (i.e., every cluster has at least one node and each node is mapped to a cluster) (Kim & Park, 2007).

Update rules. Since the resulting optimization problem is multi-block non-convex and there is no closed-form solution, we use Block Majorization-Minimization framework similar to FGC (Kumar et al., 2023). We iteratively solve the problem by updating **C** and $\tilde{\mathbf{X}}$ alternatively while keeping the other constant. These iterations are performed until convergence or until some stopping criteria is met.

Lemma 1. *The problem equation 6 with respect to $\tilde{X}$ while keeping C constant is a convex optimization problem.*

Proof. As $\text{tr}(\tilde{X}^T C^T \Theta C \tilde{X})$ and $\|C\tilde{X} - X\|_F^2$ are convex functions with respect to $\tilde{X}$ while keeping C constant and $\tilde{X} \in \mathbb{R}^{p\times n}$ together makes the problem equation 6 with respect to $\tilde{X}$ as a convex optimization problem. □

Lemma 2. *The problem equation 6 with respect to C is convex optimization problem while keeping $\tilde{X}$ constant.*

Proof. All the terms in the objective function of problem 6 with respect to C while keeping $\tilde{X}$ are convex functions and the set $\mathcal{C}$ is a closed convex set together makes the problem a convex optimization problem. More details are in supplementary material. □

Considering the objective function with respect to C as $f(C)$, The majorized function of $f(C)$ at C^t using the first order taylor series expansion is:

$$g(C|C^t) = f(C^t) + (C - C^t)\nabla f(C^t) + \frac{L}{2} \left\| C - C^t \right\|^2 \tag{7}$$

$$\min_{C \in \mathcal{S}_C} \frac{1}{2} C^T C - C^T \Big(C^t - \frac{1}{L} \nabla f(C^t) \Big) \tag{8}$$

Equation 8 is the majorized problem of equation 6. The optimal solution to Eqn. 8, found by using Karush–Kuhn–Tucker (KKT) optimality conditions is (Proof is deferred to the supplementary material B):

$$C^{t+1} = \Big(C^t - \frac{1}{L} \nabla f\big(C^t\big) \Big)^+ \tag{9}$$

$$\text{where, } \nabla f\big(C^t\big) = -2\gamma \Theta C^t ({C^t}^T \Theta C^t + J)^{-1} + \alpha (C^t \tilde{X} - X)\tilde{X}^T + 2\Theta C^t \tilde{X}\tilde{X}^T + \lambda C^t \mathbf{1}_{k\times k}$$

$$- \frac{\beta}{e} B C^t \tag{10}$$

$$\tilde{X}^{t+1} = \Big(\frac{2}{\alpha} C^T \Theta C + C^T C \Big)^{-1} C^T X \tag{11}$$

Convergence Analysis.

Theorem 1. *The sequence $\{\mathbf{C}^{t+1}, \tilde{\mathbf{X}}^{t+1}\}$ generated by Algorithm 1 converges to the set of Karush–Kuhn–Tucker (KKT) optimality points for Problem 6*

Proof. The detailed proof can be found in the supplementary material C. □

Algorithm 1 Q-FGC Algorithm

```
Require: $G(\mathbf{X}, \Theta), \alpha, \beta, \gamma, \lambda$
1: $t \leftarrow 0$
2: while Stopping Criteria not met do
3:     Update $\mathbf{C}^{t+1}$ as in Eqn. 9 and Update $\tilde{\mathbf{X}}^{t+1}$ as in Eqn. 11
4:     $t \leftarrow t + 1$
5: end while
6: return $\mathbf{C}^t, \tilde{\mathbf{X}}^t$
```

Complexity Analysis. The worst-case time complexity of a loop in Algorithm 1 is $\mathcal{O}(p^2k + pkn)$ because of the matrix multiplication of $\Theta(p \times p)$ with $\mathbf{C}(p \times k)$ and matrix multiplication of $\mathbf{C}(p \times k)$ with $\tilde{\mathbf{X}}(k \times n)$ in the update rule of $\mathbf{C}$ equation 9. Note that in clustering, k is much smaller than p. This makes our method much faster than previous optimization based methods and faster than GCN-based clustering methods which have complexities around $\mathcal{O}(p^2n + pn^2)$. We discuss this more in Supplementary Material L.

4.2 Integrating with GNNs

Our optimization framework is easily integrable with deep learning methods (GNNs) by adding equation 6 in the form of a loss function to be minimized using gradient descent. We show this integration and their results on the most widely-used architectures like Graph Convolutional Networks (GCNs) (Kipf & Welling, 2016b), VGAEs(Kipf & Welling, 2016a), and a variant of VGAE (GMM-VGAE) (Hui et al., 2020). The former two are popular because of the relative simplicity.

Note that for clustering, learning C is of more importance to us than learning $\tilde{X}$, as it tells us about which nodes belong to which clusters, whereas $\tilde{X}$ tells us about a mean/prototypical element in that cluster, which can't be compared directly with the ground truth labels. So, we just learn C instead of $\tilde{X}$, and calculate it in each iteration by multiplying X with the pseudo-inverse of C because of equation 1.

Q-GCN. We integrate our loss function ($\mathcal{L}_{MAGC}$ equation 6) into a simple three-layer GCN model. We learn the soft cluster assignments $\mathbf{C}$ by taking it to be the output of the final GCN layer. So, our input features go from $p \times n$, to $p \times k$. The architecture and loss can be seen in Figure 1.

Q-VGAE. VGAEs operate by reconstructing the adjacency matrix.

We have provided a theoretical summary of VGAE in supplementary material F. The loss can be written as

$$\mathcal{L}_{VGAE} = \lambda_{recon} \underbrace{\mathbb{E}_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}[\log p(\hat{\mathbf{A}}|\mathbf{Z})]}_{\text{Reconstruction Error}} - \lambda_{kl} \underbrace{\mathrm{KL}[q(\mathbf{Z}|\mathbf{X},\mathbf{A}) \mid\mid p(\mathbf{Z})]}_{\text{Kullback-Leibler divergence}} \tag{12}$$

where, $\mathbf{Z}$ represents the latent space of the VGAE.

On top of this architecture, we add a GCN layer taking Z as input and predicting C. So, for a VGAE, we are minimizing the sum of three losses: the reconstruction loss, the KL-divergence loss and our loss. $\mathcal{L}_{Q-VGAE} = \mathcal{L}_{MACG} + \mathcal{L}_{VGAE}$

Q-GMM-VGAE. This variant of VGAE uses a Gaussian Mixture Model (GMM) on the latent space to more effectively discover data distributions. This works well because it minimizes the evidence lower bound (ELBO/variational lower bound (Hui et al., 2020; Kingma & Welling, 2014; Kipf & Welling, 2016a)) using multiple priors instead of a single Gaussian prior in a normal VGAE. Intuitively, taking as many priors as number of clusters seems like a good idea, which is what Hui et al. (2020) do.

5 Experiments

5.1 Benchmark Datasets and Baselines

We evaluate our method on a range of datasets, ranging from small attributed datasets like Cora and CiteSeer to larger ones like PubMed, and even unattributed datasets like Airports (Brazil, Europe and USA). We also experiment with very large datasets like CoauthorCS/Physics, AmazonPhoto/PC and ogbn-arxiv. A summary of all the datasets used in our paper is given in the supplementary material D.

We compare the performance of our method against three types of existing state-of-the-art methods based on the provided input and type of architecture: a) methods that use only the node attributes; b) methods that only use graph-structure; c) methods that use both graph-structure and node attributes.

