

# Supplementary Materials: Gaussian Mutual Information Maximization for Efficient Graph Self-Supervised Learning: Bridging Contrastive-based to Decorrelation-based

Anonymous Authors

## 1 DERIVATIONS OF GAUSSIAN MUTUAL INFORMATION

The formal derivation relies on the following lemma, which can also be found in standard linear algebra textbooks and is not regarded as our contributions.

**Lemma 1.1.** For matrices  $\mathbf{A} \in \mathbb{R}^{N \times K}$  and  $\mathbf{B} \in \mathbb{R}^{K \times N}$ ,

$$\text{tr}(\mathbf{AB}) = \text{tr}(\mathbf{BA}), \quad (1)$$

where  $\text{tr}(\cdot)$  denotes the trace of a matrix.

PROOF.

$$\text{tr}(\mathbf{AB}) = \sum_{i=1}^N \sum_{j=1}^K A_{ij} \cdot B_{ji} = \sum_{j=1}^K \sum_{i=1}^N B_{ji} \cdot A_{ij} = \text{tr}(\mathbf{BA}).$$

□

The mutual information  $I_G(X; Y)$  can be expanded as follows:

$$\begin{aligned} I_G(X; Y) &= \int_X \int_Y p_{x,y}(X, Y) \ln \frac{p_{x,y}(X, Y)}{p_X(X) \cdot p_Y(Y)} dXdY \\ &= \int_X \int_Y p_{x,y}(X, Y) \ln p_{x,y}(X, Y) dXdY \\ &\quad - \int_X \int_Y p_{x,y}(X, Y) \ln p_X(X) dXdY \\ &\quad - \int_X \int_Y p_{x,y}(X, Y) \ln p_Y(Y) dXdY. \end{aligned} \quad (2)$$

In order to obtain the desired result, we will perform integration on the three terms in Eq. (2), respectively.

For a Gaussian variable  $X$  with mean  $\mu_X$  and covariance matrix  $\Sigma_X$ , its probability density function can be expressed as

$$p_X(X) = \frac{1}{\sqrt{(2\pi)^d \det(\Sigma_X)}} \exp \left( -\frac{1}{2} (X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X) \right). \quad (3)$$

Thus, it can be known that

$$\begin{aligned} & - \int_X \int_Y p_{x,y}(X, Y) \ln p_X(X) dXdY \\ &= - \int_X \int_Y p_{x,y}(X, Y) dY \ln p_X(X) dX \\ &= - \int_X p_X(X) \ln p_X(X) dX \\ &= - \int_X p_X(X) \left( \ln \frac{1}{\sqrt{(2\pi)^d \det(\Sigma_X)}} - \frac{1}{2} (X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X) \right) dX \\ &= \frac{1}{2} \int_X p_X(X) \ln((2\pi)^d \det(\Sigma_X)) dX \\ &\quad + \frac{1}{2} \int_X p_X(X) (X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X) dX \\ &= \frac{\ln((2\pi)^d \det(\Sigma_X))}{2} \int_X p_X(X) dX \\ &\quad + \frac{1}{2} \int_X p_X(X) (X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X) dX \\ &= \frac{1}{2} \ln \det(\Sigma_X) + \frac{d}{2} \ln(2\pi) + \frac{1}{2} \int_X p_X(X) (X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X) dX. \end{aligned} \quad (4)$$

We will deal specially with  $\int_X p_X(X) (X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X) dX$ . Actually,  $X - \mu_X$  is vector  $\in \mathbb{R}^d$  and  $(X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X)$  is a scalar value. If we regard  $X - \mu_X$  as a matrix  $\in \mathbb{R}^{d \times 1}$ ,  $(X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X)$  will be a matrix  $\in \mathbb{R}^{1 \times 1}$ . The original expression can be rephrased as  $\text{tr}((X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X))$ . Taking  $(X - \mu_X)^T$  as  $\mathbf{A}$  in Eq. (1) and  $\Sigma_X^{-1} (X - \mu_X)$  as  $\mathbf{B}$ , respectively, we can know that

$$\begin{aligned} & \int_X p_X(X) \text{tr}((X - \mu_X)^T \Sigma_X^{-1} (X - \mu_X)) dX \\ &= \int_X p_X(X) \text{tr}(\Sigma_X^{-1} (X - \mu_X) (X - \mu_X)^T) dX \\ &= \text{tr}(\Sigma_X^{-1} \int_X p_X(X) (X - \mu_X) (X - \mu_X)^T dX) \\ &= \text{tr}(\Sigma_X^{-1} \Sigma_X) \\ &= d. \end{aligned} \quad (5)$$

Plugging the result of Eq. (5) into Eq. (4), it can be concluded that

$$\begin{aligned} & - \int_X \int_Y p_{x,y}(X, Y) \ln p_X(X) dXdY \\ &= \frac{1}{2} \ln \det(\Sigma_X) + \frac{d}{2} \ln(2\pi) + \frac{d}{2} \\ &= \frac{1}{2} \ln \det(\Sigma_X) + \frac{d}{2} \ln(2\pi e). \end{aligned} \quad (6)$$

Symmetrically, it can be obtained that

$$-\int_X \int_Y p_{x,y}(X, Y) \ln p_y(Y) dX dY = \frac{1}{2} \ln \det(\Sigma_Y) + \frac{d}{2} \ln(2\pi e). \quad (7)$$

Similarly,

$$\begin{aligned} & \int_X \int_Y p_{x,y}(X, Y) \ln p_{x,y}(X, Y) dX dY \\ &= -\frac{1}{2} \ln \det(\Sigma_{X,Y}) - \frac{d+d}{2} \ln(2\pi e). \end{aligned} \quad (8)$$

Plugging Eq. (6), (7), and (8) into  $I_G(X; Y)$  in Eq. (2), it results in the following closed-form of Gaussian mutual information:

$$I_G(X; Y) = \frac{1}{2} \ln \frac{\det(\Sigma_X) \cdot \det(\Sigma_Y)}{\det(\Sigma_{X,Y})}. \quad (9)$$

Some contents about KL divergence and entropy of Gaussian variables can be referenced from other literature [3, 5, 28]. However, few studies have provided a complete derivation process of Gaussian mutual information. Moreover, many individuals in the community may be unfamiliar with the derivation and even the concept of Gaussian mutual information. Thus, for the self-completeness of this paper, we give the complete derivation here, which can also serve as a contribution of this paper.

## 2 DERIVATION AND ANALYSIS ABOUT GMIM-IC

According to Property 1 of the main text, the Gaussian mutual information  $I_G(X; Y) = \frac{1}{2} \ln \frac{\det(\Sigma_X) \cdot \det(\Sigma_Y)}{\det(\Sigma_{X,Y})}$  can be restated as

$$I_G(X; Y) = H_G(X) - H_G(X|Y), \quad (10)$$

where  $H_G(X) = -\int_X p_X(X) \ln p_X(X) dX$  is the entropy of  $X$  and  $H_G(X|Y)$  is its conditional entropy given  $Y$ . Based on Eq. (4) and (6),  $H_G(X) = \frac{1}{2} \ln \det(\Sigma_X) + \frac{d}{2} \ln(2\pi e)$ , that is,  $H_G(X) \propto \ln \det(\Sigma_X)$ . As mentioned in the main text, directly optimizing  $\ln \det(\Sigma_X)$  can lead to numerical instability. After adjusting the eigenvalues by applying shifting and scaling operations, we can obtain a feasible substitution  $\ln \det(\mathbf{I} + \eta \cdot \Sigma_X)$ . Therefore, maximizing  $\ln \det(\mathbf{I} + \eta \cdot \Sigma_X)$  can be equivalent to increasing the entropy  $H_G(X)$ .

As discussed in the main text, the conditional entropy  $H_G(X|Y)$  is minimized when the relationship between  $X$  and  $Y$  can be determined by a function. Considering the prior of network design, which has two shared branches, we expect that this function is an identity mapping. Concretely, this is realized by imposing identity constraint to Eq. (10). In our practice, the node representations from view  $A$  can be regarded as  $N$  empirical samples of  $X$  while those from view  $B$  are related to  $Y$ .

