Generalized Laplacian Eigenmaps

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph contrastive learning attracts/disperses node representations for similar/dissimilar node pairs under some notion of similarity. It may be combined with a low-dimensional embedding of nodes to preserve intrinsic and structural properties of a graph. Some recent graph contrastive methods combine traditional graph embedding and negative sampling into one framework, which minimizes the trace difference between the within-class scatter matrix encapsulating the graph connectivity and the total scatter matrix encapsulating negative sampling. In this paper, we propose a more essential framework for graph embedding, called Generalized Laplacian EigeNmaps (GLEN), to learn graph representation by maximizing the rank difference between the total scatter matrix and the within-class scatter matrix, resulting in the minimum class separation guarantee. However, the rank difference minimization is an NP-hard problem. Herein, we replace the trace difference that corresponds to the difference of nuclear norms by the difference of logdet expressions, which we argue is a more accurate surrogate for the NP-hard rank difference than the trace difference. We show that the logdet loss can be interpreted as an upper bound of the Jensen-Bregman LogDet Divergence (JBLD), and the Affine-invariant Riemannian metric (AIRM) while enjoying a lesser computational burden. We show on popular benchmarks/backbones that GLEN offers favourable accuracy/scalability compared to state-of-the-art baselines.

1 Introduction

Laplacian Eigenmaps [4] and IsoMap [30] are popular graph embedding methods that minimise the dimensionality of data which exists on a low-dimensional manifold. The objective function in such models encourages node embeddings to lie near each other in the embedding space if they are close in the original space. However, classical methods focus solely on modeling the related node pairs and ignore modeling of unrelated node pairs.

In contrast, modern graph embedding models are often unified under the (sampled) Noise Contrastive Estimation (NCE) framework [24, 20] and extended to network embedding [29, 10, 37] and Graph Contrastive Learning (GCL) [31, 46, 11, 47, 45]. Most GCL methods do not incorporate graph information into the loss but follow the setting from computer vision, *i.e.*, they assume that randomly drawn pairs should be dissimilar and the original sample and its augmentations should be similar [33]. In contrast, Contrastive Laplacian Eigenmaps [45] propose a framework to learn (graph) neural networks with Laplacian eigenmaps utilizing graph information within a contrastive loss. However, their method minimizes the trace difference of Laplacians which follows the NCE framework.

In this paper, we analyze the relation among within-class, between-class and total scatter matrices under the rank inequality, and prove that, under a simple assumption, the distance between any dissimilar (negative) samples would be greater/equal than the inter-class distance between their corresponding class centres. Based on this condition, we derive GLEN, a reformulation of graph embedding into a rank difference problem, which is a more essential framework than other graph

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

embedding frameworks. Furthermore, we demonstrate how to optimize the rank difference problem with a difference of logdet expressions, a differentiable function which is easy to use within (graph) neural networks. Moreover, we provide theoretical considerations regarding the low-rank optimization and connection to the Riemannian manifold in order to interpret our approach.

In summary, our contributions are threefold:

i. We propose a rank-based condition connecting within-class, between-class and total scatter matrices under which we can guarantee the worst margin on the class separation. We propose a loss function, Generalized Laplacian EigenNaps (GLEN), that realizes this condition.

ii. However, as the rank difference problem is NP-hard, we turn it into a difference of logdet expressions to learn node embeddings, as opposed to the trace difference (an upper bound of the difference of logdet terms and a poorer surrogate) emerging in traditional graph embedding models.

iii. We study the distance between Symmetric Positive Definite (SPD) matrices realized by the logdet-based implementation of GLEN which is lower-bounded by Jesen-Bregman LogDet Divergence (JBLD) and the Affine-invariant Riemmanian Metric (AIRM). We also explain how GLEN connects to other graph embedding methods.

2 Related Works

Graph Embeddings. By assuming that the data lies on a low-dimensional manifold, graph embedding methods such as Laplacian Eigenmaps [4] and IsoMap [30] optimize low-dimensional data embeddings. These methods [6] construct a similarity graph by measuring the similarity of high-dimensional feature vectors and embed the nodes into a low-dimensional space. However, the similarity graph does not contain the structural graph information.

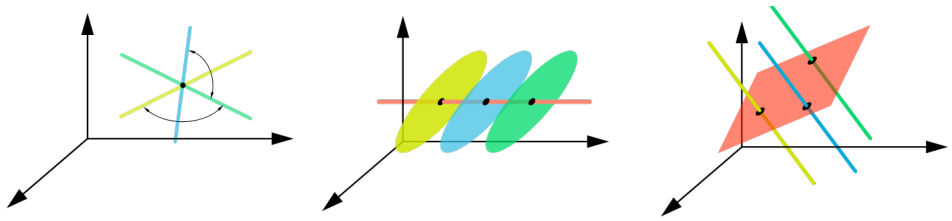
DeepWalk [27] uses truncated random walks to explore the graph structure and the skip-gram model for word embedding to determine the embedding vectors of nodes to address the above shortcomings. By setting the walk length to one and using negative sampling [23], LINE [29] explores a similar idea with an explicit objective function. To capture different graph neighborhoods, Node2Vec [9] interpolates between breadth- and depth-first sampling. Contrastive Laplacian Eigenmaps (COLES) unifies the traditional graph embedding and negative sampling by introducing a contrastive term that captures the graph structure, whereas the connectivity for negative sampling is arranged randomly. COLES solves the trace difference problem akin to traditional graph embedding models [36].

In this paper, we propose a more essential loss function for graph embedding. COLES is a special case of GLEN, *i.e.*, COLES uses the trace difference (difference of nuclear norms) which is a specific relaxation of generic GLEN that encapsulates a variety of contrastive properties.

Graph embedding techniques [36] provide a general framework for dimensionality reduction such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Locality Preserving Projections (LPP) [12]. All methods within this category can be considered as solving the same problem under varying assumptions, *i.e.*, maximising the intra- and inter-class separation by optimizing the trace difference, also used in metric learning [19].

However, such a family of objective functions is not motivated by the guarantee on the minimum margin between feature vectors from different categories. GLEN, in its purest NP-hard form, provides the minimum margin guarantee and can be realised by several formulations depending on trade-offs.