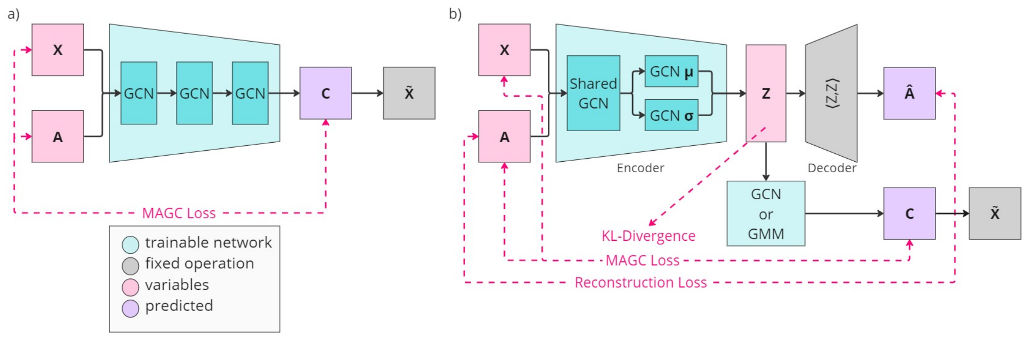


Figure 1: **a) The architecture of Q-GCN.** We train the encoder to learn the soft cluster assignment matrix $\mathbf{C}$. $\tilde{\mathbf{X}}$ is obtained using the properties of coarsened graphs $\tilde{\mathbf{X}}^{t+1} = \mathbf{C}^{t+1^\dagger}\mathbf{X}$ (Eqn 1). Our proposed MAGC loss is now computed using $\mathbf{C}$ and $\tilde{\mathbf{X}}$.
b) The architecture of Q-VGAE/Q-GMM-VGAE. The three-layered GCN encoder, which takes $\mathbf{X}$ and $\mathbf{A}$ as input, learns the latent representation of the graph $\mathbf{Z}$. $\mathbf{Z}$ is then passed through an inner-product decoder to reconstruct the adjacency matrix $\hat{\mathbf{A}}$ Reconstruction loss is calculated between $\hat{\mathbf{A}}$ and $\mathbf{A}$ and KL-Divergence on $\mathbf{Z}$. In Q-VGAE (Q-GMM-VGAE), $\mathbf{Z}$ is parallelly passed through a GCN layer (GMM) to output the soft cluster assignments $\mathbf{C}$. MAGC loss is now computed in the same way as Q-GCN.

The last category can be further subdivided into three sets: i) graph coarsening methods; ii) GCN-based architectures; iii) VGAE-based architectures and contrastive methods; iv) largely modified VGAE architectures.

5.2 Metrics

We analyse the performance of our method using label alignment metrics which relate ground truth node labels to cluster assignments. We report Normalised Mutual Information (NMI), Adjusted Rand Index (ARI), and Accuracy (ACC). Higher value for these metrics are desirable. Please refer to Supplementary Material D for an explanation.

We chose NMI as the primary metric for selecting the most performant models because of these reasons and because it is considered most important by the majority of graph clustering literature.

Training Details. We have used the same architectures for Q-GCN, Q-VGAE and Q-GMM-VGAE throughout. Q-GCN is composed of 3 GCN layers with hidden sizes as 128 and 64. Q-VGAE and Q-GMM-VGAE are composed of 3 GCN layers for the encoder (1 shared with output size 128, and 1 each for μ and σ of output size 64). Q-VGAE has an additional GCN layer after the latent space to generate C, whereas Q-GMM-VGAE utilises a GMM. More training details are available in the Supplementary Material E.

5.3 Attributed Graph Clustering

We highlight our key results on the three classical datasets Cora, CiteSeer, and PubMed (Sen et al., 2008) in Table 1. Our proposed method surpasses all existing methods in terms of NMI while also achieving competitive performance in terms of Accuracy and ARI. As mentioned above the best models were selected based on the NMI scores. The results for very large datasets are present in the supplementary material I. Note that we perform full-batch training (passing the whole graph) instead of randomly-sampled batches (which involves cutting the graph into multiple subgraphs, adding a great degree of bias as the community structure is broken) like in some other works such as S3GC (Devvrit et al., 2022). For very large graphs such as ogbn-arxiv, we have no choice but to use batching.

5.4 Non-Attributed Graph Clustering

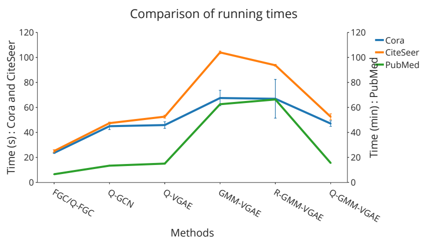
For non-attributed graphs, we use a one-hot vector of the degree vector as the features. This is a primitive way of making features, and there are learning based methods such as DeepWalk (Perozzi et al., 2014b), node2vec (Grover & Leskovec, 2016) etc. This was done for the sake of fair comparison, as the other methods also use this as node features. We present our results in Table 2a. Clearly, we achieve competitive or even higher performance in terms of NMI even for non-attributed datasets.

	Cora			CiteSeer			PubMed		
Method	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**
K-means	34.7	16.7	25.4	38.5	17.1	30.5	57.3	29.1	57.4
Spectral Clustering	34.2	19.5	30.2	25.9	11.8	29.5	39.7	3.5	52.0
DeepWalk (Perozzi et al., 2014a)	46.7	31.8	38.1	36.2	9.7	26.7	61.9	16.7	47.1
Louvain (Blondel et al., 2008b)	52.4	42.7	24.0	49.9	24.7	9.2	30.4	20.0	10.3
GAE [NeurIPS'16] (Kipf & Welling, 2016a)	61.3	44.4	38.1	48.2	22.7	19.2	63.2	24.9	24.6
DGI [ICLR'19] (Veličković et al., 2019)	71.3	56.4	51.1	68.8	44.4	45.0	53.3	18.1	16.6
GIC [PAKDD'21] (Mavromatis & Karypis, 2021)	**72.5**	53.7	50.8	69.6	45.3	46.5	**67.3**	31.9	**29.1**
DAEGC [IJCAI'19] (Wang et al., 2019)	70.4	52.8	49.6	67.2	39.7	41.0	67.1	26.6	27.8
GALA [ICCV'19] (Park et al., 2019)	74.5	57.6	53.1	69.3	44.1	44.6	69.3	32.7	32.1
AGE [KDD'20] (Cui et al., 2020)	73.5	57.5	50.0	69.7	44.9	34.1	71.1	31.6	33.4
DCRN [AAAI'22] (Liu et al., 2022)	61.9	45.1	33.1	70.8	45.8	47.6	69.8	32.2	31.4
FGC [JMLR'23] (Kumar et al., 2023)	53.8	23.2	20.5	54.2	31.1	28.2	67.1	26.6	27.8
Q-FGC (Ours)	65.8	51.8	42.0	65.9	40.8	40.1	66.7	**32.8**	27.9
Q-GCN (Ours)	71.6	**58.3**	**53.6**	**71.5**	**47.0**	**49.1**	64.1	32.1	26.5
SCGC [IEEE TNNLS'23] (Liu et al., 2023a)	**73.8**	56.1	51.7	**71.0**	45.2	46.2	-	-	-
MVGRL [ICML'20] (Hassani & Khasahmadi, 2020)	73.2	56.2	**51.9**	68.1	43.2	43.4	69.3	**34.4**	**32.3**
VGAE [NeurIPS'16] (Kipf & Welling, 2016a)	64.7	43.4	37.5	51.9	24.9	23.8	**69.6**	28.6	31.7
ARGA [IJCAI'18] (Pan et al., 2018)	64.0	35.2	61.9	57.3	34.1	**54.6**	59.1	23.2	29.1
ARVGA [IJCAI'18] (Pan et al., 2018)	63.8	37.4	62.7	54.4	24.5	52.9	58.2	20.6	22.5
R-VGAE [IEEE TKDE'22] (Mrabah et al., 2022)	71.3	49.8	48.0	44.9	19.9	12.5	69.2	30.3	30.9
Q-VGAE (Ours)	72.7	**58.6**	49.6	66.1	**47.4**	50.2	64.3	31.6	28.0
VGAECD-OPT [Entropy'20] (Choong et al., 2020)	27.2	37.3	22.0	51.8	25.1	15.5	32.2	25.0	26.1
Mod-Aware VGAE [NN'22] (Salha-Galvan et al., 2022)	67.1	52.4	44.8	51.8	25.1	15.5	-	30.0	29.1
GMM-VGAE [AAAI'20] (Hui et al., 2020)	71.9	53.3	48.2	67.5	40.7	42.4	71.1	29.9	33.0
R-GMM-VGAE [IEEE TKDE'22] (Mrabah et al., 2022)	**76.7**	57.3	**57.9**	68.9	42.0	43.9	**74.0**	33.4	**37.9**
Q-GMM-VGAE (Ours)	76.2	**58.7**	56.3	**72.7**	**47.4**	**48.8**	69.0	**34.8**	34.0

Table 1: Comparison of all methods on attributed datasets.