Taking all the above factors into consideration, we can derive an objective based on Eq. (10):

$$\mathcal{L}_{\text{GMIM-IC}}^A = \frac{1}{N} \sum_{v \in \mathcal{V}} \|\mathbf{h}_v^A - \mathbf{h}_v^B\|_2^2 - \gamma \cdot \ln \det(\mathbf{I} + \eta \cdot \Sigma_A), \quad (11)$$

where  $\gamma$  indicates a balancing factor. Symmetrically, we can obtain an objective  $\mathcal{L}_{\text{GMIM-IC}}^B$  corresponding to view  $B$ . Combining the two terms, it results in

$$\mathcal{L}_{\text{GMIM-IC}} = \frac{1}{N} \sum_{v \in \mathcal{V}} \|\mathbf{h}_v^A - \mathbf{h}_v^B\|_2^2 - \sum_{* \in \{A, B\}} \beta \cdot \ln \det(\mathbf{I} + \eta \cdot \Sigma_*). \quad (12)$$

Minimizing the objective  $\mathcal{L}_{\text{GMIM-IC}}$  is equivalent to maximizing Gaussian mutual information while imposing identity constraint across various views.

## 3 PROOFS AND DERIVATIONS IN SECTION 4 AND 5 OF THE MAIN TEXT

### 3.1 Proof of Lemma 4.1

For convenience, we restate Lemma 4.1:

**Lemma 4.1.** For a square matrix  $\mathbf{M}$ ,  $\det(\exp(\mathbf{M})) = \exp(\text{tr}(\mathbf{M}))$ . Replace  $\mathbf{M}$  with  $\ln(\mathbf{I} + \eta \cdot \Sigma_*)$ :

$$\ln \det(\mathbf{I} + \eta \cdot \Sigma_*) = \text{tr}(\ln(\mathbf{I} + \eta \cdot \Sigma_*)), \quad (13)$$

where  $* \in \{A, B\}$ . Applying Taylor expression to the logarithmic function in  $\text{tr}(\ln(\mathbf{I} + \eta \cdot \Sigma_*))$ , it can be known that

$$\ln \det(\mathbf{I} + \eta \cdot \Sigma_*) = \text{tr} \left( \sum_{k=1}^{+\infty} \frac{(-1)^{k+1}}{k} (\eta \cdot \Sigma_*)^k \right). \quad (14)$$

PROOF. Assuming  $\{\lambda'_1, \lambda'_2, \dots, \lambda'_d\}$  are  $d$  eigenvalues of the matrix  $\mathbf{M}$ ,  $\{e^{\lambda'_1}, e^{\lambda'_2}, \dots, e^{\lambda'_d}\}$  are  $d$  eigenvalues of the matrix  $\exp(\mathbf{M})$  accordingly. Thus,  $\det(\exp(\mathbf{M})) = \prod_{i=1}^d e^{\lambda'_i} = \exp(\sum_{i=1}^d \lambda'_i) = \exp(\text{tr}(\mathbf{M}))$ . Taking  $\mathbf{M} = \ln(\mathbf{I} + \eta \cdot \Sigma_*)$ , we can obtain  $\det(\mathbf{I} + \eta \cdot \Sigma_*) = \exp(\text{tr}(\ln(\mathbf{I} + \eta \cdot \Sigma_*)))$ , that is,  $\ln \det(\mathbf{I} + \eta \cdot \Sigma_*) = \text{tr}(\ln(\mathbf{I} + \eta \cdot \Sigma_*))$ .

Applying the Taylor expression  $\ln(1+x) = \sum_{k=1}^{\infty} \frac{(-1)^{k+1} \cdot x^k}{k}$ , we have

$$\begin{aligned} & \ln \det(\mathbf{I} + \eta \cdot \Sigma_*) \\ &= \text{tr}(\ln(\mathbf{I} + \eta \cdot \Sigma_*)) \\ &= \text{tr} \left( \sum_{k=1}^{+\infty} \frac{(-1)^{k+1}}{k} (\eta \cdot \Sigma_*)^k \right). \end{aligned} \quad (15)$$

□

Furthermore, we can obtain a second-order Taylor approximation:

$$\begin{aligned} & -\ln \det(\mathbf{I} + \eta \cdot \Sigma_*) \\ &\approx -\text{tr} \left( \sum_{k=1}^2 \frac{(-1)^{k+1}}{k} (\eta \cdot \Sigma_*)^k \right) \\ &= \frac{\eta^2}{2} \cdot \text{tr} \left( (\Sigma_*)^2 \right) - \eta \cdot \text{tr}(\Sigma_*) \\ &= \frac{\eta^2}{2} \cdot \|\Sigma_*\|_F^2 - \eta \cdot d \\ &= \frac{\eta^2}{2} \cdot \sum_{i=1}^d \sum_{j=1, j \neq i}^d (\Sigma_*^{ij})^2 + \frac{\eta^2}{2} \cdot d - \eta \cdot d. \end{aligned} \quad (16)$$

Ignoring constant terms,  $\sum_{i=1}^d \sum_{j=1, j \neq i}^d (\Sigma_*^{ij})^2$  is equivalent to a second-order Taylor expansion of  $-\ln \det(\mathbf{I} + \eta \cdot \Sigma_*)$ . Therefore, minimizing  $\sum_{i=1}^d \sum_{j=1, j \neq i}^d (\Sigma_*^{ij})^2$  has a similar effect to reducing  $-\ln \det(\mathbf{I} + \eta \cdot \Sigma_*)$ .

We have completed the entire deviation.

### 3.2 Proof of Proposition 4.2

The formal proof of Proposition 4.2 relies on the following lemma:

**Lemma 3.1.** For a real symmetric matrix  $\mathbf{A}$  whose eigenvalues are all 1, it must be the identity matrix.

PROOF. For a real symmetric matrix  $\mathbf{A}$ , it can be diagonalized by an orthogonal matrix, that is,  $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$  with the orthogonal matrix  $\mathbf{U}$  and the diagonal matrix  $\mathbf{D}$ . Since the eigenvalues of  $\mathbf{A}$  are all 1,  $\mathbf{D}$  is equal to an identity matrix  $\mathbf{I}$ . Thus,  $\mathbf{A} = \mathbf{U}\mathbf{U}^\top = \mathbf{I}$ .  $\square$

For convenience, we restate Proposition 4.2 here.

**Proposition 4.2.** *When  $\ln \det(\mathbf{I} + \eta \cdot \Sigma_*)$  or  $\ln \det(\Sigma_*)$  is maximized, the empirical covariance matrix  $\Sigma_*$  will converge to an identity matrix.*

PROOF. Assuming  $\{\lambda_1, \lambda_2, \dots, \lambda_d\}$  are  $d$  eigenvalues of the covariance matrix  $\Sigma_*$ ,  $\det(\mathbf{I} + \eta \cdot \Sigma_*) = \prod_{i=1}^d (1 + \eta \cdot \lambda_i)$ . Besides,  $\sum_{i=1}^d (1 + \eta \cdot \lambda_i) = \text{tr}(\mathbf{I} + \eta \cdot \Sigma_*) = d + \eta \cdot d$ . According to the AM-GM Inequality [9], it can be known that

$$\begin{aligned} & \det(\mathbf{I} + \eta \cdot \Sigma_*) \\ &= \prod_{i=1}^d (1 + \eta \cdot \lambda_i) \\ &\leq \left( \frac{1 + \eta \cdot \lambda_1 + 1 + \eta \cdot \lambda_2 + \dots + 1 + \eta \cdot \lambda_d}{d} \right)^d \\ &= (1 + \eta)^d. \end{aligned} \quad (17)$$

$\det(\mathbf{I} + \eta \cdot \Sigma_*)$  achieves the upper bound of  $(1 + \eta)^d$  when the eigenvalues  $\{\lambda_1, \dots, \lambda_d\}$  of  $\Sigma_*$  are all equal to 1. Similarly, applying the above derivation to  $\det(\Sigma_*)$ , we can easily conclude that  $\det(\Sigma_*)$  reaches a maximum value of 1 when all eigenvalues are equal to 1.

