A Neural Collapse Perspective on Feature Evolution in Graph Neural Networks

Vignesh Kothapalli[1,3], Tom Tirer[2], Joan Bruna[1]

1. New York University
2. Bar-Ilan University
3. LinkedIn Engineering

NeurIPS 2023

Outline

Supervised training of DNNs for classification tasks can be formulated as an Empirical Risk Minimization (ERM) problem:

$$\widehat{\mathbf{R}}(\Theta) = \min_{\Theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\psi_{\Theta}(\mathbf{X}_i), \mathbf{Y}_i). \quad (1)$$

Here:

- $\mathbf{X}_i \in \mathbb{R}^{d_0 \times N}, \mathbf{Y}_i \in \mathbb{R}^{C \times N}$ represent the input and label matrices.
- $\psi_{\Theta} : \mathbb{R}^{d_0} \to \mathbb{R}^{C}$ is an overparameterized feed-forward DNN.
- $\mathcal{L} : \mathbb{R}^{C} \times \mathbb{R}^{C} \to \mathbb{R}$ is the loss function (cross-entropy, MSE)

Training beyond zero-classification error, towards zero $\widehat{\mathbf{R}}(\Theta)$ (a.k.a Terminal Phase of Training (TPT)) leads to the "Neural Collapse" phenomenon!

Supervised training of DNNs for classification tasks can be formulated as an Empirical Risk Minimization (ERM) problem:

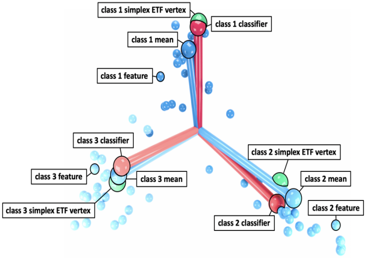
$$\widehat{\mathbf{R}}(\Theta) = \min_{\Theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\psi_{\Theta}(\mathbf{X}_i), \mathbf{Y}_i). \tag{1}$$

Here:

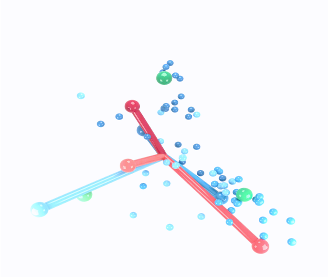
- $\mathbf{X}_i \in \mathbb{R}^{d_0 \times N}, \mathbf{Y}_i \in \mathbb{R}^{C \times N}$ represent the input and label matrices.
- $\psi_{\Theta} : \mathbb{R}^{d_0} \rightarrow \mathbb{R}^{C}$ is an overparameterized feed-forward DNN.
- $\mathcal{L} : \mathbb{R}^{C} \times \mathbb{R}^{C} \rightarrow \mathbb{R}$ is the loss function (cross-entropy, MSE)

Training beyond zero-classification error, towards zero $\widehat{\mathbf{R}}(\Theta)$ (a.k.a Terminal Phase of Training (TPT)) leads to the "Neural Collapse" phenomenon!

NC is characterized by four properties (NC1-4) pertaining to the penultimate layer features and the final layer classifier.



(a)



(b)

Figure 1: Penultimate layer features and final layer classifier: VGG13 + 3 classes from CIFAR10 [Papyan et.al 2020]

Feature means and covariances

For all "balanced" classes $c \in [C]$ and data points $i \in [n]$ within a class, the penultimate layer features are denoted as $\mathbf{h}_{c,i} \in \mathbb{R}^{d_{L-1}}$.

$$\text{class means: } \boldsymbol{\mu}_c = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_{c,i}$$

$$\text{global mean: } \boldsymbol{\mu}_G = \frac{1}{C}\sum_{c=1}^{C} \boldsymbol{\mu}_c$$

$$\text{within class covariance: } \Sigma_W = \frac{1}{Cn}\sum_{c=1}^{C}\sum_{i=1}^{n} \left((\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)(\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)^\top\right)$$

$$\text{between class covariance :} \Sigma_B = \frac{1}{C}\sum_{c=1}^{C} \left((\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top\right)$$

For all "balanced" classes $c \in [C]$ and data points $i \in [n]$ within a class, the penultimate layer features are denoted as $\mathbf{h}_{c,i} \in \mathbb{R}^{d_{L-1}}$.

$$\text{class means: } \boldsymbol{\mu}_c = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_{c,i}$$

$$\text{global mean: } \boldsymbol{\mu}_G = \frac{1}{C}\sum_{c=1}^{C} \boldsymbol{\mu}_c$$

$$\text{within class covariance: } \Sigma_W = \frac{1}{Cn}\sum_{c=1}^{C}\sum_{i=1}^{n} \left((\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)(\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)^\top\right)$$

$$\text{between class covariance :} \Sigma_B = \frac{1}{C}\sum_{c=1}^{C} \left((\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top\right)$$

Feature means and covariances

For all "balanced" classes $c \in [C]$ and data points $i \in [n]$ within a class, the penultimate layer features are denoted as $\mathbf{h}_{c,i} \in \mathbb{R}^{d_{L-1}}$.

$$\text{class means: } \boldsymbol{\mu}_c = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_{c,i}$$

$$\text{global mean: } \boldsymbol{\mu}_G = \frac{1}{C}\sum_{c=1}^{C} \boldsymbol{\mu}_c$$

$$\text{within class covariance: } \Sigma_W = \frac{1}{Cn}\sum_{c=1}^{C}\sum_{i=1}^{n}\left((\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)(\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)^\top\right)$$

$$\text{between class covariance :} \Sigma_B = \frac{1}{C}\sum_{c=1}^{C}\left((\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top\right)$$

For all "balanced" classes $c \in [C]$ and data points $i \in [n]$ within a class, the penultimate layer features are denoted as $\mathbf{h}_{c,i} \in \mathbb{R}^{d_{L-1}}$.

$$\text{class means: } \boldsymbol{\mu}_c = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_{c,i}$$

$$\text{global mean: } \boldsymbol{\mu}_G = \frac{1}{C}\sum_{c=1}^{C} \boldsymbol{\mu}_c$$

$$\text{within class covariance: } \Sigma_W = \frac{1}{Cn}\sum_{c=1}^{C}\sum_{i=1}^{n} \left((\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)(\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)^\top\right)$$

$$\text{between class covariance :} \Sigma_B = \frac{1}{C}\sum_{c=1}^{C} \left((\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top\right)$$

Feature means and covariances

For all "balanced" classes $c \in [C]$ and data points $i \in [n]$ within a class, the penultimate layer features are denoted as $\mathbf{h}_{c,i} \in \mathbb{R}^{d_{L-1}}$.

$$\text{class means: } \boldsymbol{\mu}_c = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_{c,i}$$

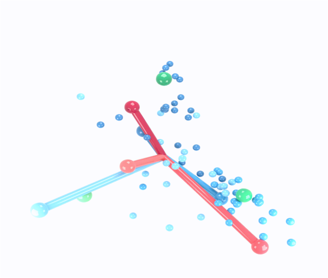
$$\text{global mean: } \boldsymbol{\mu}_G = \frac{1}{C}\sum_{c=1}^{C} \boldsymbol{\mu}_c$$

$$\text{within class covariance: } \Sigma_W = \frac{1}{Cn}\sum_{c=1}^{C}\sum_{i=1}^{n} \left((\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)(\mathbf{h}_{c,i} - \boldsymbol{\mu}_c)^\top\right)$$

$$\text{between class covariance :} \Sigma_B = \frac{1}{C}\sum_{c=1}^{C} \left((\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top\right)$$

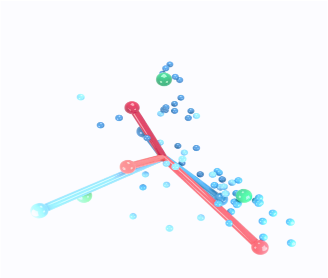
NC1: *Collapse of Variability:* For all classes $c \in [C]$ and data points $i \in [n]$ within a class, the penultimate layer features $\mathbf{h}_{c,i} \in \mathbb{R}^{d_{L-1}}$ collapse to their class means $\boldsymbol{\mu}_c = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_{c,i}$.

$$\mathcal{NC}1 := \frac{1}{C}\mathrm{tr}\{\Sigma_W \Sigma_B^{\dagger}\} \to 0 \tag{2}$$



NC2: *Preference towards a simplex ETF:* The re-centered class means $\boldsymbol{\mu}_c - \boldsymbol{\mu}_G, \forall c \in [C]$ are equidistant and equiangular from each other. Formally, matrix $\mathbf{M} \in \mathbb{R}^{C \times d_{L-1}}$ with columns $\frac{\boldsymbol{\mu}_c - \boldsymbol{\mu}_G}{\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2} \in \mathbb{R}^{d_{L-1}}, \forall c \in [C]$ represents a simplex ETF.