Unsupervised Representation Learning for Graph Neural Networks (GNN). Broadly, unsupervised GNN training can be reconstruction based, contrastive based or diffusion based. To train a graph encoder in an unsupervised manner, GCN [17] minimizes a reconstruction error which only considers the similarity matrix and ignores the dissimilarity information. At various scales of the graph, contrastive methods determine the positive and negative sets. For example, local-local CL and global-local CL strategies are highly popular. GraphSAGE [10] incorporates objectives inspired by DeepWalk [27], *i.e.*, the contrastive loss encourages neighbor nodes to have similar representations, while preserving dissimilarity between representations of disparate nodes. DGI [31], inspired by Deep InfoMax (DIM) [13], proposes an objective with global-local sampling strategy, which maximizes the Mutual Information (MI) between global and local graph embeddings. In contrast, Augmented Multiscale Deep InfoMax (AMDIM) [2] maximizes MI between multiple views



(a) $\mathrm{Rank}(S_b) = 0$ (b) $\mathrm{Rank}(S_b) = 1$ (c) $\mathrm{Rank}(s_b) = 2$

Figure 1: Three cases of our key condition $\mathrm{Rank}(S_t) = \mathrm{Rank}(S_w) + \mathrm{Rank}(S_b)$ (Condition 1). The red line and plane indicate the space of class centres. The black dots represent class centres. Other colored lines or ellipses represent the spaces of different categories. We only illustrate some non-exhaustive set of cases.

of data. MVRLG [11] contrasts encodings from first-order neighbors and a graph diffusion. MVRLG uses GCNs to learn node embeddings for different views.

However, such contrastive approaches often require thousands of epochs to converge and perform well. In addition, many contrastive losses have an exponential increase in memory overhead w.r.t. the number of nodes. In contrast, our method does not explicitly use the local-local setting but the total scatter matrix, and thus saves computational and storage cost.

Moreover, some linear GNNs, *e.g.*, SGC [35] and S^2GC [44], capture the neighborhood and increasingly larger neighborhoods of each node by diffusion, respectively. SGC and S^2GC have no projection layer, which results in embeddings of size equal to the input dimension. Our method is able to learn a projection layer in an unsupervised manner with a linear function or MultiLayer Perceptron (MLP) for these diffusion-based GNNs.

3 Preliminaries

Notations. Let $G = (V, E)$ be a simple, connected and undirected graph with $n = |V|$ nodes and $m = |E|$ edges. Let $i \in \{1, \cdots, n\}$ be the node index of G, and d_j be the degree of node j of G. Let $\mathbf{W}$ be the adjacency matrix, and $\mathbf{D}$ be the diagonal matrix containing degrees of nodes. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ denote the node feature matrix where each node v is associated with a feature vector $\mathbf{x}_v \in \mathbb{R}^d$. Let the normalized graph Laplacian matrix be defined as $\mathbf{L} = \mathbf{I} - \tilde{\mathbf{W}} \in \mathbb{S}_+^n$, a symmetric positive semi-definite matrix and $\tilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. Let $\mathbf{Z} = f_{\boldsymbol{\Theta}}(\mathbf{X}) \in \mathbb{R}^{n \times m}$ be a generalized node embedding, *i.e.*, $\mathbf{X}$ could be identity matrix (*e.g.*, no attributes for nodes), $f_{\boldsymbol{\Theta}}(\mathbf{X})$ could be GNN or a linear function with parameters $\boldsymbol{\Theta}$. Finally, scalars and vectors are denoted by lowercase regular and bold fonts, respectively. Matrices are denoted by uppercase bold fonts.

3.1 Scatter Matrices

Let us write the standard definitions of scatter matrices, including the total scatter $\mathbf{S}_t$, within-class $\mathbf{S}_w$, and between-class $\mathbf{S}_b$ matrices:

$$\mathbf{S}_t = \sum_{i=1}^{n} (\mathbf{z}_i - \bar{\mathbf{z}})(\mathbf{z}_i - \bar{\mathbf{z}})^\top = \mathbf{Z}(\mathbf{I} - \tilde{\mathbf{W}}_t)\mathbf{Z}^\top \quad \text{where} \quad \tilde{\mathbf{W}}_t = \frac{1}{N}\mathbf{e}\mathbf{e}^\top,$$

$$\mathbf{S}_w = \sum_{i=1}^{n} (\mathbf{z}_i - \boldsymbol{\mu}_{y_i})(\mathbf{z}_i - \boldsymbol{\mu}_{y_i})^\top = \mathbf{Z}(\mathbf{I} - \tilde{\mathbf{W}}_w)\mathbf{Z}^\top \quad \text{where} \quad \tilde{\mathbf{W}}_w = \sum_{c=1}^{C} \frac{1}{n_c}\mathbf{e}^c\mathbf{e}^{c\top},$$

$$\mathbf{S}_b = \sum_{c=1}^{C} n_c (\boldsymbol{\mu}_c - \bar{\mathbf{z}})(\boldsymbol{\mu}_c - \bar{\mathbf{z}})^\top. \tag{1}$$

Let $\mathbf{e}$ be an n-dimensional vector with all coefficients equal one, $\mathbf{I}$ be an identity matrix, $\mathbf{S}_t$ be the total scatter (covariance) matrix, and $\bar{\mathbf{z}}$ be the mean of all samples. Let $\boldsymbol{\mu}_{y_i}$ be the class center of the i-th sample and $\boldsymbol{\mu}_c$ be the c-th class center. Let the total number of categories be given by C, whereas

n_c be the number of samples for the c-th category. Let $\mathbf{e}^c$ be an n-dimensional vector where a given coefficient is equal one if its corresponding node is of class c, otherwise it is equal zero.

We note that both $\mathbf{S}_t$ and $\mathbf{S}_w$ can take a form akin to Laplacian eigenmaps such that $\tilde{\mathbf{W}}_t$ and $\tilde{\mathbf{W}}_w$ are the corresponding normalized adjacent matrices. Let us also define graph Laplacian matrices $\mathbf{L}_t = \mathbf{I} - \tilde{\mathbf{W}}_t$ and $\mathbf{L}_w = \mathbf{I} - \tilde{\mathbf{W}}_w$ which will be used in the sequel. Importantly, let us assume that a graph Laplacian matrix $\mathbf{L}$ containing graph links could be seen as a noisy version of $\mathbf{L}_w$ in which all nodes of a given class c connect under weight equal $1/c_c$.

Observe that $\mathbf{S}_t = \mathbf{S}_w + \mathbf{S}_b$. Thus, $\mathrm{Rank}(\mathbf{S}_t) \leq \mathrm{Rank}(\mathbf{S}_w) + \mathrm{Rank}(\mathbf{S}_b)$ due to the rank inequality. Here, we want to highlight the condition underpinning the subsequent motivation:

Condition 1. $\mathrm{Rank}(\mathbf{S}_t) = \mathrm{Rank}(\mathbf{S}_w) + \mathrm{Rank}(\mathbf{S}_b)$.