$\Sigma_* = \frac{1}{N} \mathbf{H}_*^\top \mathbf{H}_*$  is a real symmetric matrix. According to Lemma 3.1,  $\Sigma_*$  will converge to the identity matrix when its eigenvalues are all equal to 1. Thus, we conclude the proof.  $\square$

### 3.3 Proof of Property 2

**Property 2.** *For empirical covariance matrix  $\Sigma = \frac{1}{N} \mathbf{H}^\top \mathbf{H} \in \mathbb{R}^{d \times d}$  with batch-normalized representations  $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_N]^\top \in \mathbb{R}^{N \times d}$ , which has  $d$  eigenvalues  $[\lambda_1, \lambda_2, \dots, \lambda_d]$  corresponding to  $d$  eigenvectors  $[\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_d]$ , the variance of data  $\mathbf{H}$  along the  $k$ -th principal direction (that is, the direction of  $\mathbf{q}_k$ ) is numerically equal to  $\lambda_k$ .*

PROOF. For  $N$   $d$ -dimensional data points  $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_N]^\top \in \mathbb{R}^{N \times d}$ , which has been normalized to 0-mean and 1-standard-deviation along sample direction (i.e.,  $\frac{1}{N} \sum_{i=1}^N \mathbf{h}_i = \mathbf{0}$ ), its covariance matrix is  $\Sigma = \frac{1}{N} \mathbf{H}^\top \mathbf{H}$ . After eigendecomposition for  $\Sigma$ , we can obtain  $d$  unit orthogonal eigenvectors  $[\mathbf{q}_1, \dots, \mathbf{q}_d]$  associated to eigenvalues  $[\lambda_1, \dots, \lambda_d]$ , respectively. According to  $\frac{1}{N} \mathbf{H}^\top \mathbf{H} \mathbf{q}_k = \lambda_k \mathbf{q}_k$ , it can be known that

$$\frac{1}{N} \mathbf{q}_k^\top \mathbf{H}^\top \mathbf{H} \mathbf{q}_k = \lambda_k \mathbf{q}_k^\top \mathbf{q}_k = \lambda_k. \quad (18)$$

Taking a principal direction  $\mathbf{q}_k$  as explanation, the projection of a sample  $\mathbf{h}_i$  onto this direction is  $z_i = \mathbf{q}_k^\top \mathbf{h}_i$ , and the mean of all projections is

$$\bar{z} = \frac{1}{N} \sum_{i=1}^N z_i = \frac{1}{N} \sum_{i=1}^N \mathbf{q}_k^\top \mathbf{h}_i = 0. \quad (19)$$

Thus, along the principal direction  $\mathbf{q}_k$ , the variance is

$$\begin{aligned} & \frac{1}{N} \sum_{i=1}^N (z_i - \bar{z})^2 \\ &= \frac{1}{N} \sum_{i=1}^N \mathbf{q}_k^\top \mathbf{h}_i \mathbf{h}_i^\top \mathbf{q}_k \\ &= \frac{1}{N} \mathbf{q}_k^\top \left( \sum_{i=1}^N \mathbf{h}_i \mathbf{h}_i^\top \right) \mathbf{q}_k \\ &= \frac{1}{N} \mathbf{q}_k^\top \mathbf{H}^\top \mathbf{H} \mathbf{q}_k \\ &= \lambda_k. \end{aligned} \quad (20)$$

The above equation demonstrates that the variance of data  $\mathbf{H}$  along the direction  $\mathbf{q}_k$  is equal to  $\lambda_k$ . Thus, the proof is concluded.  $\square$

In classical machine learning or pattern recognition literature, similar descriptions akin to Property 2 can also be found. For the completeness of this paper, we present it here as well.

### 3.4 Proof of Proposition 5.1

For convenience, we restate Proposition 5.1 here.

**Proposition 5.1.** *When the representations scatter over the unit hypersphere  $\mathcal{S}^{d-1}$  uniformly (that is, they obey a complete uniform distribution), their entropy will reach the maximum value.*

PROOF. Assuming that the representations follow a distribution  $p(X)$  on the unit hypersphere  $\mathcal{S}^{d-1}$ , proving Proposition 5.1 is equivalent to demonstrate that when  $p(X)$  is a uniform distribution, the entropy of variable  $X$  is maximized. The corresponding mathematical expression can be stated as follows:

$$\begin{aligned} & \max \quad - \int_{\mathcal{S}^{d-1}} p(X) \ln p(X) dX \\ & \text{s.t.} \quad \int_{\mathcal{S}^{d-1}} p(X) dX = 1 \end{aligned} \quad (21)$$

To find the optimal form of  $p(X)$  subject to the constraint  $\int_{\mathcal{S}^{d-1}} p(X) dX = 1$ , we construct the following Lagrangian function:

$$L(p(X), \lambda) = - \int_{\mathcal{S}^{d-1}} p(X) \ln p(X) dX + \lambda \cdot \left( \int_{\mathcal{S}^{d-1}} p(X) dX - 1 \right), \quad (22)$$

where  $\lambda$  denotes Lagrange multiplier.

Taking the derivative of the Lagrangian function  $L(p(X), \lambda)$  with respect to  $p(X)$  and setting it equal to zero, we know that

$$\frac{\partial L(p(X), \lambda)}{\partial p(X)} = -\ln p(X) - 1 + \lambda = 0. \quad (23)$$

Hence, the optimal form of the probability density function is

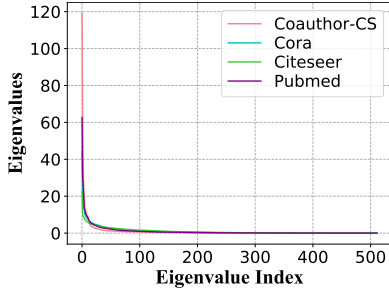
$$p(X) = e^{\lambda-1}. \quad (24)$$

To satisfy the constraint  $\int_{\mathcal{S}^{d-1}} p(X) dX = 1$ , we have

$$\int_{\mathcal{S}^{d-1}} p(X) dX = \int_{\mathcal{S}^{d-1}} e^{\lambda-1} dX = e^{\lambda-1} \int_{\mathcal{S}^{d-1}} dX = 1. \quad (25)$$

Letting  $S = \int_{\mathcal{S}^{d-1}} dX$  represent the surface area of the unit hypersphere  $\mathcal{S}^{d-1}$ , we can know that

$$\lambda = 1 - \ln S. \quad (26)$$



**Figure 1: Eigenvalues of covariance matrices of node representations from the randomly initial representation encoder.**

According to [19], we can obtain that  $S = \frac{2\pi^{d/2}}{\Gamma(d/2)}$ , where  $\Gamma(\cdot)$  denotes the gamma function. Taking Eq. (26) into Eq. (24), it can be known that

$$p(X) = \frac{1}{S} = \frac{\Gamma(d/2)}{2\pi^{d/2}}, \quad (27)$$

which is a uniform distribution on the unit hypersphere. Thus, it can be known that the entropy of representations on the unit hypersphere reach the maximum value when they obey a uniform distribution. We conclude the proof.  $\square$

## 4 VISUALIZATIONS OF INITIAL EIGENVALUES

In the main text, we have pointed out that it is inappropriate to directly construct an objective function (that is,  $\mathcal{L}_G = \ln \frac{\det(\Sigma_{A,B})}{\det(\Sigma_A) \cdot \det(\Sigma_B)}$ ) based on  $I_G(X; Y) = \frac{1}{2} \ln \frac{\det(\Sigma_X) \cdot \det(\Sigma_Y)}{\det(\Sigma_{X,Y})}$ , as it would result in numerical instability.

Figure 1 visualizes the eigenvalues of covariance matrices of node representations in the initial epoch of the pretraining phase. It can be observed that a significant portion of the eigenvalues are close to or equal to zero. This phenomenon leads to the determinant of the covariance matrix, which is numerically equivalent to the product of all eigenvalues, being zero. Therefore, including the determinant of the original covariance matrix in the objective function will potentially lead to computational instability. Our empirical experiments suggest that when representation dimension is greater than 64, the determinant of the original covariance matrix becomes zero for nearly all datasets.