	Brazil			Europe			USA		
Method	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**
GAE [NeurIPS'16]	62.6	37.8	30.8	47.6	19.9	12.7	43.9	13.6	11.8
DGI [ICLR'19]	64.9	31.0	30.4	48.6	16.1	12.3	**52.2**	22.9	**21.7**
GIC [PAKDD'21]	40.5	23.5	14.1	40.4	9.4	6.2	49.7	22.1	19.9
DAEGC [AAAI'19]	71.0	47.4	41.2	53.6	30.9	23.3	46.4	27.2	18.4
Q-GCN (Ours)	51.1	31.9	23.7	45.5	30.8	25.1	43.8	19.1	14.8
VGAE [NeurIPS'16]	64.1	38.0	30.7	49.9	23.5	16.7	45.8	**23.6**	15.7
Q-VGAE (Ours)	50.1	35.0	19.8	46.6	19.5	17.5	46.2	19.5	16.9
GMM-VGAE [AAAI'20]	70.2	**46.0**	41.9	53.1	31.1	24.4	48.1	21.9	13.2
R-GMM-VGAE [IEEE TKDE'22]	**73.3**	45.6	**42.5**	**57.4**	31.4	**25.8**	50.8	23.1	15.3
Q-GMM-VGAE (Ours)	68.4	**46.0**	42.4	47.9	**32.2**	23.5	46.6	22.5	13.1

(a) Comparison of all methods on non-attributed datasets using degree.



(b) Comparison of running times of methods. Note that the scale for PubMed is in minutes (right axis), whereas for Cora and CiteSeer is in seconds.

5.5 Ablation Studies

Visualization of the latent space First, we visualize how the latent space of the Q-VGAE and Q-GMM-VGAE changes over time for the Cora dataset. Plots for the rest of the datasets can be found in the supplementary material G. We use UMAP (Uniform Manifold Approximation and Projection) (McInnes et al., 2018) for dimensionality reduction from the latent space (64) to two dimensions.

Comparison of running times In Fig 2b we compare the running times of our method with other baselines. Our method take less than half as much time over all datasets. Especially on the largest dataset we have tested, PubMed, state-of-the-art methods GMM-VGAE and R-GMM-VGAE (unmodified) take about 60 minutes to complete whereas our Q-GMM-VGAE runs in under 15 minutes, a 75% reduction in running time, while also performing better. We also want to highlight that Q-FGC runs even faster, in just 6 minutes (for PubMed) and achieves 90% of the performance.

Modularity Metric Comparison We perform an experiment to see how much we gain in modularity over other baselines on Cora, CiteSeer and PubMed datasets. We report two types graph-based metrics, modularity $\mathcal{Q}$ and conductance $\mathcal{C}$, which don't require labels. Conductance measures the fraction of total edge volume that points outside the cluster. $\mathcal{C}$ is the average conductance across all clusters and a lower value is preferred. From Table 4a, we can see that even though DMoN (Tsitsulin et al., 2023) has the highest modularity, we achieve much higher NMI. Similarly for CiteSeer, we see a 40% improvement in NMI with only a 8% drop in the modularity, placing us closer to the ground truth. Moreover, we can also see that our methods perform better than the ones they were based on, i.e. Q-FGC > FGC, Q-VGAE > VGAE. Even though modularity is a good metric to optimize for, maximum modularity labelling of a graph does not always correspond to the ground truth labelling.

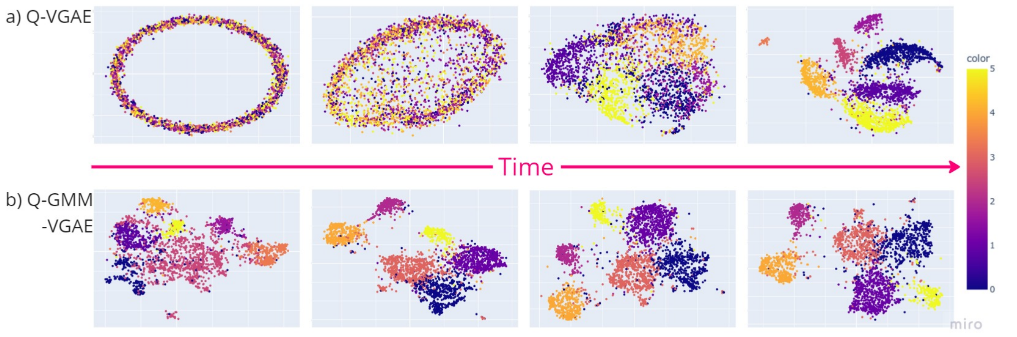
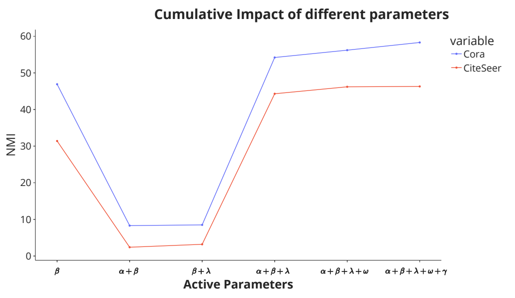


Figure 3: Evolution of the latent space of a) Q-VGAE and b) Q-GMM-VGAE over time for Cora. Colors represent cluster assignments.

	Cora			CiteSeer			PubMed		
	$\mathcal{C}\downarrow$	$\mathcal{Q}\uparrow$	NMI $\uparrow$	$\mathcal{C}\downarrow$	$\mathcal{Q}\uparrow$	NMI $\uparrow$	$\mathcal{C}\downarrow$	$\mathcal{Q}\uparrow$	NMI $\uparrow$
DMoN	12.2	**76.5**	48.8	5.1	**79.3**	33.7	17.7	**65.4**	29.8
FGC	58.4	25	23.1	41.6	41.1	31	21.6	44.1	20.5
Q-FGC	13.3	72.5	51.7	16.8	64.9	40.16	26	40.3	28.1
Q-GCN	13.6	73.3	58.3	5.8	74.5	46.7	**8.27**	55	31.5
VGAE	17.6	60.8	38.1	12.8	55.8	21	13.5	45.8	26.9
Q-VGAE	**9.5**	71.5	**58.4**	**4.6**	72.4	**47.3**	9.4	52.12	**31.8**

(a) Comparison of modularity and conductance at the best NMI with DMoN. Note that DMoN is optimizing only modularity, whereas we are optimizing other important terms as well, as mentioned in Eqn 6, and thus gain a lot on NMI by giving up a small amount of modularity, making us closer to the ground truth.



(b) Impact of active parameters on clustering performance. All terms in the loss equation 6 are represented by their parameters, for example the modularity term is represented by β. Additionally, ω represents the non-parameterized term.

For this reason, it is important to have the other terms in our formulation as well. By optimizing modularity, we can get close to the optimal model parameters (at which NMI would be 1), but will be slightly off-course. We can think of the other terms as correcting this trajectory.

Stochastic Block Model (SBM) and variants Please refer to supplementary material H.

Importance of and Evolution of different loss terms We analyze the evolution of the different loss terms during training, and also try to measure the impact of each term separately by removing terms from the loss one by one, as shown in Supplementary Material M. Also, we found that $||C\tilde{X} - X||_F^2$ is the most sensitive to change in its weight α, followed by the terms related to γ, β and then λ. This makes sense because if that constraint(relaxation) is not being met, then C would have errors. Even though some of the terms do the heavy lifting, the other regularization terms do contribute to performance and more importantly, change the nature of C : The term ω corresponds to smoothness of signals in the graph being transferred to the coarsened graph; this would affect C by encouraging local "patches"/groups to belong to the same cluster. The term γ ensures that the coarsened graph is connected - i.e. preserving inter-cluster relations, which simple contrastive methods destroy; this affects C by making it so that Θ_C has minimal multiplicity of 0-eigenvalues.

6 Conclusion

In this paper, we have developed an optimization-based attributed graph clustering framework, Q-FGC, and its integration with deep learning-based architectures Q-GCN, Q-VGAE and Q-GMM-VGAE. We have performed graph clustering tasks using the proposed methods on real-world benchmark datasets and it is evident that incorporating modularity and graph regularizations into the coarsening framework improves the clustering performance. Also, integrating the proposed method with deep learning-based architecture improves the clustering performance by a significant amount. The proposed algorithms are provably convergent and much faster than state-of-the-art algorithms. A limitation of this method is that if for a graph, the ground truth labelling gives a low modularity, our method will be slower and not as stable convergence. However, it still manages to reach and surpass state-of-the-art methods on the Airports dataset, in which all graphs have a low modularity on the ground truth labels.

REFERENCES

Functional cartography of complex metabolic networks. *Nature*, 433(7028):895–900, February 2005. ISSN 0028-0836. doi: 10.1038/nature03288.

Agarwal, G. and Kempe, D. Modularity-maximizing graph communities via mathematical programming. *Eur. Phys. J. B*, 66(3):409–418, 2008. doi: 10.1140/epjb/e2008-00425-1. URL https://doi.org/10.1140/epjb/e2008-00425-1.

Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJzRZ-WCZ.

Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008 (10):P10008, oct 2008a. doi: 10.1088/1742-5468/2008/10/P10008. URL https://dx.doi.org/10.1088/1742-5468/2008/10/P10008.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008 (10):P10008, October 2008b. ISSN 1742-5468. doi: 10.1088/1742-5468/2008/10/p10008. URL http://dx.doi.org/10.1088/1742-5468/2008/10/P10008.

Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. Structural deep clustering network. In *WWW*, pp. 1400–1410. ACM / IW3C2, 2020.

Igor Bogdanov and Vladimir Shchur. Variational autoencoders with euclidean and hyperbolic latent spaces for population genetics. In *2021 XVII International Symposium "Problems of Redundancy in Information and Control Systems" (REDUNDANCY)*, pp. 91–94, 2021. doi: 10.1109/REDUNDANCY52534.2021.9606448.

Ulrik Brandes, Daniel Delling, Marco Gaertler, R. Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 20:172 – 188, 03 2008. doi: 10.1109/TKDE.2007.190689.

Gecia Bravo Hermsdorff and Lee Gunderson. A unifying framework for spectrum-preserving graph sparsification and coarsening. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/cd474f6341aeffd65f93084d0dae3453-Paper.pdf.

David Buterez, Ioana Bica, Ifrah Tariq, Helena Andrés-Terré, and Pietro Liò. CellVGAE: an unsupervised scRNA-seq analysis workflow with graph attention networks. *Bioinformatics*, 38 (5):1277–1286, 12 2021. ISSN 1367-4803. doi: 10.1093/bioinformatics/btab804. URL https://doi.org/10.1093/bioinformatics/btab804.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

Nutan Chen, Alexej Klushyn, Francesco Ferroni, Justin Bayer, and Patrick Van Der Smagt. Learning flat latent manifolds with VAEs. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1587–1596. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20i.html.

Yi Cheng and Xiuli Ma. scGAC: a graph attentional architecture for clustering single-cell RNA-seq data. *Bioinformatics*, 38(8):2187–2193, 02 2022. ISSN 1367-4803. doi: 10.1093/bioinformatics/btac099. URL https://doi.org/10.1093/bioinformatics/btac099.

Jun Jin Choong, Xin Liu, and Tsuyoshi Murata. Optimizing variational graph autoencoder for community detection with dual optimization. *Entropy*, 22(2), 2020. ISSN 1099-4300. URL `https://www.mdpi.com/1099-4300/22/2/197`.

Marissa Connor, Gregory Canal, and Christopher Rozell. Variational autoencoder with learned latent structure. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 2359–2367. PMLR, 13–15 Apr 2021. URL `https://proceedings.mlr.press/v130/connor21a.html`.

Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. Adaptive graph encoder for attributed graph embedding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 976–985, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403140. URL `https://doi.org/10.1145/3394486.3403140`.

Fnu Devvrit, Aditya Sinha, Inderjit S Dhillon, and Prateek Jain. S3GC: Scalable self-supervised graph clustering. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=ldl2V3vLZ5`.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric, 2019. URL `http://arxiv.org/abs/1903.02428`. cite arxiv:1903.02428.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2): 298–305, 1973. URL `http://eudml.org/doc/12723`.

Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007. doi: 10.1073/pnas.0605965104. URL `https://www.pnas.org/doi/abs/10.1073/pnas.0605965104`.

Hongyang Gao and Shuiwang Ji. Graph u-nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2083–2092. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/gao19a.html`.

Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 855–864, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939754. URL `https://doi.org/10.1145/2939672.2939754`.

Roger Guimerà and Luís Amaral. Cartography of complex networks: Modules and universal roles. *Journal of statistical mechanics (Online)*, 2005:nihpa35573, 03 2005. doi: 10.1088/1742-5468/2005/02/P02001.

Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. Knowledge transfer for out-of-knowledge-base entities : A graph neural network approach. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 1802–1808, 2017. doi: 10.24963/ijcai.2017/250. URL `https://doi.org/10.24963/ijcai.2017/250`.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of International Conference on Machine Learning*, pp. 3451–3461. 2020.

Binyuan Hui, Pengfei Zhu, and Qinghua Hu. Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4215–4222, Apr. 2020. doi: 10.1609/aaai.v34i04.5843. URL `https://ojs.aaai.org/index.php/AAAI/article/view/5843`.

Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83:016107, Jan 2011. doi: 10.1103/PhysRevE.83.016107. URL `https://link.aps.org/doi/10.1103/PhysRevE.83.016107`.

Hyunsoo Kim and Haesun Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 23(12): 1495–1502, jun 2007. ISSN 1367-4803.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders, 2016a.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. arXiv, 2016b. doi: 10.48550/ARXIV.1609.02907. URL `https://arxiv.org/abs/1609.02907`.

Manoj Kumar, Anurag Sharma, and Sandeep Kumar. A unified framework for optimization-based graph coarsening. *Journal of Machine Learning Research*, 24(118):1–50, 2023. URL `http://jmlr.org/papers/v24/22-1085.html`.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *Proceedings of the 36th International Conference on Machine Learning*, 09–15 Jun 2019.

Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Statistical properties of community structure in large social and information networks. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pp. 695–704, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605580852. doi: 10.1145/1367497.1367591. URL `https://doi.org/10.1145/1367497.1367591`.

Yue Liu, Wenxuan Tu, Sihang Zhou, Xinwang Liu, Linxuan Song, Xihong Yang, and En Zhu. Deep graph clustering via dual correlation reduction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7603–7611, 2022.

Yue Liu, Xihong Yang, Sihang Zhou, Xinwang Liu, Siwei Wang, Ke Liang, Wenxuan Tu, and Liang Li. Simple contrastive graph clustering. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2023a. doi: 10.1109/TNNLS.2023.3271871.

Yue Liu, Xihong Yang, Sihang Zhou, Xinwang Liu, Zhen Wang, Ke Liang, Wenxuan Tu, Liang Li, Jingcan Duan, and Cancan Chen. Hard sample aware network for contrastive deep graph clustering. In *Proc. of AAAI*, 2023b.

Andreas Loukas. Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, 20(116):1–42, 2019. URL `http://jmlr.org/papers/v20/18-680.html`.

Costas Mavromatis and G. Karypis. Graph infoclust: Maximizing coarse-grain mutual information in graphs. In *PAKDD*, 2021.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. doi: 10.21105/joss.00861. URL `https://doi.org/10.21105/joss.00861`.

Anindya Mondal, Shashant R, Jhony H. Giraldo, Thierry Bouwmans, and Ananda S. Chowdhury. Moving object detection for event-based vision using graph spectral clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 876–884, October 2021.

Nairouz Mrabah, Mohamed Bouguessa, Mohamed Fawzi Touati, and Riadh Ksantini. Rethinking graph auto-encoder models for attributed graph clustering. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–15, 2022. doi: 10.1109/TKDE.2022.3220948.

M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, Jun 2004. doi: 10.1103/PhysRevE.69.066133. URL `https://link.aps.org/doi/10.1103/PhysRevE.69.066133`.

Mark E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103 23:8577–82, 2006.

V Nicosia, G Mangioni, V Carchiolo, and M Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(03):P03024, mar 2009. doi: 10.1088/1742-5468/2009/03/P03024. URL `https://dx.doi.org/10.1088/1742-5468/2009/03/P03024`.

Krzysztof Nowicki and Tom A. B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001. doi: 10.1198/016214501753208735. URL `https://doi.org/10.1198/016214501753208735`.

Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 2609–2615. International Joint Conferences on Artificial Intelligence Organization, 7 2018. doi: 10.24963/ijcai.2018/362. URL `https://doi.org/10.24963/ijcai.2018/362`.

J. Park, M. Lee, H. Chang, K. Lee, and J. Choi. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6518–6527, Los Alamitos, CA, USA, nov 2019. IEEE Computer Society. doi: 10.1109/ICCV.2019.00662. URL `https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00662`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '14. ACM, August 2014a. doi: 10.1145/2623330.2623732. URL `http://dx.doi.org/10.1145/2623330.2623732`.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pp. 701–710, New York, NY, USA, 2014b. Association for Computing Machinery. ISBN 9781450329569. doi: 10.1145/2623330.2623732. URL `https://doi.org/10.1145/2623330.2623732`.

Ketan Rajawat and Sandeep Kumar. Stochastic multidimensional scaling. *IEEE Transactions on Signal and Information Processing over Networks*, 3(2):360–375, 2017. doi: 10.1109/TSIPN.2017.2668145.

Guillaume Salha-Galvan, Johannes F Lutzeyer, George Dasoulas, Romain Hennequin, and Michalis Vazirgiannis. Modularity-aware graph autoencoders for joint community detection and link prediction. *Neural Networks*, 153:474–495, 2022.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93, Sep. 2008. doi: 10.1609/aimag.v29i3.2157. URL `https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2157`.

Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. doi: 10.1109/34.868688.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: Extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pp. 990–998, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605581934. doi: 10.1145/1401890.1402008. URL `https://doi.org/10.1145/1401890.1402008`.

Anton Tsitsulin, John Palowitch, Bryan Perozzi, and Emmanuel MÃ¼ller. Graph clustering with graph neural networks. *Journal of Machine Learning Research*, 24(127):1–21, 2023. URL http://jmlr.org/papers/v24/20-998.html.

Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep Graph Infomax. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rklz9iAcKQ.

Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. In Sarit Kraus (ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 3670–3676, United States of America, 2019. Association for the Advancement of Artificial Intelligence (AAAI). doi: 10.24963/ijcai.2019/509. URL https://ijcai19.org/,https://www.ijcai.org/proceedings/2019/. International Joint Conference on Artificial Intelligence 2019, IJCAI 2019 ; Conference date: 10-08-2019 Through 16-08-2019.

Yen-Chuen Wei and Chung-Kuan Cheng. Towards efficient hierarchical designs by ratio cut partitioning. In *1989 IEEE International Conference on Computer-Aided Design. Digest of Technical Papers*, pp. 298–301, 1989. doi: 10.1109/ICCAD.1989.76957.

Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/e77dbaf6759253c7c6d0efc5690369c7-Paper.pdf.

Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. Attributed graph clustering via adaptive graph convolution. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pp. 4327–4333. AAAI Press, 2019. ISBN 9780999241141.

Han Zhao, Xu Yang, Zhenru Wang, Erkun Yang, and Cheng Deng. Graph debiased contrastive learning with joint representation clustering. In Zhi-Hua Zhou (ed.), *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pp. 3434–3440. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/473. URL https://doi.org/10.24963/ijcai.2021/473. Main Track.

A Proof of Lemma 2

When $\tilde{X}$ is kept constant, the optimization problem 6 gets reduced to:

$$\min_{C} f(C) = \operatorname{tr}(\tilde{X}^T C^T \Theta C \tilde{X}) + \frac{\alpha}{2} \left\| C\tilde{X} - X \right\|_F^2 - \frac{\beta}{2e} \operatorname{tr}(C^T B C) - \gamma \log\det(C^T \Theta C + J) + \frac{\lambda}{2} \left\| C^T \right\|_{1,2}^2 \tag{13}$$

$$\text{subject to } \mathcal{C} \in \mathcal{S}_c 1 \text{ where, } J = \frac{1}{k}\mathbf{1}_{k\times k} \tag{14}$$

The term $\operatorname{tr}(\tilde{X}^T C^T \Theta C \tilde{X})$ is convex function in C. This result can be derived easily using Cholesky Decomposition on the positive semi-definite matrix Θ (i.e. $\Theta = L^T L$):

$$\operatorname{tr}(\tilde{X}^T C^T \Theta C \tilde{X}) = \operatorname{tr}(\tilde{X}^T C^T L^T L C \tilde{X}) = \operatorname{tr}((LC\tilde{X})^T LC\tilde{X}) = \left\| LC\tilde{X} \right\|_F^2 \tag{15}$$

Frobenius norm is a convex function, and the simplified expression is linear in C. Hence we can deduce that the tr(.) term is convex in C. The terms $\left\| C\tilde{X} - X \right\|_F^2$ and $\left\| C^T \right\|_{1,2}^2$ are convex because Frobenius norm and $l_{1,2}$ norm are convex in C.

Next, for the modularity term we show

$$\operatorname{tr}(C^T B C) = \operatorname{tr}(C^T B^{\frac{1}{2}} B^{\frac{1}{2}} C) = \operatorname{tr}(C^T B^{T^{\frac{1}{2}}} B^{\frac{1}{2}} C) = \left\| B^{\frac{1}{2}} C \right\|_F^2 \tag{16}$$

Hence, this is term is also convex in C.

For proving the convexity of $-\log\det(C^T \Theta C + J)$ we restrict function to a line. We define a function g:

$$g(t) = f(z + tu) \text{where, } t \in dom(g), z \in dom(f), u \in \mathbb{R}^n. \tag{17}$$

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $g : \mathbb{R} \to \mathbb{R}$ is convex.

The graph Laplacian matrix of the coarsened graph (Θ_c) is symmetric and positive semi-definite having a rank of k-1. To convert Θ_c to positive definite matrix, we add a rank 1 matrix $J = \frac{1}{k}\mathbf{1}_{k\times k}$. ($\Theta_c + J = L^T L$)

$$f(L) = -\log\det(C^T \Theta C + J) = -\log\det(L^T L) \tag{18}$$

Now substituting L = Z + tU in the above equation.

$$g(t) = -\log\det((Z + tU)^T (Z + tU)) \tag{19}$$
$$= -\log\det(Z^T Z + t(Z^T U + U^T Z) t^2 U^T U) \tag{20}$$
$$= -\log\det(Z^T (I + t(UZ^{-1} + (UZ^{-1})^T) + t^2 (Z^{-1})^T U^T U Z^{-1}) Z) \tag{21}$$
$$\text{substituting } P = VZ^{-1} \tag{22}$$
$$= -(\log\det(Z^T Z) + \log\det(I + t(P + P^T) + t^2 P^T P)) \tag{23}$$
$$\text{Eigenvalue decomposition of} P = Q\Lambda Q^T \text{and} QQ^T = I \tag{24}$$
$$= -(\log\det(Z^T Z) + \log\det(QQ^T + 2tQ\Lambda Q^T + t^2 Q\Lambda^2 Q^T)) \tag{25}$$
$$= -(\log\det(Z^T Z) + \log\det(Q(I + 2t\Lambda + t^2\Lambda^2)Q^T)) \tag{26}$$
$$= -\log\det(Z^T Z) - \sum_{i=1}^{n} \log(1 + 2t\lambda_i + t^2\lambda^2) \tag{27}$$

Finding double derivative of $g(t)$:

$$g"(t) = \sum_{i=1}^{n} \frac{2\lambda_i^2(1+t\lambda_i)^2}{(1+2t\lambda_i+t^2\lambda_i^2)^2} \tag{28}$$

Since $g"(t) \geq 0 \forall\, t \in \mathbb{R}$, $g(t)$ is a convex function in t. This implies $f(L)$ is convex in L. We know that, $C^T\Theta C + J = L^T L$ so,

$$L = \Theta^{\frac{1}{2}}C + \frac{1}{\sqrt{kp}}\mathbf{1}_{p\times k} \tag{29}$$

Since L is linear in C and $f(L)$ is convex in L, $-\log\det(C^T\Theta C + J)$ is convex in C.

B Optimal Solution of Optimization Objective

We first show that the function $f(C)$ is $L - Lipschitz$ continuous gradient function with $L = \max(L_1, L_2, L_3, L_4, L_5)$, where $L_1, L_2, L_3, L_4, andL_5$ are the Lipschitz constants of tr$(\tilde{X}^T C^T\Theta C\tilde{X})$, $\frac{\alpha}{2}\left\|C\tilde{X} - X\right\|_F^2$, $-\frac{\beta}{2e}\text{tr}(C^TBC)$, $-\gamma\log\det(C^T\Theta C + J), and \frac{\lambda}{2}\left\|C^T\right\|_{1,2}^2$.