3.2 Motivation

Figure 1 shows three possible optimal solutions for Condition 1. The rank of between-class scatter matrix $\mathbf{S}_b$ for the whole data set is at most $C-1$ (where C is the number of classes). Since $\mathrm{Rank}(AB) \leq \min(\mathrm{Rank}(A), \mathrm{Rank}(B))$, we have $\mathrm{Rank}(\mathbf{S}_w^{-1}\mathbf{S}_b) \leq \mathrm{Rank}(\mathbf{S}_b) \leq C-1$. Rank equals the number of non-zero eigenvalues of a matrix which means that $\mathbf{S}_w^{-1}\mathbf{S}_b$ has at most $C-1$ non-zero eigenvalues (and corresponding eigenvectors). Condition 1 implies that $\mathrm{Rank}(\mathbf{S}_w^{-1}\mathbf{S}_b) = 0$ which leads us to the minimum class separation guarantee under that condition.

Theorem 1. *Let the feature dimension be larger than the class number (*i.e.*, $m >C$) and Condition 1 hold. Then, the minimum class separation is equal to the class centre distance. In other words, the distance between any two vectors $\mathbf{z}_i$ and $\mathbf{z}_j$ with labels y_i and y_j from different classes,* i.e.*, $y_i \neq y_j$, is greater/equal the distance between class centers $\boldsymbol{\mu}_{y_i}$ and $\boldsymbol{\mu}_{y_j}$:*

$$\|\boldsymbol{\mu}_{y_i} - \boldsymbol{\mu}_{y_j}\|_2 \leq \|\mathbf{z}_i - \mathbf{z}_j\|_2, \ \forall y_i \neq y_j, \ i,j \in \{1, \cdots, C\}. \tag{2}$$

Proof. As $\mathbf{S}_w$ is the orthogonal complement of $\mathbf{S}_b$, *i.e.*, $\mathbf{S}_w^{-1}\mathbf{S}_b = \mathbf{0}$, $\mathbf{S}_w + \mathbf{S}_b = \mathbf{U}\Sigma\mathbf{U}^\top$, $\mathbf{S}_w = \mathbf{U}_{1:k}\Sigma_{1:k}\mathbf{U}_{1:k}^\top$ and $\mathbf{S}_b = \mathbf{U}_{k+1:m}\Sigma_{k+1:m}\mathbf{U}_{k+1:m}^\top$ where $1 \leq k < m$. Let $\mathbf{z}_i = \boldsymbol{\mu}_{y_i} + \mathbf{U}^\top\boldsymbol{\epsilon}_i$ where $\boldsymbol{\epsilon}_i$ is the representation under the basis $\mathbf{U}$ and $\boldsymbol{\epsilon}_{(k+1:m),i} = \mathbf{0}$ because only top k components $1:k$ represent $\mathbf{S}_w$. Thus orthogonal projection $\mathbf{U}_{k+1:m}$ fulfills $\|\mathbf{U}_{k+1:m}(\mathbf{z}_i - \mathbf{z}_j)\|_2 \leq \|\mathbf{z}_i - \mathbf{z}_j\|_2$. Moreover, $\mathbf{U}_{k+1:m}(\mathbf{z}_i - \boldsymbol{\mu}_{y_i}) = \mathbf{U}_{k+1:m}(\mathbf{U}^\top\epsilon_i) = 0$. That is, all $\{\mathbf{z}_i : y_i = c\}$ are projected onto the mean $\boldsymbol{\mu}_c$ Thus, the inequality in Eq. 2 holds. □

Theorem 2 is attractive as it guarantees the worst inter-class distance (no known graph embedding methods that maximize inter-class distances and minimize intra-class distances provide such guarantees). Figure 1 shows possible cases that meet Condition 1. However, Figure 1a shows the case for which the class centers collapse to a single point and thus the inter-class distance equals zero (collapse of feature space). Figures 1b and 1c show two other valid cases.

4 Methodology

Condition 1 points to a promising research direction in learning discriminative feature spaces. However, optimizing over the rank is NP-hard and non-differentiable. In what follows, we provide the formulation of Generalized Laplacian EigeNmaps (GLEN) and its relaxation, which is differentiable.

4.1 Generalized Laplacian Eigenmaps

As solving Condition 1 is NP-hard, we propose a relaxation where $\mathrm{Rank}(\mathbf{S}_t)$ is encouraged to be as large as possible (bounded by the feature dimension m). On the contrary, if $\mathrm{Rank}(\mathbf{S}_t) \approx \mathrm{Rank}(\mathbf{S}_b)$ then the small $\mathrm{Rank}(\mathbf{S}_w)$ limits the feature diversity. In the extreme case, if $\mathrm{Rank}(\mathbf{S}_w) = 0$, the feature representation collapses. Larger $0 < \mathrm{Rank}(\mathbf{S}_b) \leq C-1$ improves the inter-class diversity.

In this paper, we propose a new **Generalized Laplacian EigeNmaps (GLEN)** framework for unsupervised network embedding. In the most general form, GLEN maximizes the difference of rank terms:

$$\boldsymbol{\Theta}^* = \arg\max_{\boldsymbol{\Theta}} \ \mathrm{Rank}\left(\mathbf{S}_t\left(f_{\boldsymbol{\Theta}}(\mathbf{X})\right)\right) - \mathrm{Rank}\left(\mathbf{S}_w\left(f_{\boldsymbol{\Theta}}(\mathbf{X})\right)\right). \tag{3}$$

However, the general matrix Rank Minimization Problem (RMP) [8] is NP-hard and the difference of rank terms in Eq. 3 is possibly a harder problem to solve. Thus, we relax this problem by the difference of logdet terms that serve as a surrogate of the NP-hard problem. We define:

$$\delta_{rf}(\mathbf{S}_t, \mathbf{S}_w; \alpha) = \log\det(\mathbf{I} + \alpha\mathbf{S}_t) - \lambda \log\det(\mathbf{I} + \alpha\mathbf{S}_w), \tag{4}$$

where $\lambda \geq 0$ controls the impact of $\log\det(\mathbf{S}_w)$ term. If $\lambda = 0$, δ_{rf} encourages $\mathrm{Rank}(f_{\boldsymbol{\Theta}}(\mathbf{X})) = m$. Moreover, let $\mathbf{S}_t = f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_t f_{\boldsymbol{\Theta}}(\mathbf{X})$ and $\mathbf{S}_w = f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_w f_{\boldsymbol{\Theta}}(\mathbf{X})$ which yields:

$$\boldsymbol{\Theta}^* = \arg\max_{\boldsymbol{\Theta}} \ \log\det\left(\mathbf{I} + \alpha f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_t f_{\boldsymbol{\Theta}}(\mathbf{X})\right) - \log\det\left(\mathbf{I} + \alpha f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_w f_{\boldsymbol{\Theta}}(\mathbf{X})\right), \tag{5}$$

where $\mathbf{I}$ ensures $\mathbf{I} + \alpha f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L} f_{\boldsymbol{\Theta}}(\mathbf{X}) > 0$ as $f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L} f_{\boldsymbol{\Theta}}(\mathbf{X})$ may be semi-SPD matrix leading to $\det(f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L} f_{\boldsymbol{\Theta}}(\mathbf{X})) = 0$. Thus, we use $\log\det(\mathbf{I} + \alpha\mathbf{Y})$ as a smooth surrogate for $\mathrm{Rank}(\mathbf{Y})$.