## 5 RELATION BETWEEN OVERALL PERFORMANCE AND DEVIATION FROM NORMALITY

To further validate the performance of our method in non-Gaussian scenarios, we explicitly enforce representations to deviate from normality during training. To this end, we impose probability constraints to drive node representations towards other distributions, such as the Cauchy distribution. For the representation matrix  $\mathbf{H}_A$  from view  $A$ , its  $j$ -th channel can be regarded as  $N$  empirical points of a single-dimensional variable. We can calculate their mean  $\mu_A^j$  and standard deviation  $\sigma_A^j$  and further construct a Cauchy distribution  $p_{cau}(x|\mu_A^j, \sigma_A^j)$ . It is worth noting that the mean and standard

deviation correspond to the location and scale of the Cauchy distribution, respectively. The similar operations can be expanded to other channels and views. Therefore, we can constrain node representations to approach the Cauchy distribution by minimizing negative log-likelihood of all channels:

$$\begin{aligned} \mathcal{L}_{cau} = & \frac{1}{N \cdot d} \sum_{j=1}^d \sum_{i=1}^N -\log p_{cau}(H_A^{ij} | \mu_A^j, \sigma_A^j) \\ & + \frac{1}{N \cdot d} \sum_{j=1}^d \sum_{i=1}^N -\log p_{cau}(H_B^{ij} | \mu_B^j, \sigma_B^j). \end{aligned} \quad (28)$$

Taking GMIM as an example,  $\mathcal{L}_{cau}$  can be appended to  $\mathcal{L}_{GMIM-IC}$  to jointly supervise the training process. The ultimate objective function can be formulated as

$$\mathcal{L}_{GMIM-IC} + \alpha \cdot \mathcal{L}_{cau}, \quad (29)$$

where the weighted coefficient  $\alpha$  is utilized to regulate the degree of deviation from normality (that is, the proximity to the Cauchy distribution). Apart from the Cauchy distribution, we can also formulate other analogous objective functions to drive representations towards alternative distributions, including the Student's t-distribution and the Laplace distribution. With various probability constraints, Figure 2 shows classification performance under distinct degrees of deviation of node representations from the Gaussian distribution. Overall, as the degree of deviation from the normal distribution increases, our method can maintain its performance, which grants the method higher robustness and broader application scenarios. In summary, although our method is derived and proposed under the Gaussian assumption, it remains effective even when the representations deviate from the Gaussian distribution.

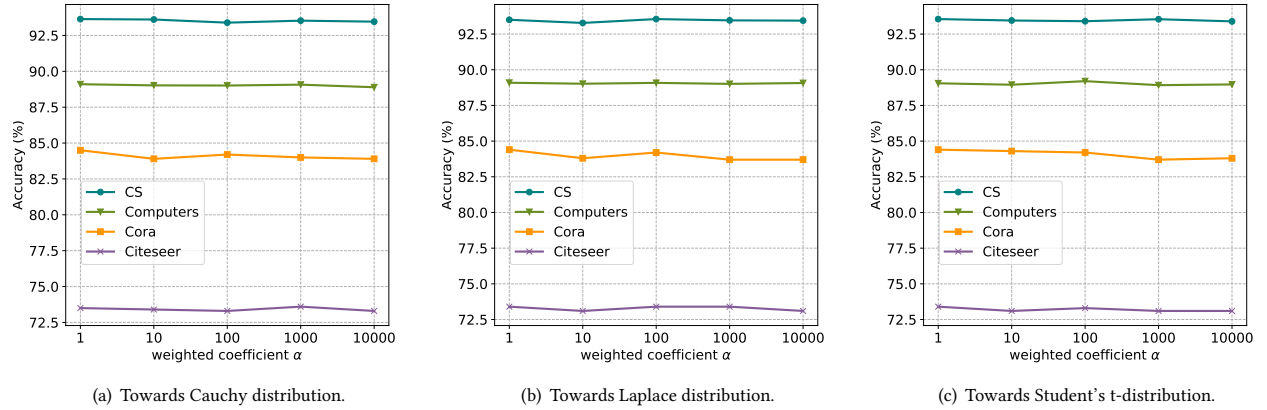
## 6 RELATION WITH DISENTANGLED REPRESENTATION LEARNING AND COMPARISON WITH PEER WORKS

### 6.1 Relation with Disentangled Representation Learning

Our method expects to enforce the representations to achieve an isotropic Gaussian distribution with the aim of decoupling various dimensions and learning diverse representations, which is closely linked to a branch of deep learning called Disentangled Representation Learning (DRL) [2, 18, 22]. The objective of disentangled representation learning is to achieve a clear separation of the distinct, independent, and informative generative factors inherent in the data [2]. DRL emphasizes the statistical independence among latent variables, which can be traced back to Independent Component Analysis (ICA) [23].

Independent Component Analysis, a computationally efficient Blind Source Separation (BSS) [16] technique, thinks that the observed mixed signals are obtained through a linear combination of source signals and aims to recover latent variables from observations. The traditional ICA assumes that the source signals follow non-Gaussian distributions and are statistically independent. The assumption of statistical independence among latent variables is, in fact, a disentangled or independence constraint. Non-linear





**Figure 2: Classification performance under different degrees of deviation of node representations from the Gaussian distribution. A larger value of  $\alpha$  indicates a greater degree of deviation from the Gaussian distribution.**

ICA [11] posits that the observed signals are obtained through a nonlinear transformation of the source signals.

The variational autoencoder (VAE) [13] is a modification of the autoencoder that incorporates the concept of variational inference, which can realize dimension-wise disentanglement. In VAEs, there is a term used to minimize the KL divergence between the variational posterior and the prior distribution. The chosen prior distribution is typically selected to satisfy certain independent properties, such as an isotropic Gaussian distribution. As a result, the KL divergence term potentially imposes a independent constraint on the latent variables. The  $\beta$ -VAE [8] multiplies the KL divergence term by a penalty factor  $\beta$  to enhance the disentangling effect on the latent variables. The KL divergence term shares a similar underlying principle with the entropy maximization term in our paper. [4] demonstrate that the penalty term in  $\beta$ -VAE tends to enhance the dimension-wise independence of the latent variable, but it also diminishes the capacity of the latent variables to preserve information from the input. Similar to decorrelation-based self-supervised learning methods such as CCA-SSG, DIP-VAE [14] directly regularizes the elements in the covariance matrix of the posterior distribution, making it approach the identity matrix. FactorVAE [12] introduces a term known as Total Correlation to quantify the level of dimension-wise independence.

## 6.2 Comparison with CorInfoMax

Towards the conclusion of our work, we observed that a peer study [17], called CorInfoMax, shares certain similarities with our method, especially in terms of the objective function. Despite the similarities, the two works still exhibit significant distinctions as follows:

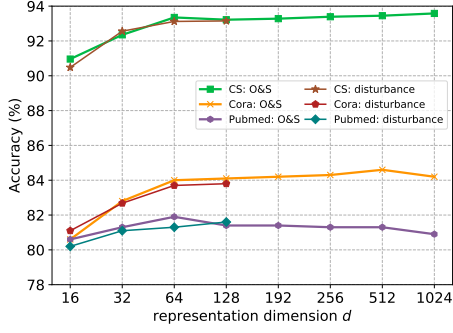
**Motivation.** As described in the main text, our research starts from the Gaussian assumption with the expectation of enabling direct computation of mutual information, thereby eliminating mutual information neural estimators and simplifying existing contrastive learning methods. The core motivation of CorInfoMax is

to leverage information entropy maximization to overcome the collapse issues and learn informative representations, as can be reflected in the title, abstract, and introduction of their paper.