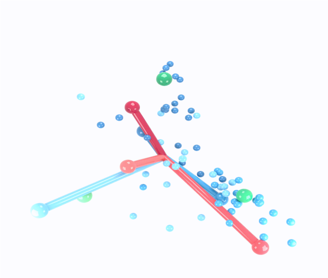
$$\mathcal{NC}2 := \left\| \frac{\mathbf{M}\mathbf{M}^\top}{\|\mathbf{M}\mathbf{M}^\top\|_F} - \frac{1}{\sqrt{C-1}} \left(\mathbf{I}_C - \frac{1}{C} \mathbf{1}_C \mathbf{1}_C^\top \right) \right\|_F \to 0 \qquad (3)$$



NC3: *Self-dual alignment:* The last-layer classifier $\mathbf{W} \in \mathbb{R}^{C \times d_{L-1}}$ is in alignment with the simplex ETF of $\mathbf{M}$ (up to rescaling) as:

$$\frac{\mathbf{W}}{\|\mathbf{W}\|_F} = \frac{\mathbf{M}}{\|\mathbf{M}\|_F}$$

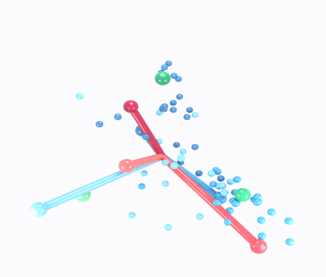
$$\mathcal{NC}3 := \left\| \frac{\mathbf{W}\mathbf{M}^\top}{\|\mathbf{W}\mathbf{M}^\top\|_F} - \frac{1}{\sqrt{C-1}} \left(\mathbf{I}_C - \frac{1}{C} \mathbf{1}_C \mathbf{1}_C^\top \right) \right\|_F \to 0 \qquad (4)$$



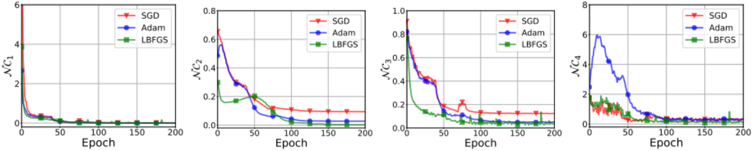
NC4: *Choose the nearest class mean:* for any new test point $\mathbf{x}_{test}$, the classification result is determined by: $argmin_{c \in [C]} \|\mathbf{h}_{test} - \boldsymbol{\mu}_c\|_2$. During training, one can track this property on $\mathbf{X}$ as a sanity check.

$$\mathcal{NC}4 := \frac{1}{Cn} \sum_{c=1}^{C} \sum_{i=1}^{n} \mathbb{I}(argmax_{c' \in [C]}(\langle \mathbf{w}_{c'}, \mathbf{h}_{c,i} \rangle + \mathbf{b}_{c'}) \neq argmin_{c' \in [C]} \|\mathbf{h}_{c,i} - \boldsymbol{\mu}_{c'}\|_2) \rightarrow 0. \tag{5}$$

Here $\mathbb{I}(.)$ is the indicator function and $\mathbf{b}_c \in \mathbb{R}$ is the c^{th} element of bias vector.



Experimental results



(e) $\mathcal{NC}_1$ (CIFAR10) (f) $\mathcal{NC}_2$ (CIFAR10) (g) $\mathcal{NC}_3$ (CIFAR10) (h) $\mathcal{NC}_4$ (CIFAR10)

Figure 2: NC1-4: ResNet18 + CIFAR10 [Zhu et.al 2021]

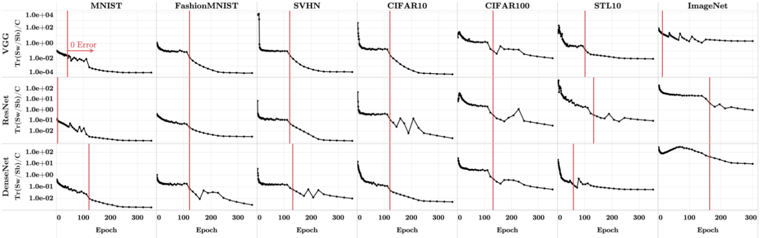


Figure 3: NC1 for VGG, ResNet, DenseNet on various datasets [Papyan et.al 2020]

Benefits!

- Better in-distribution generalization!
- Improved robustness to adversarial examples!
- Reduction in training time by fixing the last layer linear classifier as simplex ETF!
- Improved performance on imbalanced datasets by fixing the last layer linear classifier as simplex ETF!
- ... so on

How can we form a theoretical understanding of this phenomenon?

Benefits!

- Better in-distribution generalization!
- Improved robustness to adversarial examples!
- Reduction in training time by fixing the last layer linear classifier as simplex ETF!
- Improved performance on imbalanced datasets by fixing the last layer linear classifier as simplex ETF!
- ... so on

How can we form a theoretical understanding of this phenomenon?

- Better in-distribution generalization!
- Improved robustness to adversarial examples!
- Reduction in training time by fixing the last layer linear classifier as simplex ETF!
- Improved performance on imbalanced datasets by fixing the last layer linear classifier as simplex ETF!
- ... so on

How can we form a theoretical understanding of this phenomenon?

Benefits!

- Better in-distribution generalization!
- Improved robustness to adversarial examples!
- Reduction in training time by fixing the last layer linear classifier as simplex ETF!
- Improved performance on imbalanced datasets by fixing the last layer linear classifier as simplex ETF!
- ... so on

How can we form a theoretical understanding of this phenomenon?

Benefits!

- Better in-distribution generalization!
- Improved robustness to adversarial examples!
- Reduction in training time by fixing the last layer linear classifier as simplex ETF!
- Improved performance on imbalanced datasets by fixing the last layer linear classifier as simplex ETF!
- ... so on

How can we form a theoretical understanding of this phenomenon?

Benefits!

- Better in-distribution generalization!
- Improved robustness to adversarial examples!
- Reduction in training time by fixing the last layer linear classifier as simplex ETF!
- Improved performance on imbalanced datasets by fixing the last layer linear classifier as simplex ETF!
- ... so on

How can we form a theoretical understanding of this phenomenon?

- Under the assumption that the DNN is expressive enough to reach TPT, the "Unconstrained Features Model (UFM)" peels away the first 'L-1' hidden layers.
- The penultimate layer features are treated as freely optimizable!
- An idealistic model to explain neural collapse.

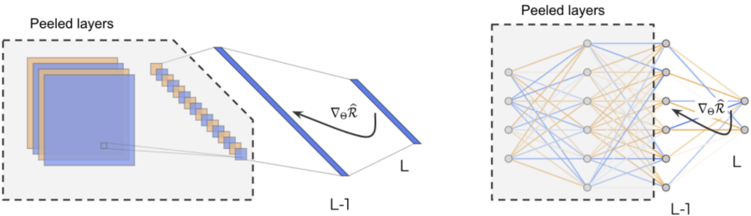


Figure 4: Unconstrained Features Model for CNN (left) and MLP (right) [Kothapalli 2023]

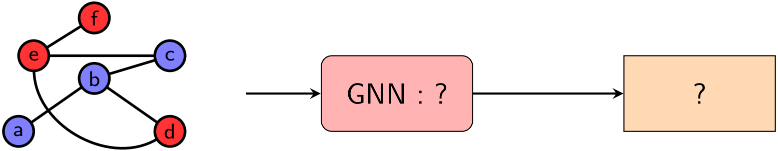
Consider the ERM with MSE loss and regularization as follows:

$$\widehat{\mathcal{R}}(\mathbf{W}, \mathbf{H}) := \frac{1}{2N} \|\mathbf{WH} - \mathbf{Y}\|_F^2 + \frac{\lambda_H}{2} \|\mathbf{H}\|_F^2 + \frac{\lambda_W}{2} \|\mathbf{W}\|_F^2 \tag{6}$$

This setup has been studied extensively by previous works (see references in paper) and has been shown that any minimizer $(\mathbf{W}^*, \mathbf{H}^*)$ exhibits neural collapse.