Proposition 1. *Let $\sigma(\mathbf{X})$ be the vector of eigenvalues of SPD matrix $\mathbf{X}$, and $\mathrm{Eig}(\mathbf{X})$ be a diagonal matrix with $\sigma(\mathbf{X})$ as its diagonal. Let $\mathbf{X}, \mathbf{X}' \in \mathbb{P}_m$ (the set of all SPD matrices of size $m \times m$), and $\alpha > 0$. Then, $\delta_{rf}(\mathbf{X}, \mathbf{X}'; \alpha) = \delta_{rf}(\mathrm{Eig}(\mathbf{X}), \mathrm{Eig}(\mathbf{X}'); \alpha)$,* i.e., *$\delta_{rf}(\cdot)$ depends on eigenvalues rather than eigenvectors of $\mathbf{X}$ and $\mathbf{X}'$.*

5 Theoretical Analysis

Below, we demonstrate the comparison between our approach and other methods by looking at (i) the low-rank optimization and (ii) the non-Euclidean distances between SPD matrices.

5.1 Nuclear Norm *vs*. Logdet for Rank Minimization

Claim 1. *COLES [45] is a convex relaxation (using the nuclear norm) of the rank difference in Eq. 3:*

$$\boldsymbol{\Theta}^* = \arg\max_{\boldsymbol{\Theta}} \mathrm{Tr}\left(f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_t f_{\boldsymbol{\Theta}}(\mathbf{X})\right) - \lambda\, \mathrm{Tr}\left(f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_w f_{\boldsymbol{\Theta}}(\mathbf{X})\right) \quad \textit{s.t.} \ \ \Omega(f_{\boldsymbol{\Theta}}(\mathbf{X})) = B, \tag{6}$$

where $\mathrm{Tr}\left(f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_t f_{\boldsymbol{\Theta}}(\mathbf{X})\right) = \|\mathbf{S}_t\|_*$ *and* $\mathrm{Tr}\left(f_{\boldsymbol{\Theta}}(\mathbf{X})^\top \mathbf{L}_w f_{\boldsymbol{\Theta}}(\mathbf{X})\right) = \|\mathbf{S}_w\|_*$.

The nuclear norm $\|\cdot\|_*$ can be regarded as the ℓ_1 norm over singular values. As the ℓ_1 norm induces sparsity, the nuclear norm encourages sparse singular values leading to low-rank solutions. If $f_{\boldsymbol{\Theta}}(\mathbf{X})^\top f_{\boldsymbol{\Theta}}(\mathbf{X})$ is restricted to be diagonal, $\|f_{\boldsymbol{\Theta}}(\mathbf{X})^\top f_{\boldsymbol{\Theta}}(\mathbf{X})\|_* = \|\mathrm{Diag}\left(f_{\boldsymbol{\Theta}}(\mathbf{X})^\top f_{\boldsymbol{\Theta}}(\mathbf{X})\right)\|_1$ and the nuclear norm surrogate for the rank minimization reduces to the ℓ_1 norm surrogate for the cardinality (rank) minimization. However, for the m-dimensional embedding, the solution of trace difference lies on a subspace of dimension less than $m-1$ [4]. Thus, the constraint $\Omega(f_{\boldsymbol{\Theta}}(\mathbf{X})) = B$ prevents the dimensional collapse, *i.e.*, $f_{\boldsymbol{\Theta}}(\mathbf{X})^\top f_{\boldsymbol{\Theta}}(\mathbf{X}) = \mathbf{I}$.

Compared with the trace-based relaxation, logdet is more suitable for cardinality minimization as it is less sensitive to large singular values. Also, the difference of logdet terms does not require decorrelation of features to prevent the dimensional collapse. We discuss this matter in the supplementary material. In our case, the difference of logdet terms is always bounded by the difference of trace terms as follows.

Proposition 2. *Given an embedding matrix $f_{\boldsymbol{\Theta}}(\mathbf{X}) \in \mathbb{R}^{n \times m}$, a fixed small constant $\alpha > 0$, we have the following inequality:*

$$\log\det\left(\mathbf{I} + \alpha\mathbf{S}_t\right) - \log\det\left(\mathbf{I} + \alpha\mathbf{S}_w\right) < \alpha\, \mathrm{Tr}(\mathbf{S}_t - \mathbf{S}_w). \tag{7}$$

Proof.

$$\begin{aligned} &\log\det\left(\mathbf{I} + \alpha\mathbf{S}_t\right) - \log\det\left(\mathbf{I} + \alpha\mathbf{S}_w\right) = \log\det\left(\mathbf{I} + \alpha\,\mathrm{Eig}(\mathbf{S}_t)\right) - \log\det\left(\mathbf{I} + \alpha\,\mathrm{Eig}(\mathbf{S}_w)\right) \\ &= \mathrm{Tr}\left(\log(\mathbf{I} + \alpha\,\mathrm{Eig}(\mathbf{S}_t) - \log(\mathbf{I} + \alpha\,\mathrm{Eig}(\mathbf{S}_w)\right) < \alpha\, \mathrm{Tr}(\mathbf{S}_t - \mathbf{S}_w). \end{aligned} \tag{8}$$

□

Proposition 2 is important because $\mathrm{Rank}(\mathbf{Y}) \leq \log\det(\mathbf{Y}) \leq \mathrm{Tr}(\mathbf{Y})$ [8].

5.2 Distance between Positive Definite Matrices

Below, we provide a perspective on non-Euclidean distances between SPD matrices to compare the proposed method with other graph embeddings, *e.g.*, Laplacian Eigenmaps [4] and COLES [45]. For the sake of comparison, we reformulate the Laplacian eigenmaps and COLES into forms in Prop. 3 and 4.