**Network Architecture.** Our method does not utilize additional projection heads. Our loss function directly operates on the output representations of the encoder  $f_{\theta}(\cdot)$ , while CorInfoMax first employs a projector of 3-layer MLP to map the outputs of the encoder to a new embedding space and then calculate loss function in the new space. Our approach reduces model complexity and enhances efficiency by directly optimizing the output space of the encoder. This distinction is not only reflected in the variations in network architecture but, more importantly, it actually indicates the disparities in motivations and underlying concepts between the two works. The purpose of our research is to enable the direct calculation of mutual information under the Gaussian assumption without relying on estimators and extra designs. The projectors introduced in CorInfoMax completely deviate from our initial motivation.

**Numerical Stability During Training.** An important issue we addressed in our research is that directly incorporating the logarithm of the determinant of the covariance matrix in the objective function can lead to numerical instability, which is detailedly analyzed and discussed in Subsection 3.2 of the main text and Section 4 of this supplement. To cope with this issue, we adopt the strategy of *offsetting and scaling eigenvalues* to be around 1. CorInfoMax turns to *adding a disturbance* in their objective. The comparisons between the two strategies are placed in Figure 3. When dimensions are higher than 128, the training process under the strategy of adding disturbance is terminated due to numerical instability, because it cannot change the fact that many eigenvalues remain very close to 0.

**Explanations of Relationship Between Various SSL Methods.** We establish connections between contrastive and decorrelation-based methods, providing an explanation for decorrelation-based methods from the perspective of MI maximization. These contributions



**Figure 3: Performance of two strategies under various dimensions. O&S: offsetting and scaling eigenvalues in our work; disturbance: the strategy of adding a disturbance in CorInfoMax. For “disturbance”, when dimension is set to higher than 128, the training process is terminated due to numerical instability, and the results are not obtainable.**

are of significant importance in establishing a unified theoretical framework for self-supervised learning methods. The work [17] does not involve these aspects.

*Research on Dimensional Collapse.* In Section 5.1 of the main paper, we theoretically demonstrate from the perspective of eigen spectrum that the entropy maximization term can prevent dimensional collapse. In Section 5.2 of the main paper, we show that *InfoNCE* potentially maximizes information entropy. The work [17] does not involve these contents, which primarily provides empirical evidence for the effectiveness of their method in preventing dimension collapse.

Finally, we express our gratitude to the authors of CorInfoMax for their outstanding contributions to the self-supervised learning community.

### 6.3 Discussion and Comparison with M-ILBO

Here, we discuss a work called M-ILBO [15], which also performs graph self-supervised learning by enhancing consistency between representations from various views and maximizing information entropy.

**General framework and universal principles of multi-view self-supervised learning.** Multi-view self-supervised learning, including contrastive learning, can be divided into two basic modules. The first module is for extracting cross-view invariant information, which can be achieved by minimizing the Mean Squared Error (MSE) between representations of two views or maximizing their cosine similarity. The second module is for preventing model collapse, including dimension collapse and complete collapse. Existing strategies include using negative samples [25], decorrelation strategies [1] and asymmetric architectures [6]. Both M-ILBO and our paper can be encompassed within this framework, and they have implemented the two basic modules using different strategies.

**Discussion about our identity constraint and the cross-view consistency in M-ILBO.** Multi-view self-supervised learning aims to extract invariant information across different augmented views. Both the identity constraint in our paper and the

cross-view consistency in M-ILBO align representations from two views using Mean Squared Error (MSE), thereby extracting common information across views. Due to its effectiveness in aligning views, Mean Squared Error is commonly used in self-supervised learning [1, 25, 27].

**Discussion about  $\mathcal{L}_{cl}$  in M-ILBO and our entropy maximization term.** Generally speaking, both  $\mathcal{L}_{cl}$  in M-ILBO and entropy maximization term of our paper achieve the same purpose through various strategies, namely, avoiding model collapse and learning diverse representations by maximizing the entropy of representations. M-ILBO provides a strategy of achieving an estimation of entropy through neural estimation of mutual information based on Jensen-Shannon estimator. Our approach takes a different approach, which assumes a Gaussian distribution for node representations and directly obtains the exact value of entropy without relying on neural estimators.

## 7 MORE EXPERIMENTS AND STATISTICS OF DATASETS

### 7.1 Visualization of Correlation Matrix

Figure 4 provides visualizations of correlation matrices of node representations under various settings on Cora and Pubmed. Specifically, for a representation matrix  $H \in \mathbb{R}^{N \times d}$  which has been normalized to 0-mean and 1-standard deviation, the correlation matrix is  $\frac{1}{N} H^T H$ . In other words, each element of correlation matrix denotes the Pearson correlation coefficient of two variables (*i.e.*, two channels). As shown in Figure 4(a,d), the off-diagonal elements of correlation matrices are large without considering entropy maximization term in GMIM-IC, indicating that various channels of representation matrix coupled together. That is to say, the issue of dimensional collapse has occurred. Moreover, the two proposed variants, GMIM and GMIM-IC, can effectively decorrelate various representation channels and mitigate dimensional collapse issue.

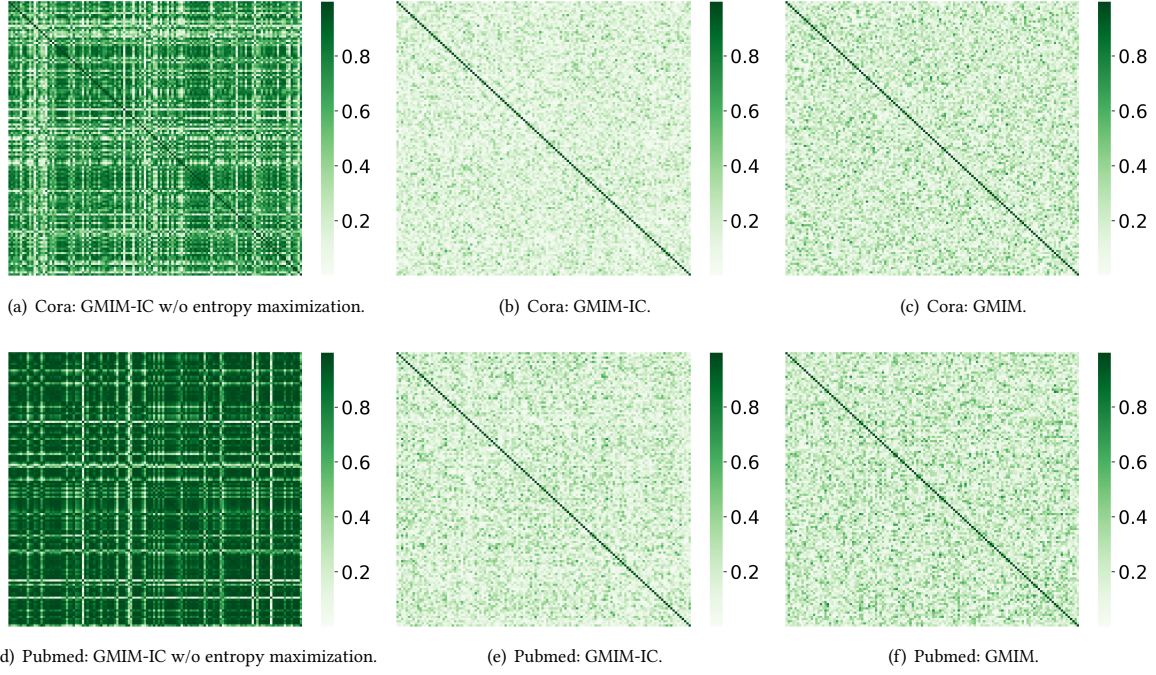
### 7.2 Synergistic Evolution Between GMI and that with Shifted and Scaled Eigenvalues

In the main context, considering that directly designing the objective function based on Gaussian mutual information  $I_G(X; Y) = \frac{1}{2} \ln \frac{\det(\Sigma_X) \cdot \det(\Sigma_Y)}{\det(\Sigma_{X,Y})}$  will lead numerical instability, we proposed a feasible alternative by shifting and scaling the eigenvalues of the covariance matrix, denoted as  $I'_G(X; Y) = \ln \frac{\det(I + \eta \cdot \Sigma_X) \cdot \det(I + \eta \cdot \Sigma_Y)}{\det(I + \eta \cdot \Sigma_{X,Y})}$ .