For the tr$(\tilde{X}^T C^T\Theta C\tilde{X})$ term, we apply triangle inequality and employ the property of the norm of the trace operator: $||tr|| = \underset{M\neq 0}{sup}\frac{|tr(M)|}{||M||_F}$.

$$\begin{aligned}
&|tr(\tilde{X}^T C_1^T\Theta C_1\tilde{X}) - tr(\tilde{X}^T C_2^T\Theta C_2\tilde{X})| && (30)\\
&= |tr(\tilde{X}^T C_1^T\Theta C_1\tilde{X}) - tr(\tilde{X}^T C_2^T\Theta C_1\tilde{X}) + tr(\tilde{X}^T C_2^T\Theta C_1\tilde{X}) - tr(\tilde{X}^T C_2^T\Theta C_2\tilde{X})| && (31)\\
&\leq |tr(\tilde{X}^T C_1^T\Theta C_1\tilde{X}) - tr(\tilde{X}^T C_2^T\Theta C_1\tilde{X})| + |tr(\tilde{X}^T C_2^T\Theta C_1\tilde{X}) - tr(\tilde{X}^T C_2^T\Theta C_2\tilde{X})| && (32)\\
&\leq ||tr||||\tilde{X}^T(C_1 - C_2)^T\Theta C_1\tilde{X}||_F + ||tr||||\tilde{X}^T C_2^T\Theta(C_1 - C_2)\tilde{X}||_F && (33)\\
&\leq ||tr||||\tilde{X}||_F||\Theta||||C_1 - C_2||_F(||C_1||_F + ||C_2||_F) \text{ (Frobenius Norm Property)} && (34)\\
&\leq 2\sqrt{p}||tr||||\tilde{X}||_F||\Theta||||C_1 - C_2||_F \ \ (||C_1||_F = ||C_2||_F = \sqrt{p}) && (35)\\
&\leq L_1||C_1 - C_2||_F && (36)
\end{aligned}$$

The second term is $\frac{\alpha}{2}\left\|C\tilde{X} - X\right\|_F^2$ can be written as:

$$\begin{aligned}
&\frac{\alpha}{2}tr((C\tilde{X} - X)^T(C\tilde{X} - X)) && (37)\\
&= \frac{\alpha}{2}tr(\tilde{X}^T C^T C\tilde{X} - X^T C\tilde{X} + X^T X - \tilde{X}^T C^T X) && (38)\\
&= \frac{\alpha}{2}(tr(\tilde{X}^T C^T C\tilde{X}) - tr(X^T C\tilde{X}) + tr(X^T X) - tr(\tilde{X}^T C^T X)) && (39)
\end{aligned}$$

All the terms except $tr(X^TX)$ (constant with respect to C) in obtained in the expression will follow similar proofs to tr$(\tilde{X}^T C^T\Theta C\tilde{X})$.

Next we consider the modularity term:

$$\begin{aligned}
&|tr(C_1^T BC_1) - tr(C_2^T BC_2)| && (40)\\
&= |tr(C_1^T BC_1) - tr(C_2^T BC_1) + tr(C_2^T BC_1) - tr(C_2^T BC_2)| && (41)\\
&\leq |tr(C_1^T BC_1) - tr(C_2^T BC_1)| + |tr(C_2^T BC_1) - tr(C_2^T BC_2)| && (42)\\
&\leq ||tr||||(C_1 - C_2)^T BC_1||_F + ||tr||||(C_1 - C_2)^T BC_2||_F && (43)\\
&\leq ||tr||||B||||C_1 - C_2||_F(||C_1||_F + ||C_2||_F) \text{ (Frobenius Norm Property)} && (44)\\
&\leq L_3||C_1 - C_2||_F && (45)
\end{aligned}$$

The Lipschitz constant for $-\gamma\log\det(C^T\Theta C + J)$ is linked to the smallest non-zero eigenvalue of the coarsened Laplacian matrix (Θ_c) and is bounded by $\frac{\delta}{(k-1)^2}$ (Rajawat & Kumar, 2017), where δ is the minimum non-zero weight of G_c.

Lastly, in the term $\frac{\lambda}{2}\left\|C^T\right\|_{1,2}^2$ we can write $|C|_{ij} = C_{ij} \geq 0$ because $C \in \mathcal{S}_c$ and contains non-negative numbers.

$$\left\|C^T\right\|_{1,2}^2 = \sum_{i=1} p(\sum_{j=1} kC_{ij})^2 \tag{46}$$

$$= \sum_{i=1}^{p}([C^T]_i\mathbf{1})^2 \tag{47}$$

$$= ||C\mathbf{1}||_F^2 \qquad = tr(\mathbf{1}^T C^T C\mathbf{1}) \tag{48}$$

$tr(\mathbf{1}^T C^T C\mathbf{1})$ can be proved to be $L_5 - Lipschitz$ like the modularity and Dirichlet energy (smoothness) terms. This concludes the proof.

The majorized problem for L-Lipschitz and differentiable functions can now be applied. The Lagrangian of the majorized problem, 8 is:

$$\mathcal{L}(C, \tilde{X}, \mu) = \frac{1}{2}C^T C - C^T A - \mu_1^T C + \mu_2^T \Big[\left\|C_1^T\right\|_2^2 - 1, \cdots, \left\|C_i^T\right\|_2^2 - 1, \cdots, \left\|C_p^T\right\|_2^2 - 1\Big]^T \tag{49}$$

where $\mu = \mu_1 || \mu_2$ are the dual variables and $A = \Big(C - \frac{1}{L}\nabla f(C)\Big)^+$

The corresponding KKT conditions (w.r.t C) are:

$$C - A - \mu_1 + 2[\mu_{2o}C_0^T, \cdots, \mu_{2i}C_i^T, \cdots, \mu_{2p}C_p^T] = 0 \tag{50}$$

$$\mu_2^T \Big[\left\|C_1^T\right\|_2^2 - 1, \cdots, \left\|C_i^T\right\|_2^2 - 1, \cdots, \left\|C_p^T\right\|_2^2 - 1, \Big]^T = 0 \tag{51}$$

$$\mu_1^T C = 0 \tag{52}$$

$$\mu_1 \geq 0 \tag{53}$$

$$\mu_2 \geq 0 C \qquad \geq 0 \tag{54}$$

$$\left\|[C^T]_i\right\|_2^2 \leq 1 \; \forall i \tag{55}$$

The optimal solution to these KKT conditions is:

$$C = \frac{(A)^+}{\sum_i \left\|[A^T]_i\right\|_2} \tag{56}$$

C Proof of convergence

In this section, we prove that the sequence $\{\mathbf{C}^{t+1}, \tilde{\mathbf{X}}^{t+1}\}$ generated by Algorithm 1 converges to the set of Karush–Kuhn–Tucker (KKT) optimality points for Problem 6.

The Lagrangian of Problem 6 comes out to be:

$$\mathcal{L}(C, \tilde{X}, \mu) = \text{tr}(\tilde{X}^T C^T \Theta C \tilde{X}) + \frac{\alpha}{2}\left\|C\tilde{X} - X\right\|_F^2 - \frac{\beta}{2e}\text{tr}(C^T BC) \tag{57}$$

$$- \gamma \log\det(C^T \Theta C + J) + \frac{\lambda}{2}\left\|C^T\right\|_{1,2}^2 - \mu_1^T C + \sum_i \mu_{2i}\Big[\left\|C_i^T\right\|_2^2 - 1\Big] \tag{58}$$

where $\mu = \mu_1 || \mu_2$ are the dual variables.

w.r.t. C, the KKT conditions are

$$2\Theta C\tilde{X}\tilde{X}^T + \alpha(C\tilde{X} - X)\tilde{X}^T - \frac{\beta}{e}BC - 2\gamma\Theta C(C^T\Theta C + J)^{-1} \tag{59}$$

$$+\lambda C\mathbf{1}_{k\times k} - \mu_1 + 2[\mu_{2_o}C_0^T, \cdots, \mu_{2_i}C_i^T, \cdots, \mu_{2_p}C_p^T] = 0$$

$$\mu_2^T\Big[\left\|C_1^T\right\|_2^2 - 1, \cdots, \left\|C_i^T\right\|_2^2 - 1, \cdots, \left\|C_p^T\right\|_2^2 - 1\Big]^T = 0 \tag{60}$$

$$\mu_1^T C = 0 \tag{61}$$

$$\mu_1 \geq 0 \tag{62}$$

$$\mu_2 \geq 0 \tag{63}$$

$$C \geq 0 \tag{64}$$

$$\left\|[C^T]_i\right\|_2^2 \leq 1\ \forall i \tag{65}$$

Now, $C^\infty \equiv \lim_{t\to\infty} C^t$ is found from Equation 9 as:

$$C^\infty = C^\infty + \frac{1}{L}\Bigg(2\Theta C^\infty\tilde{X}^\infty\tilde{X}^\infty + \alpha(C^\infty\tilde{X} - X)\tilde{X}^\infty - \frac{\beta}{e}BC^\infty \tag{66}$$

$$- 2\gamma\Theta C^\infty(C^{\infty T}\Theta C^\infty + J)^{-1} + \lambda C^\infty\mathbf{1}_{k\times k}\Bigg)$$

$$0 = 2\Theta C^\infty\tilde{X}^\infty\tilde{X}^\infty + \alpha(C^\infty\tilde{X} - X)\tilde{X}^\infty - \frac{\beta}{e}BC^\infty \tag{67}$$

$$- 2\gamma\Theta C^\infty(C^{\infty T}\Theta C^\infty + J)^{-1} + \lambda C^\infty\mathbf{1}_{k\times k}$$

So, for $\mu = 0$, C^∞ satisfies the KKT conditions.

w.r.t. $\tilde{X}$, the KKT conditions are:

$$2C^T\Theta C\tilde{X} + \alpha C^T(C\tilde{X} - X) = 0 \tag{68}$$

So, $X^\infty \equiv \lim_{t\to\infty} X^t$ found from Equation 11 will satisfy this as that equation is just a rearrangement of the KKT condition.