Proposition 3. *Laplacian Eigenmaps [4] equals to maximizing the Frobenius norm:*

$$\mathbf{\Theta}^* = \arg\max_{\mathbf{\Theta}} \|f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top - \mathbf{L}_w\|_F^2 \quad s.t. \quad f_{\mathbf{\Theta}}(\mathbf{X})^\top f_{\mathbf{\Theta}}(\mathbf{X}) = \mathbf{I}. \tag{9}$$

Proposition 4. *Contrastive Laplacian Eigenmaps [45] equals to maximizing the difference of Frobenius norm terms:*

$$\mathbf{\Theta}^* = \arg\max_{\mathbf{\Theta}} \|f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top - \mathbf{L}_w\|_F^2 - \|f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top - \mathbf{L}_t\|_F^2 \ s.t. f_{\mathbf{\Theta}}(\mathbf{X})^\top f_{\mathbf{\Theta}}(\mathbf{X}) = \mathbf{I}. \tag{10}$$

Proof.

$$\begin{aligned}\|f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top - \mathbf{L}\|_F^2 &= \mathrm{Tr}(f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top - 2f_{\mathbf{\Theta}}(\mathbf{X})\mathbf{L}f_{\mathbf{\Theta}}(\mathbf{X})^\top + \mathbf{L}^\top\mathbf{L}) \\ &= constant - 2\,\mathrm{Tr}(f_{\mathbf{\Theta}}(\mathbf{X})\mathbf{L}f_{\mathbf{\Theta}}(\mathbf{X})^\top) \geq 0.\end{aligned} \tag{11}$$

□

Note that Eq. 9 encourages the linear kernel matrix $f_{\mathbf{\Theta}}(\mathbf{X})f_{\mathbf{\Theta}}(\mathbf{X})^\top$ to be close to $\tilde{\mathbf{W}}_w$ while Eq. 10 encourage the linear kernel matrix to be far from the $\tilde{\mathbf{W}}_w$ at the same time.

Our loss follows the non-Euclidean geometry. In the following part, we demonstrate the relation among Eq. 4, the Affine-invariant Riemannian metric (AIRM), and Jessen-Bregman LogDet Divergence (JBLD).

Indeed, our loss function is bounded from both sides by AIRM and AIRM scaled by $\sqrt{m}$ respectively.

Proposition 5. *Let $\sigma(\mathbf{X})$ be the vector of eigenvalues of $\mathbf{X}$, for any semi-SPD matrix $\mathbf{S}_t, \mathbf{S}_w \in \mathbb{P}_m^*$ (asterisk * denotes semi-SPD set), we have:*

$$\begin{aligned}\|\log((\mathbf{I}+\mathbf{S}_t)^{-1/2}(\mathbf{I}+\mathbf{S}_w)(\mathbf{I}+\mathbf{S}_t)^{-1/2})\|_F &\leq \log\det(\mathbf{I}+\mathbf{S}_t) - \log\det(\mathbf{I}+\mathbf{S}_w) \\ &\leq \sqrt{m}\|\log((\mathbf{I}+\mathbf{S}_t)^{-1/2}(\mathbf{I}+\mathbf{S}_w)(\mathbf{I}+\mathbf{S}_t)^{-1/2})\|_F.\end{aligned} \tag{12}$$

Proof. Given $\mathbf{A} = \mathbf{I} + \mathbf{S}_t$ and $\mathbf{B} = \mathbf{I} + \mathbf{S}_w$, we have:

$$\begin{aligned}\log\det(\mathbf{A}) - \log\det(\mathbf{B}) &= \log(\det(\mathbf{A})\det(\mathbf{B}^{-1})) = \log(\det(\mathbf{A})\det(\mathbf{B}^{-1/2})\det(\mathbf{B}^{-1/2})) \\ &= \mathrm{Tr}\log(\mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2}).\end{aligned} \tag{13}$$

We have $\mathrm{Tr}(\mathbf{A}) = \|\sigma(\mathbf{A})\|_1$, $\|\mathbf{A}\|_F = \|\sigma(\mathbf{A})\|_2$ and $\|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_1 \leq \sqrt{m}\|\mathbf{x}\|_2$. □

Thus, Eq. 4 is trying to find a mapping function maximizing an approximation of AIRM distance between the total scatter matrix and the within-class matrix.

Definition 1. *Given any SPD matrix $\mathbf{A}, \mathbf{B} \in \mathbb{P}_m$, we define the regularized Jensen Bregman Logdet Divergence as follows:*

$$J(\mathbf{A}, \mathbf{B}; \alpha) = \log\det\left(\frac{\mathbf{I}+\alpha\mathbf{A}+\alpha\mathbf{B}}{2}\right) - \frac{1}{2}\big(\log\det(\mathbf{I}+\alpha\mathbf{A}) + \log\det(\mathbf{I}+\alpha\mathbf{B})\big) \ \textit{for} \ \mathbf{A}, \mathbf{B}, \alpha \geq 0, \tag{14}$$

where $\mathbf{I}$ can be interpreted as a regularization to prevent $\det(\mathbf{A}) = 0$ and $\log(\det(\mathbf{A})) = -\infty$.

Proposition 6. *For any $\mathbf{S}_w, \mathbf{S}_t \in \mathbb{P}_m$ and $\alpha \geq 0$, we have:*

$$\begin{aligned}\delta_{ld}(\mathbf{S}_t, \mathbf{S}_w; \alpha) &\geq J(\mathbf{S}_w, \mathbf{S}_b; \alpha) \\ &= \log\det\left(\frac{\mathbf{I}+\alpha\mathbf{S}_w+\alpha\mathbf{S}_b}{2}\right) - \frac{1}{2}\left(\log\det(\mathbf{I}+\alpha\mathbf{S}_w) + \log\det(\mathbf{I}+\alpha\mathbf{S}_b)\right),\end{aligned} \tag{15}$$

where the equality holds only if $\alpha = 0$ and then $\det(\mathbf{I}+\alpha\mathbf{S}_b) = 1$.

Proof. Based on Eq. 14, we notice JBLD is a lower bound of our loss:

$$
\begin{aligned}
J(\mathbf{S}_w, \mathbf{S}_b; \alpha) &= \log\det\left(\frac{\mathbf{I} + \alpha\mathbf{S}_w + \alpha\mathbf{S}_b}{2}\right) - \frac{1}{2}\left(\log\det(\mathbf{I} + \alpha\mathbf{S}_w) + \log\det(\mathbf{I} + \alpha\mathbf{S}_b)\right) \\
&= \frac{1}{2}\log\det\left(\mathbf{I} + \alpha\mathbf{S}_t\right) - \frac{1}{2}\left(\log\det(\mathbf{I} + \alpha\mathbf{S}_w) + \log\det(\mathbf{I} + \alpha\mathbf{S}_b)\right) \\
&\leq \frac{1}{2}(\log\det\left(\mathbf{I} + \alpha\mathbf{S}_t\right) - \log\det(\mathbf{I} + \alpha\mathbf{S}_w)).
\end{aligned}
\tag{16}
$$

Thus, Eq. 4 can be also interpreted as find a mapping function maximizing the regularized JBLD between the within-class matrix and the between-class matrix. □

6 Experiments

We evaluate GLEN (its relaxation) on transductive and inductive node classification tasks and node clustering. GLEN is compared to popular unsupervised, contrastive, and (semi-)supervised approaches. Except for the classifier, unsupervised models do not use labels. To learn similarity/dissimilarity, contrastive models employ the contrastive setting. Labels are used to train the projection layer and classifier in semi-supervised models. A fraction of nodes (*i.e.*, 5 or 20 per class) used for training are labeled for semi-supervised setting. A SoftMax classifier is used for (semi-)supervised models, while a logistic regression classifier is used for unsupervised and contrastive approaches.