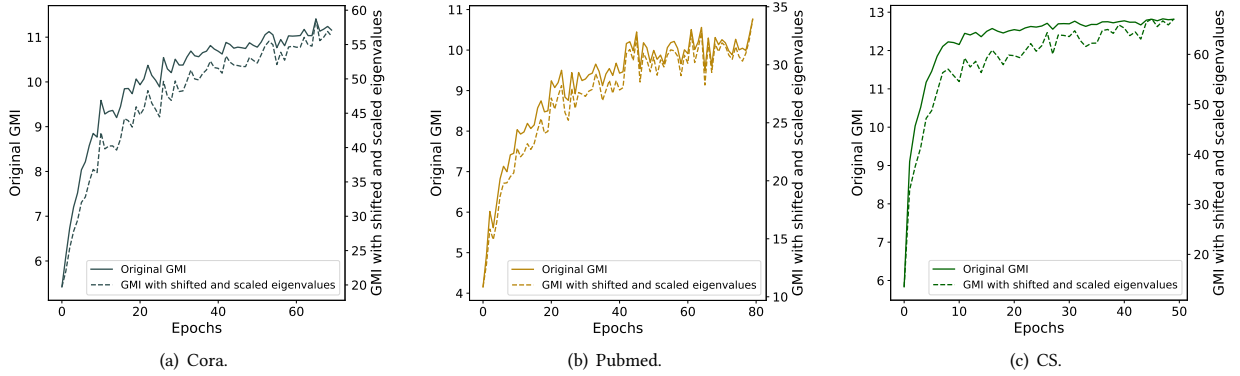
We visualize the synergistic evolution between  $I_G$  and  $I'_G$  during the training phase in Figure 5. It is worth mentioning that all experiments are conducted under low-dimensional settings, where the original GMI  $I_G$  remains normal and valid values. It can be observed that the two lines exhibit consistent patterns of variation in each subfigure. This phenomenon provides evidence supporting the rationality of obtaining a viable objective function through shifting and scaling the eigenvalues.

### 7.3 Effect of Augmentation Intensity

We conduct a sensitivity analysis on the augmentation intensity by examining the effects of various combinations of the edge removal ratio  $p_e$  and the feature masking ratio  $p_f$ . The results, presented



**Figure 4: Visualizations of the correlation matrices (absolute value) of representations under various settings on Cora and Pubmed.**

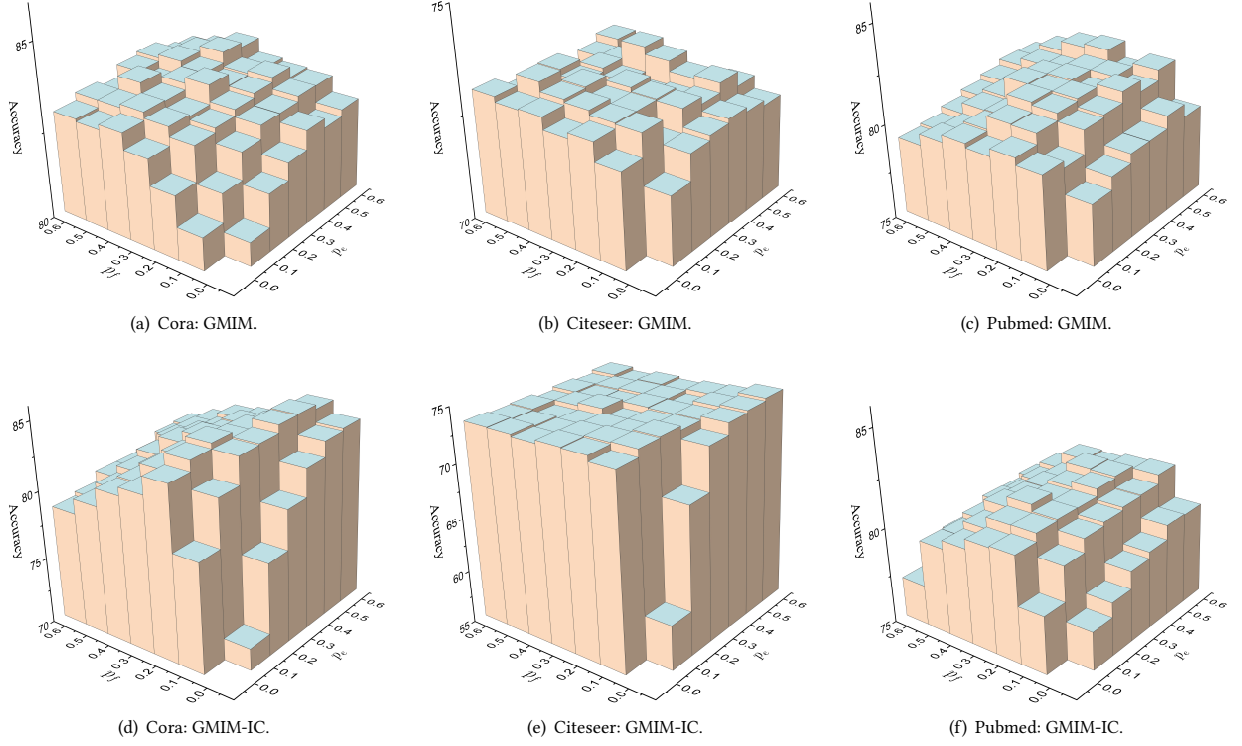


**Figure 5: Synergistic evolution between GMI and that with shifted and scaled eigenvalues. Specifically, the original GMI (solid line) is adopted as the optimization objective while the corresponding values (dashed line) of GMI with shifted and scaled eigenvalues are recorded.**

in Figure 6, indicate that our method is more sensitive to augmentation in features ( $p_f$ ) compared to that in graph structure ( $p_e$ ). Overall, within an appropriate range of  $p_e$  and  $p_f$ , our approach consistently achieves competitive results. Even when subjected to strong augmentation (e.g.,  $p_e = 0.6$  and  $p_f = 0.6$ ), our method still maintains a satisfactory performance level.

#### 7.4 t-SNE Visualizations.

To achieve a more profound understanding of our approach and conduct a comprehensive comparison with different methods, a series of t-SNE plots [24] is employed to visualize the raw features and learned representations under various methods and distinct configurations in Figure 7. In Figure 7(a), the 2-dimensional t-SNE embeddings of the raw features present a chaotic distribution and can not show discriminative clusters. The visualization in Figure 7(b), characterized by a complex elliptical shape, highlights that the



**Figure 6: The classification accuracy under various combinations of feature masking ratio  $p_f$  and edge removal ratio  $p_e$ .**

method lacking the identity constraint term can not capture semantically meaningful information. In Figure 7(c), the two dimensions of the t-SNE embeddings show a certain correlation, potentially indicating dimensional collapse issue in high-dimensional representation space. This phenomenon illustrates the effect of the entropy maximization term in learning diverse representations and avoiding dimensional collapse. In Figure 7(d), the t-SNE results form discernible and interpretable clusters based on their true categories, indicating that our method can learn meaningful and diverse representations. The second row in Figure 7 shows t-SNE embeddings of the other four methods, and the visual results of various methods do not exhibit significantly distinct appearance. However, upon closer inspection, our method demonstrates better inter-class discriminability, especially concerning the clusters in purple, green, and blue.

## 7.5 Experiments on Ogbn-Arxiv

To further evaluate the effectiveness and efficiency of our method, we conduct experiments on a large-scale graph Ogbn-Arxiv [10]. Table 1 reports the validation and test accuracy of various graph self-supervised methods, where our method obtains good performance. It is worth mentioning that GRACE and GCA do not operate on a full graph manner but a subset of nodes are sampled as negative samples to avoid memory issues. Moreover, Figure 8 simultaneously presents the test accuracy and training time, indicating that our method can effectively balance performance and efficiency.