What if structural connectivity exists between data points?

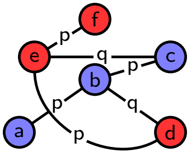
- How can we modify the UFM in graph settings?
- Do GNNs exhibit NC?



- We consider the task of detecting communities/clusters in sparse Symmetric Stochastic Block Model (SSBM) graphs.
- SSBM graphs are random graphs where nodes belonging to the same cluster are connected with a probability p and nodes belonging to different clusters are connected with probability q.
- We sample K random SSBM graphs $\{\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)\}_{k=1}^{K}$, each with N nodes, C clusters, $p = \frac{a \log N}{N}$, $q = \frac{b \log N}{N}$ (regime of exact recovery).

- We consider the task of detecting communities/clusters in sparse Symmetric Stochastic Block Model (SSBM) graphs.
- SSBM graphs are random graphs where nodes belonging to the same cluster are connected with a probability p and nodes belonging to different clusters are connected with probability q.
- We sample K random SSBM graphs $\{\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)\}_{k=1}^{K}$, each with N nodes, C clusters, $p = \frac{a \log N}{N}$, $q = \frac{b \log N}{N}$ (regime of exact recovery).

- We consider the task of detecting communities/clusters in sparse Symmetric Stochastic Block Model (SSBM) graphs.
- SSBM graphs are random graphs where nodes belonging to the same cluster are connected with a probability p and nodes belonging to different clusters are connected with probability q.
- We sample K random SSBM graphs $\{\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)\}_{k=1}^{K}$, each with N nodes, C clusters, $p = \frac{a \log N}{N}, q = \frac{b \log N}{N}$ (regime of exact recovery).



- For a GNN ψ_Θ, the ERM for supervised community detection can be given as:

$$\widehat{\mathbf{R}} = \min_{\Theta} \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}(\psi_\Theta(\mathcal{G}_k), y_k) + \frac{\lambda}{2} \|\Theta\|_F^2 , \tag{7}$$

where $\mathcal{L}$ is based on MSE:

$$\mathcal{L}(\psi_\Theta(\mathcal{G}_k), y_k) = \min_{\pi \in S_C} \frac{1}{2N} \left\| \psi_\Theta\left(\mathcal{G}_k\right) - \pi\left(y_k\left(\mathcal{V}_k\right)\right)\right\|_2^2 . \tag{8}$$

- The performance is measured using "overlap":

$$\mathrm{overlap}(\hat{y}, y) := \max_{\pi \in S_C} \left(\frac{1}{N} \sum_{i=1}^{N} \delta_{\hat{y}(v_i), \pi(y(v_i))} - \frac{1}{C} \right) / \left(1 - \frac{1}{C}\right) \tag{9}$$

Here π indicates permutations over the labels (communities).

Supervised community detection with GNNs

- For a GNN ψ_Θ, the ERM for supervised community detection can be given as:

$$\widehat{\mathbf{R}} = \min_{\Theta} \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}(\psi_\Theta(\mathcal{G}_k), y_k) + \frac{\lambda}{2} \|\Theta\|_F^2 , \tag{7}$$

where $\mathcal{L}$ is based on MSE:

$$\mathcal{L}(\psi_\Theta(\mathcal{G}_k), y_k) = \min_{\pi \in S_C} \frac{1}{2N} \left\| \psi_\Theta \left(\mathcal{G}_k \right) - \pi \left(y_k \left(\mathcal{V}_k \right) \right) \right\|_2^2 . \tag{8}$$

- The performance is measured using "overlap":

$$\mathrm{overlap}(\hat{y}, y) := \max_{\pi \in S_C} \left(\frac{1}{N} \sum_{i=1}^{N} \delta_{\hat{y}(v_i), \pi(y(v_i))} - \frac{1}{C} \right) / \left(1 - \frac{1}{C} \right) \tag{9}$$

Here π indicates permutations over the labels (communities).

- For a GNN ψ_Θ, the ERM for supervised community detection can be given as:

$$\widehat{\mathbf{R}} = \min_{\Theta} \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}(\psi_\Theta(\mathcal{G}_k), y_k) + \frac{\lambda}{2} \|\Theta\|_F^2 \,, \tag{7}$$

where $\mathcal{L}$ is based on MSE:

$$\mathcal{L}(\psi_\Theta(\mathcal{G}_k), y_k) = \min_{\pi \in S_C} \frac{1}{2N} \|\psi_\Theta(\mathcal{G}_k) - \pi(y_k(\mathcal{V}_k))\|_2^2 \,. \tag{8}$$

- The performance is measured using "overlap":

$$\mathrm{overlap}(\hat{y}, y) := \max_{\pi \in S_C} \left(\frac{1}{N} \sum_{i=1}^{N} \delta_{\hat{y}(v_i), \pi(y(v_i))} - \frac{1}{C} \right) / \left(1 - \frac{1}{C} \right) \tag{9}$$

Here π indicates permutations over the labels (communities).

- For a GNN $\psi_{\Theta}^{\mathcal{F}}$ with L layers, the node features $\mathbf{H}_k^{(l)} \in \mathbb{R}^{d_l \times N}$ at layer $l \in [L]$ is given by:

$$\begin{aligned} \mathbf{X}_k^{(l)} &= \mathbf{W}_1^{(l)} \mathbf{H}_k^{(l-1)} + \mathbf{W}_2^{(l)} \mathbf{H}_k^{(l-1)} \widehat{\mathbf{A}}_k, \\ \mathbf{H}_k^{(l)} &= \sigma(\mathbf{X}_k^{(l)}), \end{aligned} \tag{10}$$

  where $\mathbf{H}_k^{(0)} = \mathbf{X}_k$, and $\sigma(\cdot)$ represents a point-wise activation function such as ReLU. $\mathbf{W}_1^{(l)}, \mathbf{W}_2^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ are the weight matrices and $\widehat{\mathbf{A}}_k = \mathbf{A}_k \mathbf{D}_k^{-1}$ is the normalized adjacency matrix, also known as the random-walk matrix.

- A simpler variant $\psi_{\Theta}^{\mathcal{F}'}$ is given by:

$$\begin{aligned} \mathbf{X}_k^{(l)} &= \mathbf{W}_2^{(l)} \mathbf{H}_k^{(l-1)} \widehat{\mathbf{A}}_k, \\ \mathbf{H}_k^{(l)} &= \sigma(\mathbf{X}_k^{(l)}). \end{aligned} \tag{11}$$

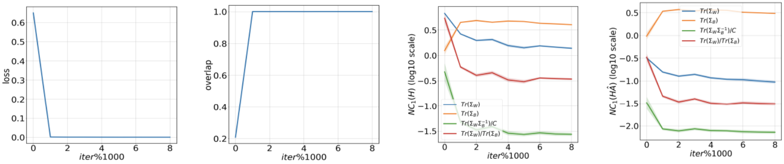
- For a GNN $\psi_{\Theta}^{\mathcal{F}}$ with L layers, the node features $\mathbf{H}_k^{(l)} \in \mathbb{R}^{d_l \times N}$ at layer $l \in [L]$ is given by:

$$\begin{aligned} \mathbf{X}_k^{(l)} &= \mathbf{W}_1^{(l)} \mathbf{H}_k^{(l-1)} + \mathbf{W}_2^{(l)} \mathbf{H}_k^{(l-1)} \widehat{\mathbf{A}}_k, \\ \mathbf{H}_k^{(l)} &= \sigma(\mathbf{X}_k^{(l)}), \end{aligned} \tag{10}$$

  where $\mathbf{H}_k^{(0)} = \mathbf{X}_k$, and $\sigma(\cdot)$ represents a point-wise activation function such as ReLU. $\mathbf{W}_1^{(l)}, \mathbf{W}_2^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ are the weight matrices and $\widehat{\mathbf{A}}_k = \mathbf{A}_k \mathbf{D}_k^{-1}$ is the normalized adjacency matrix, also known as the random-walk matrix.
- A simpler variant $\psi_{\Theta}^{\mathcal{F}'}$ is given by:

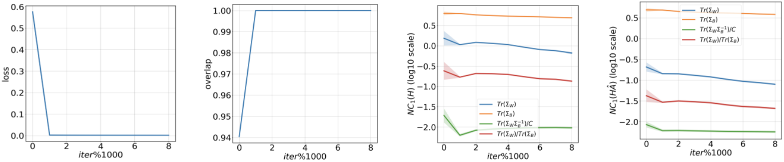
$$\begin{aligned} \mathbf{X}_k^{(l)} &= \mathbf{W}_2^{(l)} \mathbf{H}_k^{(l-1)} \widehat{\mathbf{A}}_k, \\ \mathbf{H}_k^{(l)} &= \sigma(\mathbf{X}_k^{(l)}). \end{aligned} \tag{11}$$

Experimental results: GNN



(a) loss (b) overlap (c) NC1 for $\mathbf{H}$ (d) NC1 for $\mathbf{H}\widehat{\mathbf{A}}$

Figure 5: GNN $\psi_\Theta^{\mathcal{F}}$: Illustration of loss, overlap, and $\mathcal{NC}_1$ plots for $\mathbf{H}, \mathbf{H}\widehat{\mathbf{A}}$ during training.



(a) loss (b) overlap (c) NC1: $\mathbf{H}$ (d) NC1: $\mathbf{H}\widehat{\mathbf{A}}$

Figure 6: GNN $\psi_\Theta^{\mathcal{F}'}$: Illustration of loss, overlap, and $\mathcal{NC}_1$ plots for $\mathbf{H}, \mathbf{H}\widehat{\mathbf{A}}$ during training.

The extent of reduction in NC1 is ‘less’ when compared to the DNN case!

By treating $\{\mathbf{H}_k^{(L-1)}\}_{k=1}^K$ as freely optimizable variables, the empirical risk based on the gUFM can be formulated as follows:

$$\widehat{\mathcal{R}}^{\mathcal{F}'}(\mathbf{W}_2, \{\mathbf{H}_k\}_{k=1}^K) := \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{2N}\left\|\mathbf{W}_2\mathbf{H}_k\widehat{\mathbf{A}}_k - \mathbf{Y}\right\|_F^2 + \frac{\lambda_{H_k}}{2}\|\mathbf{H}_k\|_F^2\right) + \frac{\lambda_{W_2}}{2}\|\mathbf{W}_2\|_F^2 \tag{12}$$

Theorem 3.1

Consider the gUFM with $K = 1$ and denote the fraction of neighbors of node $v_{c,i}$ that belong to class c' as $s_{cc',i} = \frac{|\mathcal{N}_{c'}(v_{c,i})|}{|\mathcal{N}(v_{c,i})|}$. Let the condition **C** based on $s_{cc',i}$ be given by:

$$(s_{c1,1}, \cdots, s_{cC,1}) = \cdots = (s_{c1,n}, \cdots, s_{cC,n}), \quad \forall c \in [C]. \tag{C}$$

If a graph $\mathcal{G}$ satisfies condition **C**, then there exist minimizers of the gUFM that are collapsed (w.r.t NC1). Conversely, when either $\sqrt{\lambda_H \lambda_{W_2}} = 0$, or $\sqrt{\lambda_H \lambda_{W_2}} > 0$ and G is regular (so that $\widehat{\mathbf{A}} = \widehat{\mathbf{A}}^\top$), if there exists a collapsed non-degenerate minimizer of gUFM, then condition **C** necessarily holds.

Structural condition for collapsed minimizers

By treating $\{\mathbf{H}_k^{(L-1)}\}_{k=1}^K$ as freely optimizable variables, the empirical risk based on the gUFM can be formulated as follows:

$$\widehat{\mathcal{R}}^{\mathcal{F}'}(\mathbf{W}_2, \{\mathbf{H}_k\}_{k=1}^K) := \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{2N}\left\|\mathbf{W}_2\mathbf{H}_k\widehat{\mathbf{A}}_k - \mathbf{Y}\right\|_F^2 + \frac{\lambda_{H_k}}{2}\|\mathbf{H}_k\|_F^2\right) + \frac{\lambda_{W_2}}{2}\|\mathbf{W}_2\|_F^2 \tag{12}$$

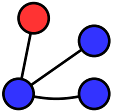
Theorem 3.1

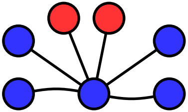
Consider the gUFM with $K = 1$ and denote the fraction of neighbors of node $v_{c,i}$ that belong to class c' as $s_{cc',i} = \frac{|\mathcal{N}_{c'}(v_{c,i})|}{|\mathcal{N}(v_{c,i})|}$. Let the condition **C** based on $s_{cc',i}$ be given by:

$$(s_{c1,1}, \cdots, s_{cC,1}) = \cdots = (s_{c1,n}, \cdots, s_{cC,n}), \quad \forall c \in [C]. \tag{C}$$

If a graph $\mathcal{G}$ satisfies condition **C**, then there exist minimizers of the gUFM that are collapsed (w.r.t NC1). Conversely, when either $\sqrt{\lambda_H\lambda_{W_2}} = 0$, or $\sqrt{\lambda_H\lambda_{W_2}} > 0$ and G is regular (so that $\widehat{\mathbf{A}} = \widehat{\mathbf{A}}^\top$), if there exists a collapsed non-degenerate minimizer of gUFM, then condition **C** necessarily holds.

- Homophilic neighborhoods ($p > q$) satisfying **cond (C)**.

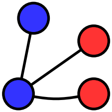
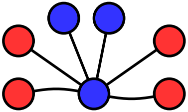




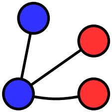
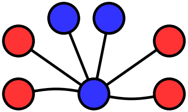
- Heterophilic neighborhoods ($q > p$) satisfying **cond (C)**.

- Note that the $\widehat{\mathbf{A}} = \widehat{\mathbf{A}}^\top$ condition is only an artifact of the proof and not a blocker for empirical analysis.
- Previous works (for ex: Ma et.al) have empirically shown good GNN performance on heterophilic graphs with structure approximately satisfying cond **(C)**. We provide an optimization-based theory for such behaviour.

- Homophilic neighborhoods ($p > q$) satisfying **cond (C)**.

- Heterophilic neighborhoods ($q > p$) satisfying **cond (C)**.

- Note that the $\widehat{\mathbf{A}} = \widehat{\mathbf{A}}^\top$ condition is only an artifact of the proof and not a blocker for empirical analysis.
- Previous works (for ex: Ma et.al) have empirically shown good GNN performance on heterophilic graphs with structure approximately satisfying cond **(C)**. We provide an optimization-based theory for such behaviour.

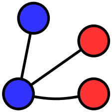
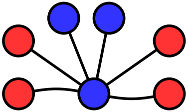
- Homophilic neighborhoods ($p > q$) satisfying **cond (C)**.

- Heterophilic neighborhoods ($q > p$) satisfying **cond (C)**.

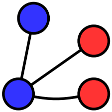
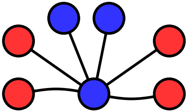
- Note that the $\widehat{\mathbf{A}} = \widehat{\mathbf{A}}^\top$ condition is only an artifact of the proof and not a blocker for empirical analysis.
- Previous works (for ex: Ma et.al) have empirically shown good GNN performance on heterophilic graphs with structure approximately satisfying cond **(C)**. We provide an optimization-based theory for such behaviour.

cond (C): graph view

- Homophilic neighborhoods ($p > q$) satisfying **cond (C)**.

- Heterophilic neighborhoods ($q > p$) satisfying **cond (C)**.

- Note that the $\widehat{\mathbf{A}} = \widehat{\mathbf{A}}^{\top}$ condition is only an artifact of the proof and not a blocker for empirical analysis.
- Previous works (for ex: Ma et.al) have empirically shown good GNN performance on heterophilic graphs with structure approximately satisfying cond **(C)**. We provide an optimization-based theory for such behaviour.