Datasets. GLEN is evaluated on four citation networks: Cora, Citeseer, Pubmed, Cora Full [17, 5] for transductive setting. We also employ the large scale Ogbn-arxiv from OGB [14]. Finally, the Reddit [40] dataset is used in inductive setting. See supplementary material for details of datasets.

Metrics. As fixed data splits [38] often on transductive models benefit models that overfit, we average results over 50 random splits for each dataset. We evaluate performance for 5 and 20 samples per class. The inductive task is typically tested on graphs that are significantly larger. As in prior works, we use fixed data splits and publish Micro-F1 scores averaged across 10 runs.

Baseline models. We group baseline models into unsupervised, contrastive and (semi-)supervised methods, and implement them in the same framework/testbed. Contrastive methods include DeepWalk [27], GCN+SampledNCE developed as an alternative to GraphSAGE+SampledNCE [10], Graph2Gauss [5], SCE [41], DGI [31], GRACE [46], GCA [47], GraphCL [39] and COLES [45], which are our main competitors. Note that GRACE, GCA and GraphCL are based on multi-view and data augmentation, and GraphCL is mainly intended for graph classification. We do not study graph classification as it requires advanced node pooling with mixed- or high-order statistics [34, 18]. We

Table 1: Mean classification accuracy (%) and the standard dev. over 50 random splits. Numbers of labeled samples per class are in parentheses. The best accuracy per column is in bold. Models are organized into semi-supervised, contrastive and unsupervised groups. OOM means out of memory.

	Method	Cora		Citeseer		Pubmed		Cora Full	
		(5)	(20)	(5)	(20)	(5)	(20)	(5)	(20)
Semi-supervised	GCN	67.5±4.8	79.4±1.6	57.7±4.7	69.4±1.4	65.4±5.2	77.2±2.1	49.3±1.8	61.5±0.5
	GAT	71.2±3.5	79.6±1.5	54.9±5.0	69.1±1.5	65.5±4.6	75.4±2.3	43.9±1.5	56.9±0.6
	MixHop	67.9±5.7	80.0±1.4	54.5±4.3	67.1±2.0	64.4±5.6	75.7±2.7	47.5±1.5	61.0±0.7
Contrastive	DeepWalk	60.3±4.0	70.5±1.9	38.3±2.9	45.6±2.0	60.3±5.6	70.8±2.6	38.9±1.4	51.1±0.7
	GCN+SampledNCE	61.3±4.3	74.3±1.6	42.3±3.4	56.8±1.9	60.9±5.7	70.3±2.5	32.7±1.9	45.2±0.9
	SAGE+SampledNCE	65.0±3.5	73.8±1.5	48.0±3.5	56.5±1.6	64.1±6.1	74.6±1.9	35.0±1.4	43.6±0.6
	Graph2Gauss	72.7±2.0	76.2±1.1	60.7±3.5	65.7±1.5	67.6±3.9	74.1±2.1	38.9±1.3	49.3±0.5
	SCE	74.3±2.7	80.2±1.1	65.4±2.9	70.7±1.2	65.7±6.0	75.8±2.2	50.7±1.5	60.6±0.6
	DGI	72.9±4.0	78.1±1.8	65.7±3.6	71.1±1.1	65.3±5.7	73.9±2.3	50.5±1.4	58.4±0.6
	COLES-GCN	73.8±3.4	80.8±1.3	66.0±2.6	69.0±1.3	62.7±4.6	72.7±2.1	47.3±1.5	58.9±0.5
	COLES-GCN (Stiefel)	75.0±3.4	81.0±1.3	67.9±2.3	71.7±0.9	62.6±5.0	73.2±2.6	47.6±1.2	59.2±0.5
	COLES-S^{2}GC	76.5±2.6	81.5±1.2	67.5±2.2	71.3±1.0	66.0±5.2	77.4±1.9	51.2±1.4	61.8±0.5
	GLEN-GCN	77.5±2.6	82.7±1.2	67.6±2.6	72.0±0.9	68.7±5.7	78.2±2.4	52.7±1.5	62.0±0.5
	GLEN-S^{2}GC	**78.2±2.4**	**83.0±1.0**	**69.1±2.1**	**72.3±0.9**	**70.6±3.9**	**80.1±1.9**	**53.0±1.5**	**62.6±0.5**
Contrastive + Multiview	GraphCL	72.6±4.2	78.3±1.7	65.6±3.0	71.1±0.8	OOM	OOM	OOM	OOM
	GRACE	64.9±4.2	73.9±1.6	61.8±3.9	68.4±1.6	OOM	OOM	OOM	OOM
	GCA	61.5±4.9	75.8±1.9	43.2±3.6	55.7±1.9	OOM	OOM	OOM	OOM
Unsupervised	SGC	63.9±5.4	78.3±1.9	59.5±3.4	69.8±1.4	65.8±4.4	76.3±2.3	46.0±2.2	57.7±1.2
	S^{2}GC	71.4±4.4	81.3±1.2	60.3±4.0	69.5±1.2	67.6±4.2	73.3±2.0	41.8±1.7	60.0±0.5
	PCA-S^{2}GC	72.1±3.8	81.2±1.3	61.0±3.5	68.8±1.3	67.5±4.3	73.2±2.0	42.3±1.7	59.3±0.6
	RP-S^{2}GC	65.9±4.6	78.1±1.2	51.4±3.2	61.7±1.6	66.1±5.0	72.5±1.9	31.5±1.4	48.7±0.6

compare results with representative (semi-)supervised GCN [17], GAT [31] and MixHop [1] models. SGC and S^2GC are unsupervised spectral filter networks. They do not have any learnable parameters. COLES and GLEN could be regarded as dimension reduction techniques for SGC and S^2GC, thus we compare them to PCA-S^2GC and RP-S^2GC, which use PCA and random projections to obtain the projection layer. We set hyperparameters based on the settings described in prior papers.