**Table 1: Validation and test accuracy on Ogbn-Arxiv. “OOM” indicates out-of-memory on a GPU with 24GB memory.**

|                | Validation       | Test             |
|----------------|------------------|------------------|
| DGI            | $71.19 \pm 0.24$ | $70.28 \pm 0.23$ |
| MVGRL          | OOM              | OOM              |
| GMI            | OOM              | OOM              |
| CCA-SSG        | $72.35 \pm 0.17$ | $71.33 \pm 0.21$ |
| BGRL           | $72.58 \pm 0.14$ | $71.52 \pm 0.14$ |
| GRACE          | $71.82 \pm 0.18$ | $70.91 \pm 0.21$ |
| GCA            | $71.63 \pm 0.20$ | $70.77 \pm 0.22$ |
| GMIM (ours)    | $72.26 \pm 0.16$ | $71.27 \pm 0.21$ |
| GMIM-IC (ours) | $72.48 \pm 0.18$ | $71.42 \pm 0.19$ |

## 7.6 Effect of Backbones

We employ various graph neural networks including GCN, GAT, SGC [26], and GraphSAGE [7] as representation learners and investigate the performance of our method under their guidance. The experimental results are illustrated in Table 2, where GMIM-IC serves as the objective function. The expressive capacities of different networks vary, resulting in distinct performance across various datasets. Overall, our method achieves highly competitive results across different backbone networks, demonstrating the effectiveness, generalization, and flexibility of our approach.



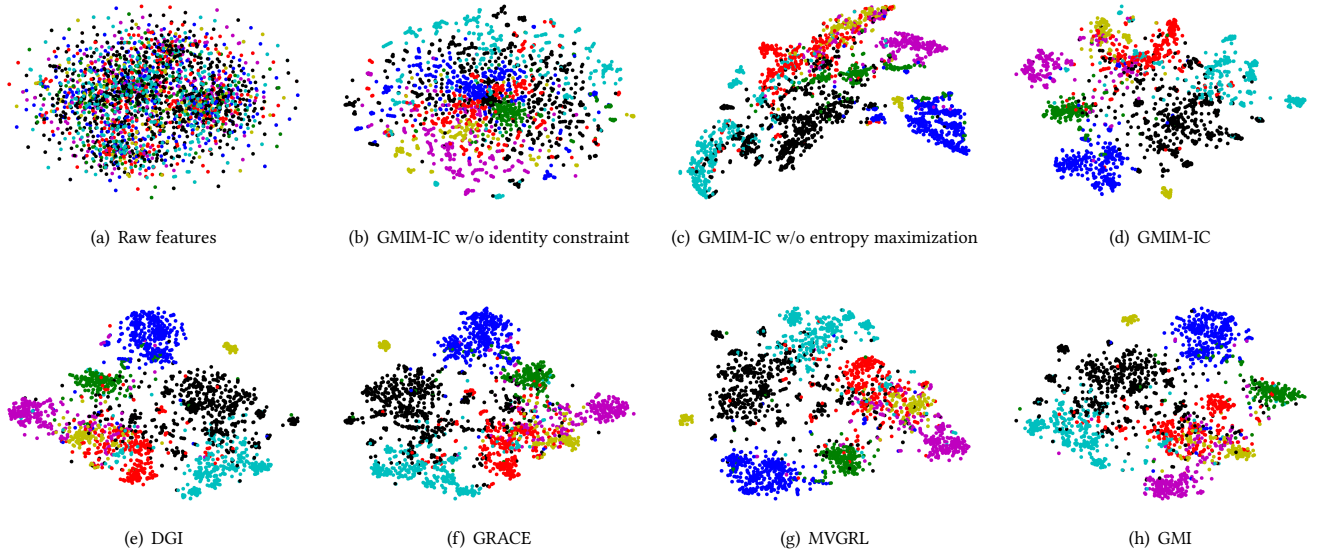


Figure 7: t-SNE visualizations of the raw features and learned representations of various methods on Cora. "w/o" stands for "without". Best viewed in colors.

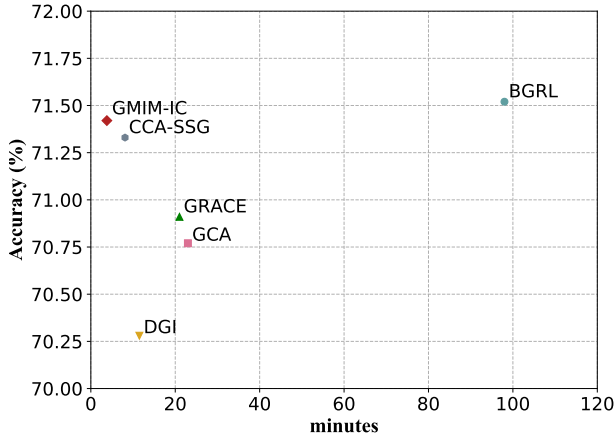


Figure 8: Classification accuracy in test set and training time on Ogbn-Arxiv.

Table 2: Classification accuracy under various backbones.

| Backbone  | Cora | Citeseer | Computers | Photo |
|-----------|------|----------|-----------|-------|
| GCN       | 84.5 | 73.6     | 89.04     | 93.17 |
| GAT       | 83.9 | 73.7     | 89.05     | 92.96 |
| SGC       | 83.5 | 73.6     | 89.02     | 93.08 |
| GraphSAGE | 83.5 | 72.4     | 88.91     | 92.69 |

## 7.7 Details of Hyperparameter Configuration

The details of hyperparameter configuration for GMIM and GMIM-IC are placed in Table 3 and 4, respectively.

Table 3: Hyperparameter configuration of the experiments for GMIM. "lr" indicates learning rate while "wd" denotes weight decay.

| Dataset     | GMIM   |          |      |    |       |       |
|-------------|--------|----------|------|----|-------|-------|
|             | Layers | Rep. dim | lr   | wd | $p_f$ | $p_e$ |
| Cora        | 2      | 512      | 1e-3 | 0  | 0.4   | 0.5   |
| Citeseer    | 1      | 512      | 1e-3 | 0  | 0.4   | 0.5   |
| Pubmed      | 2      | 128      | 1e-3 | 0  | 0.3   | 0.5   |
| Computers   | 2      | 512      | 1e-3 | 0  | 0.1   | 0.3   |
| Photo       | 2      | 512      | 1e-3 | 0  | 0.2   | 0.3   |
| Coauthor-CS | 2      | 512      | 1e-3 | 0  | 0.2   | 1.0   |

Table 4: Hyperparameter configuration of the experiments for GMIM-IC. "lr" indicates learning rate while "wd" denotes weight decay.

| Dataset     | GMIM-IC |          |         |      |    |       |       |
|-------------|---------|----------|---------|------|----|-------|-------|
|             | Layers  | Rep. dim | $\beta$ | lr   | wd | $p_f$ | $p_e$ |
| Cora        | 2       | 512      | 1.0     | 1e-3 | 0  | 0.1   | 0.5   |
| Citeseer    | 1       | 512      | 0.5     | 1e-3 | 0  | 0.0   | 0.6   |
| Pubmed      | 2       | 64       | 3.0     | 1e-3 | 0  | 0.3   | 0.5   |
| Computers   | 2       | 1024     | 4.0     | 1e-3 | 0  | 0.1   | 0.3   |
| Photo       | 2       | 2048     | 5.0     | 1e-3 | 0  | 0.2   | 0.3   |
| Coauthor-CS | 2       | 1024     | 1.0     | 1e-3 | 0  | 0.2   | 1.0   |

## 7.8 Details of the Experimental Datasets

The statistics of the experimental datasets are summarized in Table 5. The details of the datasets are as follows:



**Table 5: Statistics of the experimental datasets.**

| Dataset          | Nodes   | Edges     | Features | Classes |
|------------------|---------|-----------|----------|---------|
| Cora             | 2,708   | 5,429     | 1,433    | 7       |
| Citeseer         | 3,327   | 4,732     | 3,703    | 6       |
| Pubmed           | 19,717  | 44,338    | 500      | 3       |
| Amazon-Computers | 13,752  | 245,861   | 767      | 10      |
| Amazon-Photo     | 7,650   | 119,081   | 745      | 8       |
| Coauthor-CS      | 18,333  | 81,894    | 6,805    | 15      |
| Ogbn-Arxiv       | 169,343 | 2,332,386 | 128      | 40      |

- **Cora**, **Citeseer**, and **Pubmed** [20] are citation networks where nodes represent documents and edges denote citation relationships. Each document is assigned a class label that indicates its subject category, and it is characterized by a bag-of-words feature vector.
- **Amazon-Computers** and **Amazon-Photo** [21] are two graphs derived from the Amazon dataset, capturing co-purchase relationships. The nodes in these graphs represent products, and an edge exists between two nodes if they are frequently purchased together. Each node is associated with a sparse bag-of-words feature vector based on product reviews. The category of each node is indicated by its label.
- **Coauthor-CS** [21] is an academic network in the field of computer science, where nodes represent authors and edges indicate co-authorship relationships. Two authors are connected by an edge if they have collaborated on a research paper.
- **Ogbn-Arxiv** [10] is a directed citation network among some computer science arXiv papers. Each node on the graph corresponds to an arXiv paper, while directed edges indicate the citing relationships between papers. Each paper is associated with a 128-dimensional feature vector.