D Dataset Summaries and Metrics

Refer to 2 for the dataset summary.

Metrics. A pair of nodes are said to be in agreement if they belong to the same class and are assigned to the same cluster, or they belong to different classes and have been assigned different clusters. For a particular partitioning, ARI is the fraction of agreeable nodes in the graph. Accuracy is obtained by performing a maximum weight bipartite matching between clusters and labels. NMI measures the normalized similarity between the clusters and the labels, and is robust to class imbalances. Mutual Information between two labellings X and Y of the same data is defined as $MI(X,Y) = \sum_{i=1}^{|X|}\sum_{j=1}^{|Y|}\frac{|X_i\cap Y_i|}{N}\log\frac{N|X_i\cap Y_i|}{|X_i||Y_i|}$ and it is scaled between 0 to 1.

E Training Details

All experiments were run on an NVIDIA A100 GPU and Intel Xeon 2680 CPUs. We are usually running 4-16 experiments together to utilize resources (for example, in 40GB GPU memory, we can run 8 experiments on PubMed simultaneously). Again, the memory costs are more than dominated by the dataset. All experiments used the same environment running CentOS 7, Python 3.9.12, PyTorch 2.0, PyTorch Geometric 2.2.0.

F VGAE

In a VGAE, the encoder learns mean (μ) and variance (σ): $\mu = \text{GCN}_\mu(\mathbf{X}, \mathbf{A})$and $\log \sigma = \text{GCN}_\sigma(\mathbf{X}, \mathbf{A})$ By using the reparameterization trick, we get the distribution of the latent space as: $q(\mathbf{Z}|\mathbf{X}, \mathbf{A}) = \prod_{i=1}^{N} q(\mathbf{z_i}|\mathbf{X}, \mathbf{A}) = \prod_{i=1}^{N} \mathcal{N}(\mathbf{z_i}|\mu_i, \text{diag}(\sigma_i^2))$ A common choice for decoder is inner-product of the latent space with itself which giving us the reconstructed $\hat{A}$. $p(\hat{\mathbf{A}}|\mathbf{Z}) = \prod_{i=1}^{p} \prod_{j=1}^{p} p(\hat{A}_{ij}|z_i, z_j)$, with $p(\hat{A}_{ij} = 1|z_i, z_j) = \text{sigmoid}(z_i^T z_j)$

Name	p ($\lvert\mathbf{V}\rvert$)	n ($\lvert\mathbf{x}_i\rvert$)	e ($\lvert E\rvert$)	k (y)
Cora	2708	1433	5278	7
CiteSeer	3327	3703	4614	6
PubMed	19717	500	44325	3
Coauthor CS	18333	6805	163788	15
Coauthor Physics	34493	8415	495924	5
Amazon Photo	7650	745	238162	8
Amazon PC	13752	767	491722	10
ogbn-arxiv	169343	128	1166243	40
Brazil	131	0	1074	4
Europe	399	0	5995	4
USA	1190	0	13599	4

Table 2: Datasets summary.

G Plots of evolution of latent space for other datasets

Refer to Figure 5. We can see the clusters forming in the latent space of the VGAEs. In the case of Q-VGAE, since a GCN is used on this space, it can learn non-linearities and the latent space shows different structures (like a starfish in CiteSeer). Moreoever, these structures have their geometric centres at the origin and grow out from there. In contrast, for Q-GMM-VGAE, since a GMM is being learnt over the latent space, the samples are encouraged to be normally distributed in their independent clusters, all of which have different means and comparable standard deviations. So, we see multiple "blobs", which more or less follow a normal distribution. This plot effectively shows why a GMM-VGAE is more expressive than a VGAE.

H Attributed SBM theory and results

We validate the robustness and sensitivity of proposed methods to variance in the node features and graph structure. We are also generating features using a multivariate mixture generative model such that the node features of each block are sampled from normal distributions where the centers of clusters are vertices of a hypercube.

SBM. The Stochastic Block Model (SBM)(Nowicki & Snijders, 2001) is a generative model for graphs that incorporates probabilistic relationships between nodes based on their community assignments. In the basic SBM, a network with p nodes is divided into k communities or blocks denoted by C_i, where $i = 1, 2, \cdots, k$. The SBM defines a symmetric block probability matrix B with size $(k \times k)$, where each entry B_{ij} represents the probability of an edge between a node in community C_i and a node in community C_j. Diagonal entries of this matrix represents the probabilities of intra-cluster edges. This matrix B captures the intra- and inter-community connections and is assumed to be constant. $P(i \leftrightarrow j|C_i = a, C_j = b) = B_{ab}$ denotes the probability of an edge existing between nodes i and j when node i belongs to community a and node j belongs to community b. Using these probabilities, the SBM generates a network by independently sampling the presence or absence of an edge for each pair of nodes based on their community assignments and the block probability matrix B.

Degree Corrected SBM. DC-SBM(Karrer & Newman, 2011) takes an extra set of parameters θ_i controlling the expected degree of vertex i. Now, the probability of an edge between two nodes (using the same notation as above) becomes $\theta_i \theta_j B_{ab}$. This was introduced to handle the heterogeneity of real-world graphs.

ADC-SBM Generation. We make use of the `graph_tool` library to generate the DC-SBM adjacency matrix, with $p = 1000, k = 4$. To generate the B matrix, we follow the procedure in (Tsitsulin et al., 2023), by taking expected degree for each node $d = 20$ and expected sub-degree

$d_{out} = 2$. This gives us B as:

$$\begin{bmatrix} 18 & 2 & 2 & 2 \\ 2 & 18 & 2 & 2 \\ 2 & 2 & 18 & 2 \\ 2 & 2 & 2 & 18 \end{bmatrix}$$

Also, θ is generated by sampling a power-law distribution with exponent $\alpha = 2$. We constrain the generated vector to $d_{min} = 2$ and $d_{max} = 4$.

To generate features, we use the `make_classification` function in the `sklearn` library. We generate a 128-dimensional feature vector for each node, with no redundant channels. These belong to k_f groups, where k_f might not be equal to k. We test three scenarios: a) matched clusters ($k_f = k$) b) nested features ($k_f > k$) c) grouped features ($k_f < k$) as visualized in Figure 6. Note that for better visualization, `class_sep` was increased to 5 (however, the results are given with a value of 1, which is a harder problem).

Results. Our model is able to completely recover the ground truth labels (NMI/ARI/ACC = 1) under all the specified conditions.

I Results on very large datasets

	CoauthorCS			CoauthorPhysics			AmazonPhoto			AmazonPC			ogbn-arxiv		
Method	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**	**ACC ↑**	**NMI ↑**	**ARI ↑**
FGC	69.6	70.4	61.5	69.9	60.9	49.5	44.9	38.3	22.5	46.8	36.2	23.3	24.1	8.5	9.1
Q-FGC (Ours)	70.2	76.4	60.2	75.3	67.2	66.1	70.4	66.6	**58.6**	**62.4**	51	31.1	35.8	24.4	15.6
Q-GCN (Ours)	85.4	79.6	79.7	85.2	**72**	**81.6**	66.3	57.6	48.3	56.7	42.4	28.8	34.4	27.1	19.7
Q-VGAE (Ours)	**85.6**	**79.9**	**81.6**	**86.7**	69	77.7	69.0	59.4	49.0	62.3	45.7	**47.2**	**39.5**	30.4	**24.7**
Q-GMM-VGAE (Ours)	70.1	72.5	61.6	83.1	71.5	76.9	**76.8**	**67.1**	58.3	55.5	**56.4**	40	OOM	OOM	OOM
DMoN	68.8	69.1	57.5	45.4	56.7	50.3	61.0	63.3	55.4	45.4	49.3	47.0	25.0	**35.6**	12.7

Table 3: Comparison of methods on large attributed datasets.

J Implementation

The implementations for all the experiments can be found at `https://anonymous.4open.science/r/MAGC-8880/`.

We have extensively used the PyTorch(Paszke et al., 2019) and PyTorch Geometric(Fey & Lenssen, 2019) libraries in our implementations and would like to thank the authors and developers.