6.1 Transductive Learning

In this section, we consider transductive learning where all nodes are available in the training process.

COLES *vs*. GLEN. Table 1 shows the performance of GLEN *vs*. COLES on two different backbones, *i.e*., GCN and S^2GC. On both backbones, GLEN shows non-trivial improvements on all four datasets. GLEN-S^2GC outperforms the COLES by up to 4.6%.

Contrastive Embedding Baselines *vs*. GLEN. Table 1 shows that GLEN-GCN and GLEN-S^2GC outperform unsupervised models. In particular, GLEN-GCN outperforms GCN+SampledNCE on all four datasets, which shows that GLEN has an advantage over the SampledNCE framework. In addition, GLEN-S^2GC outperforms the best contrastive baseline DGI by up to 3.4%. On Cora with 5 training samples, GLEN-S^2GC outperforms S^2GC by 6.8%. Finally, Table 3 shows that GLEN-S^2GC (small number of trainable parameters) outperforms other methods on the challenging Ogbn-arxiv.

Semi-supervised GNNs *vs*. GLEN. Table 1 shows that the contrastive GCN baselines perform worse than semi-supervised variants, especially when 20 labeled samples per class are available. In contrast, GLEN-GCN outperformed the semi-supervised GCN on Cora by 10% and 3.4% given 5 and 20 labeled samples per class. GLEN-GCN also outperforms GCN on Citeseer and Pubmed by 9.9% and 5.2% given 5 labeled samples per class. These results show the superiority of GLEN on four datasets when the number of samples per class is 5. Even for 20 labeled samples per class, GLEN-S^2GC outperforms the best semi-supervised baselines on all four datasets *e.g*., by 3.3% on Cora. Semi-supervised models (*e.g*., GAT and MixHop) are affected by the low number of labeled samples, which is consistent with [22]. The accuracy of GLEN-GCN and GLEN-S^2GC is unaffected.

Unsupervised GNNs *vs*. GLEN. SGC and S^2GC are unsupervised linear networks based on spectral filters which do not use labels (except for the classifier). As a dimension reduction method, GLEN helps both methods reduce the dimension and achieve discriminative features. Table 1 shows that GLEN-S^2GC outperforms RP-S^2GC and PCA-S^2GC under the same projection size. GLEN-S^2GC also outperforms the unsupervised S^2GC baseline (high-dimensional representation).

6.2 Inductive Learning

In inductive learning, models have no access to the test set, thus they need to generalize well to unseen samples. Table 2 shows that GLEN and COLES enjoy a significant performance gain (1% - 5% in Micro-F1 scores). GLEN (low memory footprint) outperforms SAGE-LSTM. In contrast, DGI on Reddit triggered out-of-memory error on Nvidia GTX 1080 GPU (94.0 Micro-F1 is from [31]).

Table 2: Test Micro F1 Score (%) averaged over 10 runs on Reddit. Results of other models are from original papers.

Setting	Model	Test F1
Supervised	SAGE-mean	95.0
	SAGE-LSTM	95.4
	SAGE-GCN	93.0
Contrastive	SAGE-mean	89.7
	SAGE-LSTM	90.7
	SAGE-GCN	90.8
	FastGCN	93.7
	DGI	94.0
	COLES-S^2GC	95.4
	GLEN-S^2GC	**95.6**

Table 3: Mean classification accuracy (%) and the standard dev. over 10 runs on Ogbn-arxiv. Results of other models are from original papers.

Method		Test Acc.	#Params
MLP		55.50±0.23	110,120
Node2Vec	[9]	70.07±0.13	21,818,792
GraphZoom	[7]	71.18±0.18	8,963,624
C&S	[15]	71.26±0.01	5,160
SAGE-mean	[10]	71.49±0.27	218,664
GCN	[17]	71.74±0.29	142,888
DeeperGCN	[21]	71.92±0.17	491,176
SIGN	[28]	71.95±0.11	3,566,128
FrameLet	[43]	71.97±0.12	1,633,183
S^2GC	[44]	72.01±0.25	110,120
COLES-S^2GC	[45]	72.48±0.25	110,120
GLEN-S^2GC		**72.67±0.26**	110,120

Table 4: The clustering performance on Cora, Citeseer and Pubmed.

Method	Input	Cora			Citeseer			Pubmed		
		Acc%	NMI%	F1%	Acc%	NMI%	F1%	Acc%	NMI%	F1%
k-means	Feature	34.65	16.73	25.42	38.49	17.02	30.47	57.32	29.12	57.35
Spectral-f	Feature	36.26	15.09	25.64	46.23	21.19	33.70	59.91	32.55	58.61
Spectral-g	Graph	34.19	19.49	30.17	25.91	11.84	29.48	39.74	3.46	51.97
DeepWalk	Graph	46.74	31.75	38.06	36.15	9.66	26.70	61.86	16.71	47.06
GAE	Both	53.25	40.69	41.97	41.26	18.34	29.13	64.08	22.97	49.26
VGAE	Both	55.95	38.45	41.50	44.38	22.71	31.88	65.48	25.09	50.95
ARGE	Both	64.00	44.90	61.90	57.30	35.00	54.60	59.12	23.17	58.41
ARVGE	Both	62.66	45.28	62.15	54.40	26.10	52.90	58.22	20.62	23.04
GCN	Both	59.05	43.06	59.38	45.97	20.08	45.57	61.88	25.48	60.70
SGC	Both	62.87	50.05	58.60	52.77	32.90	63.90	69.09	31.64	68.45
S^2GC	Both	68.96	54.22	65.43	69.11	42.87	64.65	68.18	31.82	67.81
COLES-GCN	Both	60.74	45.49	59.33	63.28	37.54	59.17	63.46	25.73	63.42
COLES-GCN (Stiefel)	Both	62.46	47.01	59.38	65.17	38.90	60.85	63.56	25.81	63.58
COLES-S^2GC	Both	69.70	55.35	63.06	69.20	44.41	64.70	68.76	33.42	68.12
GLEN-GCN	Both	69.27	55.34	62.14	68.25	42.98	64.10	64.64	30.08	64.12
GLEN-S^2GC	Both	**71.01**	**56.69**	**69.01**	**69.89**	**45.37**	**65.70**	**69.62**	**34.97**	**69.33**

6.3 Node Clustering

We compare GLEN-GCN and GLEN-S^2GC with three types of clustering methods:

i. Methods that use only node features *e.g.*, k-means and spectral clustering (spectral-f) construct a similarity matrix with the node features by a linear kernel.

ii. Structural clustering methods that only use the graph structure: spectral clustering (spectral-g) that takes the graph adjacency matrix as the similarity matrix, and DeepWalk [27].

iii. Attributed graph clustering methods that use node features and the graph: Graph Autoencoder (GAE), Graph Variational Autoencoder (VGAE) [17], Adversarially Regularized Graph Autoencoder (ARGE), Var. Graph Autoencoder (ARVGE) [26], SGC [35] , S^2GC [44], COLES [45].

