Neural Collapse on Graph Transformers

Caleb Stam cstam@ucsd.edu
Eduardo Spiegel espiegel@ucsd.edu
Bill Wang blw002@ucsd.edu
Yusu Wang yusuwang@ucsd.edu

Gal Minshe
gmishne@ucsd.edu

Abstract

Building upon recent theoretical and empirical investigations on neural collapse in Graph Neural Networks (GNNs), we extend this analysis to a wider variety of graph learning models, particularly graph transformers. While prior work established that traditional GNNS exhibit partial collapse behavior dependent on the input graph's structure, we examine whether more expressive architectures with attention mechanisms and enhanced message passing capabilities demonstrate different collapse characteristics. Using a systematic analysis of stochastic block model graphs, we show that graph transformers and , demonstrate . Potential Theoretical analysis take away

For your Quarter 1 Project, you won't link a website, but will link your code. For your Quarter 2 Proposal, you won't link either of these. For your actual Quarter 2 Project, you'll link both of these.

Website: https://abc.github.io/
Code: https://github.com/spiegel21/graph_transformer_collapse/tree/calebs_transformer

1 Introduction

Graph Neural Networks (GNNs) have emerged as a powerful tool for analyzing non-Euclidean data structures, particularly in scenarios where relationships between entities are crucial for understanding the underlying patterns.

1.1 Background

The study of graph neural networks lies in the domain of geometric deep learning (GDL). GDL encompasses deep learning techniques applied to non-Euclidean data structures such as graphs, hypergraphs, simplicial complexes, and manifolds. The field focuses on capturing relationships between entities within a combinatorial structure to learn meaningful representations. GNNs are the most thoroughly studied GDL architecture, and they excel in three primary tasks. First, in node classification the network assigns labels to individual nodes within a graph, such as detecting fraudulent accounts in social networks. Second, in link prediction, the network predicts connections between nodes and sees use applicable in drug interaction studies and recommendation systems. Finally, in graph classification, the network categorizes entire graphs. It is crucial for applications like drug discovery and protein classification.

1.2 Graph Neural Networks

Graph neural networks have seen significant development since their introduction. The foundational work on GCNs by Kipf and Welling (2016) established the framework for modern graph-based deep learning. Subsequent architectures like GAT Veličković et al. (2017) and GIN Xu et al. (2018) have proposed alternative approaches to aggregating and processing graph-structured data.

However, all-message passing frameworks are subject to the problems of oversquashing and oversmoothing. Oversquashing occurs when information from distant nodes becomes compressed as it passes through layers, limiting the model's ability to capture long-range dependencies. This problem was first noted by Alon and Yahav (2020), and was formally justified and analyzed by Topping et al. (2021) through a sensitivity analysis of the Jacobian on node features. Oversmoothing presents another challenge, as deep architectures tend to homogenize node features, making it difficult to distinguish between different classes or nodes Li, Han and Wu (2018). These issues, combined with computational complexity and scalability concerns, can significantly impact GNN performance, particularly when working with large, real-world graphs.

In response to these faults, graph transformers have emerged as a new paradigm for learning on graphs Dwivedi and Bresson (2020); Müller et al. (2023). These networks rely on the encoding of graph structure as features rather than message-passing to communicate structural information. This allows graph transformers to handle long-range interactions directly without losing out on the information granted by relational structure, leading to

greater expressive power Rampášek et al. (2022). Unfortunately, graph transformers struggle with scalability, increasing quadratically in complexity with respect to the number of nodes. Intuitive approaches to countering this, such as substituting a transformer's attention for that of a Performer Choromanski et al. (2020), have been shown to be equivalent to message-passing neural networks with virtual nodes in terms of expressivity Cai et al. (2023).

1.3 Neural Collapse and Graph Learning

The phenomenon of neural collapse, first identified by Papyan et al. Papyan, Han and Donoho (2020), has emerged as a crucial framework for understanding deep neural network behavior during the terminal phase of training - the period after achieving zero training error. During this phase, networks exhibit four statistical and geometric properties. First, features within each class converge to their means. Second, these class means arrange themselves into a simplex equiangular tight frame (ETF) and are maximally separated. Third, the classifier's weights align with these class means, forming an ETF in the dual space. Finally, the network's behavior reduces to the process of choosing the nearest class center of a point's last-layer activation.

Recent work by Kothapalli et al. Kothapalli, Tirer and Bruna (2023) demonstrated that traditional Graph Neural Networks (GNNs) exhibit a lesser degree collapse of collapse when compared to standard neural networks. This behavior was found to be fundamentally constrained by graph topology, with theoretical analysis revealing specific structural conditions necessary for complete collapse. Their work established that the extent of collapse varies significantly based on graph convolutional layers, network depth, and the underlying graph properties.

1.4 Contributions

Our work extends this understanding of neural collapse to graph transformers, providing several key contributions:

- First systematic investigation of neural collapse phenomena in graph transformer architectures
- Analysis of how self-attention mechanisms influence feature collapse compared to traditional message-passing GNNs
- Empirical evidence showing the relationship between attention patterns and the degree of neural collapse
- Theoretical framework connecting transformer attention structures to the structural conditions required for complete collapse

These findings not only advance our understanding of neural collapse in geometric deep learning but also provide practical insights for designing more effective graph transformer architectures. Our results suggest that the self-attention mechanism in graph transformers may offer advantages over traditional message-passing approaches in terms of feature

discrimination and class separation during the terminal phase of training.