- Recall that the computation graph is defined by $\widehat{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$.
- The value

$$s_{cc',i} = \frac{|\mathcal{N}_{c'}(v_{c,i})|}{|\mathcal{N}(v_{c,i})|}$$

represents the **sum of the column slice** corresponding to neighbors from class c' for a node $v_{c,i}$.
- For ex: Let $C = 2$ with n nodes in each class. Consider the column shown below corresponds to a node from class $c = 1$.

$$\widehat{\mathbf{A}} = \begin{bmatrix} \cdots & \square & \cdots \\ \cdots & \square & \cdots \end{bmatrix}, \quad \implies \mathbf{1}^\top \square = s_{11}, \mathbf{1}^\top \square = s_{12}, \forall i \in [n]$$

- The same applies to all nodes in class $c = 2$. Straightforward to extend this to $C > 2$ settings.

- Recall that the computation graph is defined by $\widehat{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$.
- The value

$$s_{cc',i} = \frac{|\mathcal{N}_{c'}(v_{c,i})|}{|\mathcal{N}(v_{c,i})|}$$

represents the **sum of the column slice** corresponding to neighbors from class c' for a node $v_{c,i}$.

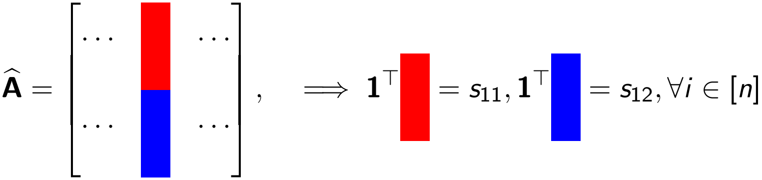
- For ex: Let $C = 2$ with n nodes in each class. Consider the column shown below corresponds to a node from class $c = 1$.

$$\widehat{\mathbf{A}} = \begin{bmatrix} \cdots & & \cdots \\ \cdots & & \cdots \end{bmatrix}, \quad \implies \mathbf{1}^\top \; = s_{11}, \mathbf{1}^\top \; = s_{12}, \forall i \in [n]$$

- The same applies to all nodes in class $c = 2$. Straightforward to extend this to $C > 2$ settings.

cond (C): matrix view

- Recall that the computation graph is defined by $\widehat{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$.
- The value
$$s_{cc',i} = \frac{|\mathcal{N}_{c'}(v_{c,i})|}{|\mathcal{N}(v_{c,i})|}$$
represents the **sum of the column slice** corresponding to neighbors from class c' for a node $v_{c,i}$.
- For ex: Let $C = 2$ with n nodes in each class. Consider the column shown below corresponds to a node from class $c = 1$.

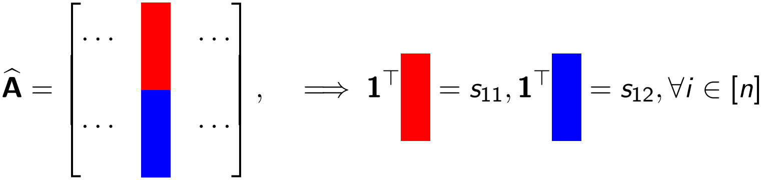


- The same applies to all nodes in class $c = 2$. Straightforward to extend this to $C > 2$ settings.

- Recall that the computation graph is defined by $\widehat{\mathbf{A}} = \mathbf{A}\mathbf{D}^{-1}$.
- The value

$$s_{cc',i} = \frac{|\mathcal{N}_{c'}(v_{c,i})|}{|\mathcal{N}(v_{c,i})|}$$

  represents the **sum of the column slice** corresponding to neighbors from class c' for a node $v_{c,i}$.
- For ex: Let $C = 2$ with n nodes in each class. Consider the column shown below corresponds to a node from class $c = 1$.



- The same applies to all nodes in class $c = 2$. Straightforward to extend this to $C > 2$ settings.

Minimizer Conjecture

Conjecture 3.1

Consider the gUFM with $K = 1$ and condition **C** as stated in theorem 3.1. The minimizers of the gUFM are collapsed (w.r.t NC1) iff the graph $\mathcal{G}$ satisfies condition **C**.

What is the probability of sampling a random SSBM graph that satisfies cond (C)? A: practically 0

Theorem 3.2

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be drawn from SSBM(N, C, p, q). For $N >> C$, we have

$$\mathbb{P}\left(\mathcal{G} \text{ obeys } \mathbf{C}\right) < \left(\sum_{t=0}^{n} \left[\binom{n}{t} q^t (1-q)^{n-t}\right]^n\right)^{\frac{C(C-1)}{2}}. \tag{13}$$

Numerical example. Let's consider a setting with $C = 2, N = 1000, p = 0.025, q = 0.0017$. This gives us $\mathbb{P}(\mathcal{G} \text{ obeys } \mathbf{C}) < 2.18 \times 10^{-188}$.

What is the probability of sampling a random SSBM graph that satisfies cond (C)? A: practically 0

Theorem 3.2

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be drawn from SSBM(N, C, p, q). For $N >> C$, we have

$$\mathbb{P}\left(\mathcal{G} \text{ obeys } \mathbf{C}\right) < \left(\sum_{t=0}^{n} \left[\binom{n}{t} q^t (1-q)^{n-t}\right]^n\right)^{\frac{C(C-1)}{2}}. \quad (13)$$

Numerical example. Let's consider a setting with $C = 2, N = 1000, p = 0.025, q = 0.0017$. This gives us $\mathbb{P}(\mathcal{G} \text{ obeys } \mathbf{C}) < 2.18 \times 10^{-188}$.

What is the probability of sampling a random SSBM graph that satisfies cond (C)? A: practically 0

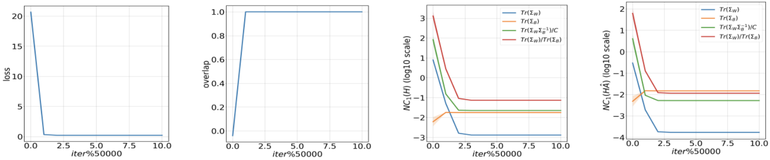
Theorem 3.2

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be drawn from SSBM(N, C, p, q). For $N >> C$, we have

$$\mathbb{P}\left(\mathcal{G} \text{ obeys } \mathbf{C}\right) < \left(\sum_{t=0}^{n}\left[\binom{n}{t} q^t (1-q)^{n-t}\right]^n\right)^{\frac{C(C-1)}{2}}. \qquad (13)$$

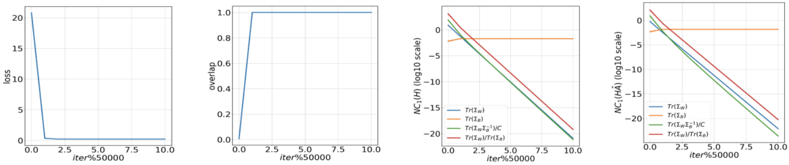
Numerical example. Let's consider a setting with $C = 2, N = 1000, p = 0.025, q = 0.0017$. This gives us $\mathbb{P}(\mathcal{G} \text{ obeys } \mathbf{C}) < 2.18 \times 10^{-188}$.

Experimental results: gUFM



(a) loss (b) overlap (c) NC1: $\mathbf{H}$ (d) NC1: $\mathbf{H}\widehat{\mathbf{A}}$

Figure 7: gUFM for $\psi_{\Theta}^{\mathcal{F}'}$: Illustration of loss, overlap, and $\mathcal{NC}_1$ plots for $\mathbf{H}$, $\mathbf{H}\widehat{\mathbf{A}}$ during training on 10 SSBM graphs which do not satisfy condition $\mathbf{C}$.



(a) loss (b) overlap (c) NC1: $\mathbf{H}$ (d) NC1: $\mathbf{H}\widehat{\mathbf{A}}$

Figure 8: gUFM for $\psi_{\Theta}^{\mathcal{F}'}$: Illustration of loss, overlap, and $\mathcal{NC}_1$ plots for $\mathbf{H}$, $\mathbf{H}\widehat{\mathbf{A}}$ during training on 10 SSBM graphs which satisfies condition $\mathbf{C}$.