K Explanation on why VAE manifolds are curved

The latent space of a VAE is not constrained to be Euclidean. Connor et al. (2021) point out that the variational posterior is selected to be a multivariate Gaussian, and that the prior is modeled as a zero-mean isotropic normal distribution which encourages grouping of latent points around the origin. Works such as (Chen et al., 2020; Bogdanov & Shchur, 2021; Arvanitidis et al., 2018) make the VAE latent space to be Euclidean/Hyperbolic/Riemannian, and show good visuals. Moreover, it can be observed in our own work (Figure 3a and Appendix Figure 5a) that the latent manifolds are curved and so, are not suitable for conventional methods such as k-means clustering, which need Euclidean distance (3.2).

L Complexities of some graph clustering methods

GCN*-based* clustering methods. Such as AGC(Zhang et al., 2019) - $\mathcal{O}(p^2nt + ent^2)$, R-VGAE(Mrabah et al., 2022) - $\mathcal{O}(pk^2n + (p(n+k) + e(p+k))$, S3GC(Devvrit et al., 2022) - $\mathcal{O}(pn^2s)$ [s is average degree], HSAN(Liu et al., 2023b) - $\mathcal{O}(pBn)$ [they state it as $\mathcal{O}(B^2d)$ but it is only for 1 batch of size B and not the whole epoch], VGAECD-OPT(Choong et al., 2020) - $\mathcal{O}(p^2nD^L)$ [where D is the size of graph filter, l is the number of linear layers] etc.

M EVOLUTION OF DIFFERENT LOSS TERMS THROUGHOUT TRAINING

Each separate series has been normalized by its absolute minimum value to see convergence behavior on the same graph easily. Every series is decreasing/converging (except gamma, which represents sparsity regularization and remains almost constant). Thus, we can be assured that no terms are counteracting and hurting the performance. The legend is provided in the graph itself. This plot is on the Cora dataset. 7

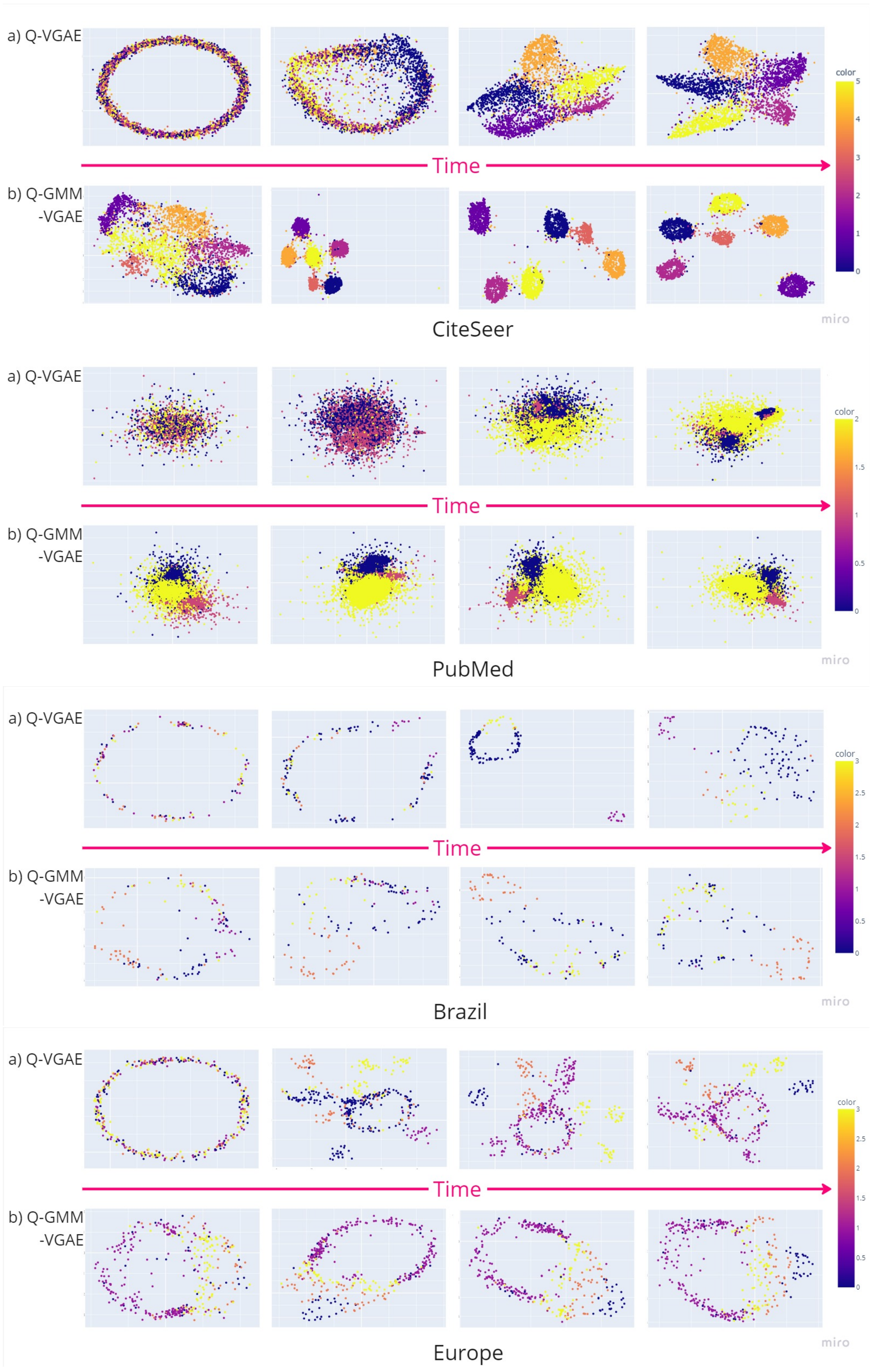
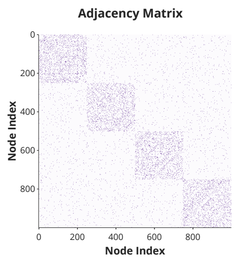
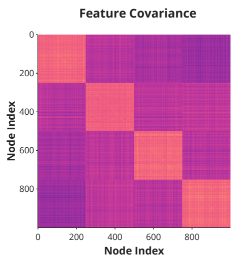


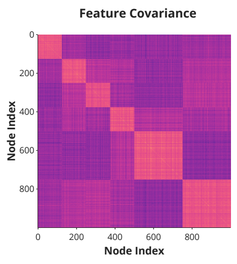
Figure 5: Plots of evolution of latent space for Q-VGAE and Q-GMM-VGAE methods for CiteSeer, PubMed, Brazil (Air Traffic) and Europe (Air Traffic) datasets.



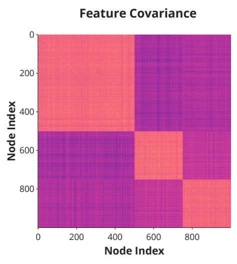
(a) Adjacency Matrix



(b) Feature Covariance Matrix
$k_f = k$



(c) Feature Covariance Matrix
$k_f > k$



(d) Feature Covariance Matrix
$k_f < k$

Figure 6: Visualization of the generated adjacency and feature covariance matrices for the ADC-SBM

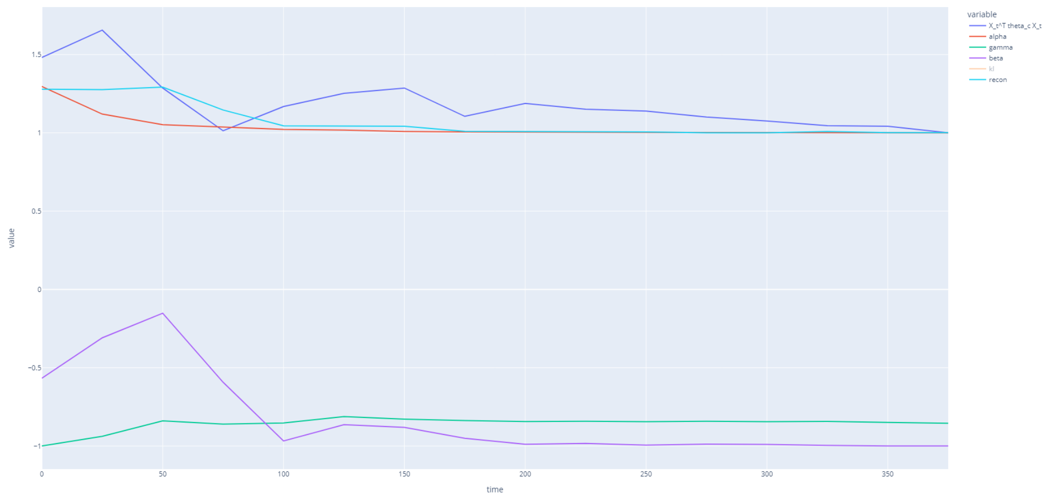


Figure 7