## 8 ALGORITHM

The overall algorithm flows for GMIM and GMIM-IC in the form of PyTorch-style pseudocode are placed in Algorithm 1 and 2, respectively.

## REFERENCES

- [1] Adrien Bardes, Jean Ponce, and Yann LeCun. 2022. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. In *International Conference on Learning Representations*.
- [2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* (2013), 1798–1828.
- [3] Nizar Bouhlel and Ali Dziri. 2019. Kullback–Leibler Divergence Between Multivariate Generalized Gaussian Distributions. *IEEE Signal Processing Letters* 26, 7 (2019), 1021–1025.
- [4] Ricky T. Q. Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. 2018. Isolating Sources of Disentanglement in Variational Autoencoders. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc.
- [5] Thomas M Cover. 1999. *Elements of information theory*. John Wiley & Sons.
- [6] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. 2020. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 21271–21284.
- [7] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. Curran Associates Inc., 1025–1035.
- [8] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations*.
- [9] Michael D Hirschhorn. 2007. The am-gm inequality. *Mathematical Intelligencer* (2007).
- [10] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open Graph Benchmark: Datasets for Machine Learning on Graphs. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 22118–22133.
- [11] Juha Karhunen. 2001. Nonlinear independent component analysis. *ICA: Principles and Practice* (2001), 113–134.
- [12] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by Factorising. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 2649–2658.
- [13] Diederik P Kingma and Max Welling. 2022. Auto-Encoding Variational Bayes. arXiv:1312.6114 [stat.ML]
- [14] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. 2018. Variational Inference of Disentangled Latent Concepts from Unlabeled Observations. In *International Conference on Learning Representations*.
- [15] Yixuan Ma, Xiaolin Zhang, Peng Zhang, and Kun Zhan. 2023. Entropy Neural Estimation for Graph Contrastive Learning. In *Proceedings of the 31st ACM International Conference on Multimedia (MM '23)*. Association for Computing Machinery, 435–443.
- [16] Ganesh R Naik, Wenwu Wang, et al. 2014. Blind source separation. *Berlin: Springer* 10 (2014), 978–3.
- [17] Serdar Ozsoy, Shadi Hamdan, Sercan Arik, Deniz Yuret, and Alper Erdogan. 2022. Self-Supervised Learning with an Information Maximization Criterion. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 35240–35253.
- [18] Abbavaram Gowtham Reddy, Benin Godfrey L, and Vineeth N Balasubramanian. 2022. On Causally Disentangled Representations. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 7 (2022), 8089–8097.
- [19] Jason DM Rennie. 2005. Volume of the n-sphere. (2005).
- [20] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.
- [21] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Pitfalls of Graph Neural Network Evaluation. arXiv:1811.05868 [cs.LG]
- [22] Xinwei Shen, Furui Liu, Hanze Dong, Qing Lian, Zhitang Chen, and Tong Zhang. 2022. Weakly Supervised Disentangled Generative Causal Representation Learning. *J. Mach. Learn. Res.* 23 (2022).
- [23] James V Stone. 2004. Independent component analysis: a tutorial introduction. (2004).
- [24] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605.
- [25] Tongzhou Wang and Phillip Isola. 2020. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research)*. PMLR, 9929–9939.
- [26] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning*, Vol. 97. PMLR, 6861–6871.
- [27] Hengrui Zhang, Qitian Wu, Junchi Yan, David Wipf, and Philip S Yu. 2021. From Canonical Correlation Analysis to Self-supervised Graph Neural Networks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 76–89.
- [28] Yufeng Zhang, Wanwei Liu, Zhenbang Chen, Ji Wang, and Kenli Li. 2023. On the Properties of Kullback-Leibler Divergence Between Multivariate Gaussian Distributions. arXiv:2102.05485 [cs.IT]

**Algorithm 1** PyTorch-style Code for GMIM.

---

```

1161 # f: shared neural encoder
1162 # d: representation dimension
1163 # adj: original graph topology
1164 # feat: original node features
1165 # eta: scaling factor
1166 # epochs: total training epochs
1167 # N: number of nodes

1168 Id, I2d = torch.eye(d), torch.eye(2 * d)
1169 for _ in range(epochs):
1170     # generate two randomly augmented views of original graph
1171     adj_A, feat_A = augment(adj, feat)
1172     adj_B, feat_B = augment(adj, feat)

1173     # get output representations of encoder
1174     H_tilde_A = f(adj_A, feat_A)
1175     H_tilde_B = f(adj_B, feat_B)

1176     # normalize representations along sample direction
1177     H_A = (H_tilde_A - H_tilde_A.mean(0)) / H_tilde_A.std(0)
1178     H_B = (H_tilde_B - H_tilde_B.mean(0)) / H_tilde_B.std(0)

1179     # compute covariance matrix
1180     Cov_A = torch.mm(H_A.T, H_A) / N
1181     Cov_B = torch.mm(H_B.T, H_B) / N
1182     CrossCov = torch.mm(H_A.T, H_B) / N
1183     JointCov = torch.cat([torch.cat([Cov_A, CrossCov], dim=1),
1184                           torch.cat([CrossCov.T, Cov_B], dim=1)], dim=0)

1185     # calculate loss function
1186     loss = torch.log(torch.det(I2d + eta * JointCov) / torch.det(Id + eta * Cov_A) / torch.det(Id + eta * Cov_B))

1187     # update parameters
1188     loss.backward()
1189     optimizer.step()

```

---

**Algorithm 2** PyTorch-style Code for GMIM-IC.

---

```

1190 # f: shared neural encoder
1191 # d: representation dimension
1192 # adj: original graph topology
1193 # feat: original node features
1194 # eta: scaling factor
1195 # beta: balancing factor
1196 # epochs: total training epochs
1197 # N: number of nodes

1198 Id = torch.eye(d)
1199 for _ in range(epochs):
1200     # generate two randomly augmented views of original graph
1201     adj_A, feat_A = augment(adj, feat)
1202     adj_B, feat_B = augment(adj, feat)

1203     # get output representations of encoder
1204     H_tilde_A = f(adj_A, feat_A)
1205     H_tilde_B = f(adj_B, feat_B)

1206     # normalize representations along sample direction
1207     H_A = (H_tilde_A - H_tilde_A.mean(0)) / H_tilde_A.std(0)
1208     H_B = (H_tilde_B - H_tilde_B.mean(0)) / H_tilde_B.std(0)

1209     # compute covariance matrix
1210     Cov_A = torch.mm(H_A.T, H_A) / N
1211     Cov_B = torch.mm(H_B.T, H_B) / N

1212     # calculate loss function
1213     loss_ic = (H_A - H_B).pow(2).sum() / N
1214     loss_em = - torch.log(torch.det(Id + eta * Cov_A) * torch.det(Id + eta * Cov_B))
1215     loss = loss_ic + beta * loss_em

1216     # update parameters
1217     loss.backward()
1218     optimizer.step()

```

---