To understand this "partial collapse" behaviour, we analyze the gradient flow along the "central path" — i.e., when $\mathbf{W}_2 = \mathbf{W}_2^*(\mathbf{H})$ is the optimal minimizer of $\widehat{\mathcal{R}}^{\mathcal{F}'}(\mathbf{W}_2, \mathbf{H})$ w.r.t. $\mathbf{W}_2$, as follows

$$\frac{d\mathbf{H}_t}{dt} = -\nabla \widehat{\mathcal{R}}^{\mathcal{F}'}(\mathbf{W}_2^*(\mathbf{H}_t), \mathbf{H}_t). \tag{14}$$

Theorem 3.3

Let $K = 1$, $C = 2$ and $\lambda_{W_2} > 0$. There exist $\alpha > 0$ and $E > 0$, such that for $0 < \lambda_H < \alpha$ and $0 < \|\mathbf{E}\| < E$, along the gradient flow stated in (14) associated with the graph $\widehat{\mathbf{A}} = \mathbb{E}\widehat{\mathbf{A}} + \mathbf{E}$, we have that: (1) $\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{H}_t))$ decreases, and (2) $\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{H}_t))$ increases. Accordingly, $\widetilde{\mathcal{NC}}_1(\mathbf{H}_t)$ decreases.

To understand this "partial collapse" behaviour, we analyze the gradient flow along the "central path" — i.e., when $\mathbf{W}_2 = \mathbf{W}_2^*(\mathbf{H})$ is the optimal minimizer of $\widehat{\mathcal{R}}^{\mathcal{F}'}(\mathbf{W}_2, \mathbf{H})$ w.r.t. $\mathbf{W}_2$, as follows

$$\frac{d\mathbf{H}_t}{dt} = -\nabla \widehat{\mathcal{R}}^{\mathcal{F}'}(\mathbf{W}_2^*(\mathbf{H}_t), \mathbf{H}_t). \tag{14}$$

Theorem 3.3

Let $K = 1$, $C = 2$ and $\lambda_{W_2} > 0$. There exist $\alpha > 0$ and $E > 0$, such that for $0 < \lambda_H < \alpha$ and $0 < \|\mathbf{E}\| < E$, along the gradient flow stated in (14) associated with the graph $\widehat{\mathbf{A}} = \mathbb{E}\widehat{\mathbf{A}} + \mathbf{E}$, we have that: (1) $\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{H}_t))$ decreases, and (2) $\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{H}_t))$ increases. Accordingly, $\widetilde{\mathcal{NC}}_1(\mathbf{H}_t)$ decreases.

Oversmoothing

(Rusch et al.): For an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *with* $|\mathcal{V}| = N$ *and* l*-th layer hidden features* $\mathbf{H}^l \in \mathbb{R}^{d_l \times N}$*, a function* $\mu : \mathbb{R}^{d_l \times N} \to \mathbb{R}_{\geq 0}$ *is called a node-similarity measure if:*

1. $\exists \mathbf{c} \in \mathbb{R}^{d_l}$ *with* $\mathbf{H}_i = \mathbf{c}$ *for all nodes* $i \in \mathcal{V} \iff \mu(\mathbf{H}) = 0$*, for* $\mathbf{H} \in \mathbb{R}^{d_l \times N}$
2. $\mu(\mathbf{H} + \mathbf{T}) \leq \mu(\mathbf{H}) + \mu(\mathbf{T})$*, for all* $\mathbf{H}, \mathbf{T} \in \mathbb{R}^{d_l \times N}$*.*

Oversmoothing with respect to μ *is now defined as the layer-wise exponential convergence of the node-similarity measure* μ *to zero*

$$\mu(\mathbf{H}^l) \leq C_1 e^{-C_2 l}, \text{ for } l = 1, \cdots, L \text{ with some constants } C_1, C_2 > 0.$$

- Oversmoothing $\implies \mathbf{\Sigma}_W(\mathbf{H}^{L-1}), \mathbf{\Sigma}_B(\mathbf{H}^{L-1}) \to 0$.
- NC $\implies \mathbf{\Sigma}_W(\mathbf{H}^{L-1})$ decreases, and $\mathbf{\Sigma}_B(\mathbf{H}^{L-1})$ is bounded from below!!

Oversmoothing

(Rusch et al.): For an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = N$ and l-th layer hidden features $\mathbf{H}^l \in \mathbb{R}^{d_l \times N}$, a function $\mu : \mathbb{R}^{d_l \times N} \to \mathbb{R}_{\geq 0}$ is called a node-similarity measure if:

1. $\exists \mathbf{c} \in \mathbb{R}^{d_l}$ *with* $\mathbf{H}_i = \mathbf{c}$ *for all nodes* $i \in \mathcal{V} \iff \mu(\mathbf{H}) = 0$, *for* $\mathbf{H} \in \mathbb{R}^{d_l \times N}$
2. $\mu(\mathbf{H} + \mathbf{T}) \leq \mu(\mathbf{H}) + \mu(\mathbf{T})$, *for all* $\mathbf{H}, \mathbf{T} \in \mathbb{R}^{d_l \times N}$.

Oversmoothing with respect to μ is now defined as the layer-wise exponential convergence of the node-similarity measure μ to zero

$$\mu(\mathbf{H}^l) \leq C_1 e^{-C_2 l}, \text{ for } l = 1, \cdots, L \text{ with some constants } C_1, C_2 > 0.$$

- Oversmoothing $\implies \mathbf{\Sigma}_W(\mathbf{H}^{L-1}), \mathbf{\Sigma}_B(\mathbf{H}^{L-1}) \to 0$.
- NC $\implies \mathbf{\Sigma}_W(\mathbf{H}^{L-1})$ decreases, and $\mathbf{\Sigma}_B(\mathbf{H}^{L-1})$ is bounded from below!!

Till now, we have analyzed the training phase of GNNs. But, what about inference? What can we say about the NC properties of features across depth?

As a baseline during inference, we perform spectral clustering using projected power iterations on the Normalized Laplacian (NL) and Bethe-Hessian (BH) matrices to approximate the Fiedler vector.

$$\mathrm{NL}(\mathcal{G}) = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}, \tag{15}$$

$$\mathrm{BH}(\mathcal{G}, r) = (r^2 - 1)\mathbf{I} - r\mathbf{A} + \mathbf{D}, \tag{16}$$

where $r \in \mathbb{R}$ is the BH scaling factor. Now, by treating $\mathbf{B}$ to be either NL or BH matrix, a projected power iteration to estimate the second largest eigenvector of $\widetilde{\mathbf{B}} = \|\mathbf{B}\|\,\mathbf{I} - \mathbf{B}$ is given by:

$$\mathbf{x}^{(l)} = \widetilde{\mathbf{B}}\mathbf{w}^{(l-1)}, \quad \text{where} \quad \mathbf{w}^{(l-1)} = \frac{\mathbf{x}^{(l-1)} - \langle \mathbf{x}^{(l-1)}, \mathbf{v}\rangle \mathbf{v}}{\left\|\mathbf{x}^{(l-1)} - \langle \mathbf{x}^{(l-1)}, \mathbf{v}\rangle \mathbf{v}\right\|_2}, \tag{17}$$

with the vector $\mathbf{v} \in \mathbb{R}^N$ denoting the largest eigenvector of $\widetilde{\mathbf{B}}$. Thus, we start with a random normal vector $\mathbf{w}^0 \in \mathbb{R}^N$ and iteratively compute the feature vector $\mathbf{x}^{(l)} \in \mathbb{R}^N$.