We measure the performance by the clustering Accuracy (Acc), Normalized Mutual Information (NMI) and macro F1-score (F1). We run each method 10 times on Cora, CiteSeer and PubMed. We set the number of propagation steps to 8 for SGC, S^2GC, COLES-S^2GC and COLES-S^2GC following [42]. Table 4 shows that GLEN-S^2GC outperforms other methods in all cases, whereas GLEN-GCN outperforms COLES-GCN, COLES-GCN (Stiefel) and contrastive GCN on all datasets.

Scalability. GraphSAGE and DGI require neighbor sampling with redundant forward/backward steps (long runtime). In contrast, GLEN-S^2GC enjoys a simple implementation with low memory usage/low runtime. For graphs with over 100 thousands nodes and 10 millions edges (Reddit), GLEN runs fast on NVIDIA 1080 GPU. Even on larger graph benchmarks, GLEN is fast as it optimizes the total scatter and the within-class matrices whose size depends on embedding size rather than the node number. The runtime of GLEN-S^2GC is also favourable in comparison to multi-view augmentation-based GraphCL. Specifically, GLEN-S^2GC took 0.54s, 0.3s, 5.3s and 15.4s on Cora, Citeseer, Pubmed and Cora Full, respectively. GraphCL took 110.19s, 101.0s, $\geq$ 8h and $\geq$ 8h respectively. Although the logdet difference is somewhat slower than the trace difference in forward/backward propagation, it converges faster, thus enjoying a similar low runtime.

7 Conclusions

In this paper, we model contrastvie learning as a rank difference problem to approximate the condition that the rank of total scatter matrix should equal the sum of ranks of within-scatter and between-scatter matrices. We relax this NP-hard assumption with a differentiable difference of logdet terms. We also show two perspectives on GLEN and the existing methods based on the low-rank optimization and distance between SPD matrices. In low-rank optimization, we explain why the logdet difference is a better surrogate function to optimize rank difference compared to the trace difference. We also show that our solution encourages linear kernel of embeddings become the geometric mean between the total scatter matrix and the within-class matrix. Our GLEN works well with many backbones outperforming many unsupervised, contrastive and (semi-)supervised methods.

References

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning*, pages 21–29, 2019.

[2] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *arXiv preprint arXiv:1906.00910*, 2019.

[3] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, Joydeep Ghosh, and John Lafferty. Clustering with bregman divergences. *Journal of machine learning research*, 6(10), 2005.

[4] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[5] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017.

[6] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637, 2018.

[7] C Deng, Z Zhao, Y Wang, Z Zhang, and Z Feng. Graphzoom: A multi-level spectral approach for accurate and scalable graph embedding. In *International Conference on Learning Representations*, 2020.

[8] Maryam Fazel, Haitham Hindi, and Stephen P Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of the 2003 American Control Conference, 2003.*, volume 3, pages 2156–2162. IEEE, 2003.

[9] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *ACM SIGKDD international conference on Knowledge discovery and Data Mining*, pages 855–864, 2016.

[10] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[11] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pages 4116–4126, 2020.

[12] Xiaofei He and Partha Niyogi. Locality preserving projections. *Advances in Neural Information Processing Systems*, 16(16):153–160, 2004.

[13] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[14] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

[15] Qian Huang, Horace He, Abhay Singh, Ser-Nam Lim, and Austin Benson. Combining label propagation and simple models out-performs graph neural networks. In *International Conference on Learning Representations*, 2021.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[18] Piotr Koniusz and Hongguang Zhang. Power normalizations in fine-grained image, few-shot image and graph classification. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[19] Brian Kulis et al. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.

[20] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. *Advances in Neural Information Processing Systems*, 27:2177–2185, 2014.

[21] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. Deepergcn: All you need to train deeper gcns. *arXiv preprint arXiv:2006.07739*, 2020.

[22] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI Conference on Artificial Intelligence*, pages 3538–3545, 2018.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 2013.

[25] Frank Nielsen and Richard Nock. On the centroids of symmetrized bregman divergences. *arXiv preprint arXiv:0711.3242*, 2007.

[26] S Pan, R Hu, G Long, J Jiang, L Yao, and C Zhang. Adversarially regularized graph autoencoder for graph embedding. In *International Joint Conference on Artificial Intelligence*, 2018.

[27] 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*, pages 701–710, 2014.

[28] Emanuele Rossi, Fabrizio Frasca, Ben Chamberlain, Davide Eynard, Michael Bronstein, and Federico Monti. Sign: Scalable inception graph neural networks. *arXiv preprint arXiv:2004.11198*, 2020.

[29] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *International Conference on World Wide Web*, pages 1067–1077, 2015.

[30] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[31] Petar Velickovic, William Fedus, William L Hamilton, Pietro Lio, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

[32] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 1(1):396–413, 2020.

[33] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, volume 119, pages 9929–9939, 2020.

[34] Zhengyang Wang and Shuiwang Ji. Second-order pooling for graph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[35] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. Simplifying graph convolutional networks. *arXiv preprint arXiv:1902.07153*, 2019.

[36] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2007.

[37] Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. Understanding negative sampling in graph representation learning. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1666–1676, 2020.

[38] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*, pages 40–48, 2016.

[39] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

[40] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

[41] Shengzhong Zhang, Zengfeng Huang, Haicang Zhou, and Ziang Zhou. Sce: Scalable network embedding from sparsest cut. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 257–265, 2020.

[42] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. Attributed graph clustering via adaptive graph convolution. *arXiv preprint arXiv:1906.01210*, 2019.

[43] Xuebin Zheng, Bingxin Zhou, Junbin Gao, Yuguang Wang, Pietro Lió, Ming Li, and Guido Montufar. How framelets enhance graph neural networks. In Marina Meila and Tong Zhang, editors, *International Conference on Machine Learning*, volume 139, pages 12761–12771, 2021.

[44] Hao Zhu and Piotr Koniusz. Simple spectral graph convolution. In *International Conference on Learning Representations*, 2021.

[45] Hao Zhu, Ke Sun, and Peter Koniusz. Contrastive laplacian eigenmaps. *Advances in Neural Information Processing Systems*, 34, 2021.

[46] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

[47] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference*, pages 2069–2080, 2021.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

i. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

ii. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]

iii. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

iv. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

v. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]