As a baseline during inference, we perform spectral clustering using projected power iterations on the Normalized Laplacian (NL) and Bethe-Hessian (BH) matrices to approximate the Fiedler vector.

$$\mathrm{NL}(\mathcal{G}) = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}, \tag{15}$$

$$\mathrm{BH}(\mathcal{G}, r) = (r^2 - 1)\mathbf{I} - r\mathbf{A} + \mathbf{D}, \tag{16}$$

where $r \in \mathbb{R}$ is the BH scaling factor. Now, by treating $\mathbf{B}$ to be either NL or BH matrix, a projected power iteration to estimate the second largest eigenvector of $\widetilde{\mathbf{B}} = \|\mathbf{B}\|\,\mathbf{I} - \mathbf{B}$ is given by:

$$\mathbf{x}^{(l)} = \widetilde{\mathbf{B}}\mathbf{w}^{(l-1)}, \quad \text{where} \quad \mathbf{w}^{(l-1)} = \frac{\mathbf{x}^{(l-1)} - \langle \mathbf{x}^{(l-1)}, \mathbf{v}\rangle \mathbf{v}}{\left\|\mathbf{x}^{(l-1)} - \langle \mathbf{x}^{(l-1)}, \mathbf{v}\rangle \mathbf{v}\right\|_2}, \tag{17}$$

with the vector $\mathbf{v} \in \mathbb{R}^N$ denoting the largest eigenvector of $\widetilde{\mathbf{B}}$. Thus, we start with a random normal vector $\mathbf{w}^0 \in \mathbb{R}^N$ and iteratively compute the feature vector $\mathbf{x}^{(l)} \in \mathbb{R}^N$.

As a baseline during inference, we perform spectral clustering using projected power iterations on the Normalized Laplacian (NL) and Bethe-Hessian (BH) matrices to approximate the Fiedler vector.

$$\mathrm{NL}(\mathcal{G}) = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}, \tag{15}$$

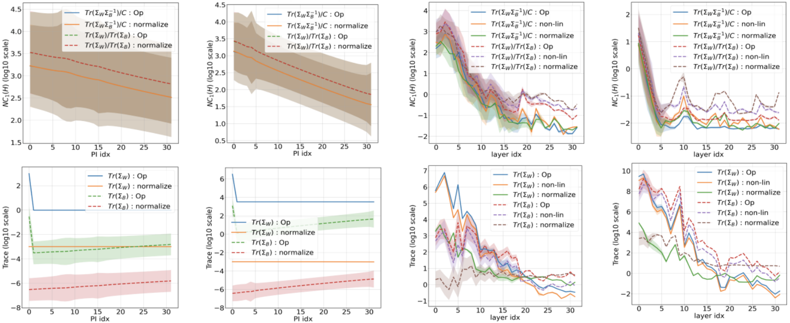
$$\mathrm{BH}(\mathcal{G}, r) = (r^2 - 1)\mathbf{I} - r\mathbf{A} + \mathbf{D}, \tag{16}$$

where $r \in \mathbb{R}$ is the BH scaling factor. Now, by treating $\mathbf{B}$ to be either NL or BH matrix, a projected power iteration to estimate the second largest eigenvector of $\widetilde{\mathbf{B}} = \|\mathbf{B}\|\,\mathbf{I} - \mathbf{B}$ is given by:

$$\mathbf{x}^{(l)} = \widetilde{\mathbf{B}}\mathbf{w}^{(l-1)}, \quad \text{where} \quad \mathbf{w}^{(l-1)} = \frac{\mathbf{x}^{(l-1)} - \langle \mathbf{x}^{(l-1)}, \mathbf{v}\rangle \mathbf{v}}{\left\| \mathbf{x}^{(l-1)} - \langle \mathbf{x}^{(l-1)}, \mathbf{v}\rangle \mathbf{v} \right\|_2}, \tag{17}$$

with the vector $\mathbf{v} \in \mathbb{R}^N$ denoting the largest eigenvector of $\widetilde{\mathbf{B}}$. Thus, we start with a random normal vector $\mathbf{w}^0 \in \mathbb{R}^N$ and iteratively compute the feature vector $\mathbf{x}^{(l)} \in \mathbb{R}^N$.

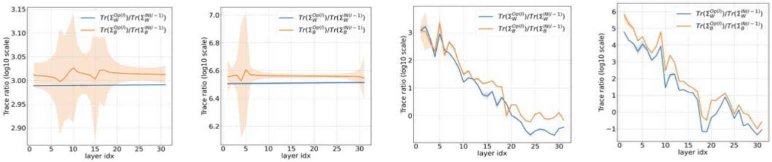
Experimental results



(a) Normalized Laplacian (b) Bethe Hessian (c) GNN $\psi_{\Theta}^{\mathcal{F}}$ (d) GNN $\psi_{\Theta}^{\mathcal{F}'}$

Figure 9: $\mathcal{NC}_1(\mathbf{H}), \widetilde{\mathcal{NC}}_1(\mathbf{H})$ metrics (top) and traces of covariance matrices (bottom) across projected power iterations for NL and BH (a,b), and across layers for GNNs $\psi_{\Theta}^{\mathcal{F}}$ and $\psi_{\Theta}^{\mathcal{F}'}$ (c,d).

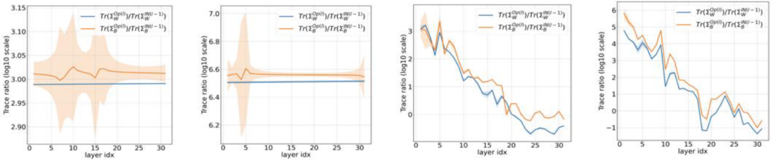
Effect of graph convolutions



(a) Normalized Laplacian (b) Bethe Hessian (c) GNN $\psi_\Theta^{\mathcal{F}}$ (d) GNN $\psi_\Theta^{\mathcal{F}'}$

Figure 10: Ratio of traces of covariance matrices across projected power iterations for NL and BH (a,b), and across layers for GNNs $\psi_\Theta^{\mathcal{F}}$ and $\psi_\Theta^{\mathcal{F}'}$ (c,d).

- Recall the layer for $\psi_\Theta^{\mathcal{F}}$: $\mathbf{X}_k^{(l)} = \mathbf{W}_1^{(l)}\mathbf{H}_k^{(l-1)} + \mathbf{W}_2^{(l)}\mathbf{H}_k^{(l-1)}\widehat{\mathbf{A}}_k$
- We consider the case of $C = 2$ (without loss of generality) and assume that the $(l-1)^{th}$-layer features $\mathbf{H}^{(l-1)}$ of nodes belonging to class $c = 1, 2$ are drawn from distributions $\mathcal{D}_1, \mathcal{D}_2$.
- Let $\boldsymbol{\mu}_1^{(l-1)}, \boldsymbol{\mu}_2^{(l-1)} \in \mathbb{R}^{d_{l-1}}$ and $\boldsymbol{\Sigma}_1^{(l-1)}, \boldsymbol{\Sigma}_2^{(l-1)} \in \mathbb{R}^{d_{l-1}\times d_{l-1}}$ as their mean vectors and covariance matrices of $\mathcal{D}_1, \mathcal{D}_2$.

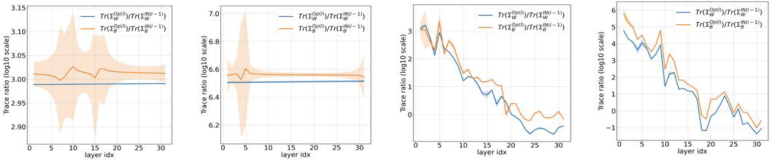


(a) Normalized Laplacian (b) Bethe Hessian (c) GNN $\psi_{\Theta}^{\mathcal{F}}$ (d) GNN $\psi_{\Theta}^{\mathcal{F}'}$

Figure 10: Ratio of traces of covariance matrices across projected power iterations for NL and BH (a,b), and across layers for GNNs $\psi_{\Theta}^{\mathcal{F}}$ and $\psi_{\Theta}^{\mathcal{F}'}$ (c,d).

- Recall the layer for $\psi_{\Theta}^{\mathcal{F}}$: $\mathbf{X}_k^{(l)} = \mathbf{W}_1^{(l)}\mathbf{H}_k^{(l-1)} + \mathbf{W}_2^{(l)}\mathbf{H}_k^{(l-1)}\widehat{\mathbf{A}}_k$
- We consider the case of $C = 2$ (without loss of generality) and assume that the $(l-1)^{th}$-layer features $\mathbf{H}^{(l-1)}$ of nodes belonging to class $c = 1, 2$ are drawn from distributions $\mathcal{D}_1, \mathcal{D}_2$.
- Let $\boldsymbol{\mu}_1^{(l-1)}, \boldsymbol{\mu}_2^{(l-1)} \in \mathbb{R}^{d_{l-1}}$ and $\boldsymbol{\Sigma}_1^{(l-1)}, \boldsymbol{\Sigma}_2^{(l-1)} \in \mathbb{R}^{d_{l-1} \times d_{l-1}}$ as their mean vectors and covariance matrices of $\mathcal{D}_1, \mathcal{D}_2$.

Effect of graph convolutions



(a) Normalized Laplacian (b) Bethe Hessian (c) GNN $\psi_\Theta^{\mathcal{F}}$ (d) GNN $\psi_\Theta^{\mathcal{F}'}$

Figure 10: Ratio of traces of covariance matrices across projected power iterations for NL and BH (a,b), and across layers for GNNs $\psi_\Theta^{\mathcal{F}}$ and $\psi_\Theta^{\mathcal{F}'}$ (c,d).

- Recall the layer for $\psi_\Theta^{\mathcal{F}}$: $\mathbf{X}_k^{(l)} = \mathbf{W}_1^{(l)}\mathbf{H}_k^{(l-1)} + \mathbf{W}_2^{(l)}\mathbf{H}_k^{(l-1)}\widehat{\mathbf{A}}_k$
- We consider the case of $C = 2$ (without loss of generality) and assume that the $(l-1)^{th}$-layer features $\mathbf{H}^{(l-1)}$ of nodes belonging to class $c = 1, 2$ are drawn from distributions $\mathcal{D}_1, \mathcal{D}_2$.
- Let $\boldsymbol{\mu}_1^{(l-1)}, \boldsymbol{\mu}_2^{(l-1)} \in \mathbb{R}^{d_{l-1}}$ and $\boldsymbol{\Sigma}_1^{(l-1)}, \boldsymbol{\Sigma}_2^{(l-1)} \in \mathbb{R}^{d_{l-1} \times d_{l-1}}$ as their mean vectors and covariance matrices of $\mathcal{D}_1, \mathcal{D}_2$.

Theorem 4.1

Let $C = 2, \lambda_i(\cdot), \lambda_{-i}(\cdot)$ indicate the i^{th} largest and smallest eigenvalue of a matrix, $\beta_1 = \frac{p-q}{p+q}, \beta_2 = \frac{p}{n(p+q)}, \beta_3 = \frac{p^2+q^2}{n(p+q)^2}$, and

$$\mathbf{T}_W = \mathbf{W}_1^{*(l)\top}\mathbf{W}_1^{*(l)} + \beta_2\left[\mathbf{W}_2^{*(l)\top}\mathbf{W}_1^{*(l)} + \mathbf{W}_1^{*(l)\top}\mathbf{W}_2^{*(l)}\right] + \beta_3\mathbf{W}_2^{*(l)\top}\mathbf{W}_2^{*(l)},$$

$$\mathbf{T}_B = \left(\mathbf{W}_1^{*(l)} + \beta_1\mathbf{W}_2^{*(l)}\right)^\top\left(\mathbf{W}_1^{*(l)} + \beta_1\mathbf{W}_2^{*(l)}\right).$$

Then, the ratios of traces $\frac{\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{X}^{(l)}))}{\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)}))}, \frac{\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{X}^{(l)}))}{\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)}))}$ for layer $l \in \{2, \cdots, L\}$ of a network $\psi_\Theta^{\mathcal{F}}$ are bounded as follows:

$$\frac{\sum_{i=1}^{d_{l-1}}\lambda_{-i}\left(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)})\right)\lambda_i(\mathbf{T}_B)}{\sum_{i=1}^{d_{l-1}}\lambda_i\left(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)})\right)} \leq \frac{\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{X}^{(l)}))}{\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)}))} \leq \frac{\sum_{i=1}^{d_{l-1}}\lambda_i\left(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)})\right)\lambda_i(\mathbf{T}_B)}{\sum_{i=1}^{d_{l-1}}\lambda_i\left(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)})\right)},$$

$$\frac{\sum_{i=1}^{d_{l-1}}\lambda_{-i}\left(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)})\right)\lambda_i(\mathbf{T}_W)}{\sum_{i=1}^{d_{l-1}}\lambda_i\left(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)})\right)} \leq \frac{\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{X}^{(l)}))}{\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)}))} \leq \frac{\sum_{i=1}^{d_{l-1}}\lambda_i\left(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)})\right)\lambda_i(\mathbf{T}_W)}{\sum_{i=1}^{d_{l-1}}\lambda_i\left(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)})\right)}.$$

Takeaway: The presence of $\mathbf{W}_1\mathbf{H}$ in the layer formulation of reduces the rate of reduction of $\frac{\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{X}^{(l)}))}{\mathrm{Tr}(\mathbf{\Sigma}_B(\mathbf{H}^{(l-1)}))}, \frac{\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{X}^{(l)}))}{\mathrm{Tr}(\mathbf{\Sigma}_W(\mathbf{H}^{(l-1)}))}$.

- By adopting a Neural Collapse (NC) perspective, we analyzed both empirically and theoretically the within- and between-class variability of GNN features along *the training epochs and along the layers during inference*.
- We showed that a partial decrease in within-class variability (and NC1 metrics) is present in the GNNs' deepest features but full collapse is not expected in practise.
- We also showed a depthwise decrease in variability metrics, which resembles the case with plain DNNs. Especially, by leveraging the analogy of feature transformation across layers in GNNs and along projected power iterations.
- Shed light on computation graphs that might be suitable for graph-rewiring techniques, addressing oversmoothing and potentially improving generalization on real-world large-scale graphs!

Summary

- By adopting a Neural Collapse (NC) perspective, we analyzed both empirically and theoretically the within- and between-class variability of GNN features along *the training epochs and along the layers during inference*.
- We showed that a partial decrease in within-class variability (and NC1 metrics) is present in the GNNs' deepest features but full collapse is not expected in practise.
- We also showed a depthwise decrease in variability metrics, which resembles the case with plain DNNs. Especially, by leveraging the analogy of feature transformation across layers in GNNs and along projected power iterations.
- Shed light on computation graphs that might be suitable for graph-rewiring techniques, addressing oversmoothing and potentially improving generalization on real-world large-scale graphs!

Summary

- By adopting a Neural Collapse (NC) perspective, we analyzed both empirically and theoretically the within- and between-class variability of GNN features along *the training epochs and along the layers during inference*.
- We showed that a partial decrease in within-class variability (and NC1 metrics) is present in the GNNs' deepest features but full collapse is not expected in practise.
- We also showed a depthwise decrease in variability metrics, which resembles the case with plain DNNs. Especially, by leveraging the analogy of feature transformation across layers in GNNs and along projected power iterations.
- Shed light on computation graphs that might be suitable for graph-rewiring techniques, addressing oversmoothing and potentially improving generalization on real-world large-scale graphs!

Summary

- By adopting a Neural Collapse (NC) perspective, we analyzed both empirically and theoretically the within- and between-class variability of GNN features along *the training epochs and along the layers during inference*.
- We showed that a partial decrease in within-class variability (and NC1 metrics) is present in the GNNs' deepest features but full collapse is not expected in practise.
- We also showed a depthwise decrease in variability metrics, which resembles the case with plain DNNs. Especially, by leveraging the analogy of feature transformation across layers in GNNs and along projected power iterations.
- Shed light on computation graphs that might be suitable for graph-rewiring techniques, addressing oversmoothing and potentially improving generalization on real-world large-scale graphs!

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

Open problems/questions

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

Open problems/questions

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

Open problems/questions

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

Open problems/questions

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

Open problems/questions

1. The connection between over-smoothing and neural collapse is not fully explored.
2. What is an ideal graph rewiring strategy to achieve cond **(C)**?
3. How do neighborhood ratios $s_{cc'}$ affect GNN performance ? Especially, can we leverage cond **(C)** for efficient neighborhood sampling in large-scale graphs?
4. Addressing Conjecture 3.1 on cond **(C)** and minimizers.
5. What can we say about other NC metrics? Especially, how does the graph structure perturb the simplex ETF structure?
6. What about graph classification tasks?
7. Can attention layers learn $\widehat{\mathbf{A}}$ that satisfies cond **(C)**?

THANK YOU!

Code: https://github.com/kvignesh1420/gnn_collapse