2 Methods

Given our 10-week timeline, we decided the most efficient approach is to build upon the existing code from the paper "A Neural Collapse Perspective on Feature Evolution in Graph Neural Networks." Our plan is to integrate a new transformer architecture into the paper's GitHub repository and evaluate its performance using neural collapse (NC) metrics. Alongside our analysis of the graph transformer, we seek to analyze NC across a range of message-passing architectures with the intent of demonstrating the ubiquity of collapse in graph learning.

2.1 Model Architecture

We currently have a few models under consideration: GAT, GIN, and a Graph Transformer.

2.1.1 Graph Attention Network (GAT)

PLACE HOLDER UNTIL WE GET ACTUAL ARCHITECTURE The **Graph Attention Network (GAT)** introduces attention mechanisms to graph-structured data, allowing nodes to weigh the importance of their neighbors' features dynamically. By leveraging multi-head self-attention, GAT effectively captures both local and global relationships, improving performance on node classification and link prediction tasks. The architecture consists of attention layers that aggregate neighbor information with learnable attention coefficients, followed by non-linear activation functions.

PLACE HOLDER UNTIL WE GET ACTUAL ARCHITECTURE

2.1.2 Graph Isomorphism Network (GIN)

The **Graph Isomorphism Network (GIN)** is designed to maximize the representational power of GNNs, achieving expressiveness equivalent to the Weisfeiler-Lehman graph isomorphism test. GIN uses sum-based aggregation functions followed by multi-layer perceptrons (MLPs) to capture structural information within graphs. Its simplicity and strong theoretical foundation make it effective for graph classification tasks, especially in distinguishing non-isomorphic graphs.

2.1.3 Graph Transformer

The graph transformer extends successes in natural language processing (NLP) to the graph domain. In a graph transformer, input data is first augmented by a choice of positional

or structural encoding. These encodings endow each node with information about graph topology that the transformer would otherwise be unaware of. Typically, a vanilla transformer is then applied to these embeddings. The transformer architectures' global attention allows the model to attend to faraway nodes if it chooses, bypassing inherent limitations of message-passing architectures. While the original graph transformer sets aside message-passing entirely, some modern architectures still incorporate neighborhood aggregation. In our study, we seek to implement the original graph transformer without message-passing. This will allow us to most directly compare the distinction between a transformer's global attention and an MPNN's local aggregation.

2.2 NC Metrics

To analyze the feature evolution in graph neural networks, we track several metrics related to the Neural Collapse phenomenon. The primary focus is on measuring both within-class variability and between-class separation of node features.

2.2.1 Core Covariance Matrices

The within-class covariance matrix $\Sigma_W(H)$ measures the average spread of features within each class:

$$\Sigma_W(H) := \frac{1}{Cn}\sum_{c=1}^{C}\sum_{i=1}^{n}(h_{c,i}-\bar{h}_c)(h_{c,i}-\bar{h}_c)^T \tag{1}$$

where $h_{c,i}$ represents the feature vector of the i-th node in class c, $\bar{h}_c$ is the mean feature vector for class c, C is the number of classes, and n is the number of nodes per class.

The between-class covariance matrix $\Sigma_B(H)$ captures the separation between different class centers:

$$\Sigma_B(H) := \frac{1}{C}\sum_{c=1}^{C}(\bar{h}_c-h_G)(\bar{h}_c-h_G)^T \tag{2}$$

where h_G represents the global mean feature vector across all classes.

2.2.2 Neural Collapse Metrics

To track the collapse of features during training, we employ two complementary metrics. The first metric, $NC_1(H)$, is based on the original Neural Collapse formulation:

$$NC_1(H) = \frac{1}{C}\mathrm{Tr}(\Sigma_W(H)\Sigma_B^{\dagger}(H)) \tag{3}$$

where $\Sigma_B^{\dagger}$ denotes the Moore-Penrose pseudo-inverse. The second metric, $\widetilde{NC}_1(H)$, provides an alternative measure that is more amenable to theoretical analysis:

$$\widetilde{NC}_1(H) = \frac{\mathrm{Tr}(\Sigma_W(H))}{\mathrm{Tr}(\Sigma_B(H))} \tag{4}$$

2.2.3 Variability Collapse in Neighborhood-Aggregated Features

For message-passing neural networks, we additionally track the variability collapse in the neighborhood-aggregated features matrix $H\hat{A}$. Similar to the core metrics, we measure both within-class and between-class variability using $\Sigma_W(H\hat{A})$ and $\Sigma_B(H\hat{A})$, which are computed using the neighborhood means $\bar{h}_c^{\mathcal{N}}$ and global neighborhood mean $\bar{h}_G^{\mathcal{N}}$.

The corresponding NC1 metrics for neighborhood-aggregated features are defined as:

$$\mathcal{NC}_1(H\hat{A}) = \frac{1}{C}\mathrm{Tr}(\Sigma_W(H\hat{A})\Sigma_B^{\dagger}(H\hat{A})) \tag{5}$$

$$\widetilde{\mathcal{NC}}_1(H\hat{A}) = \frac{\mathrm{Tr}(\Sigma_W(H\hat{A}))}{\mathrm{Tr}(\Sigma_B(H\hat{A}))} \tag{6}$$

To analyze the behavior of these metrics across multiple graph instances, we track the mean and variance of both the original metrics $\mathcal{NC}_1(H)$, $\widetilde{\mathcal{NC}}_1(H)$ and their neighborhood-aggregated counterparts $\mathcal{NC}_1(H\hat{A})$, $\widetilde{\mathcal{NC}}_1(H\hat{A})$ across all K graphs in our experiments.

3 Results

4 Discussion

5 Conclusion

References

Alon, Uri, and Eran Yahav. 2020. "On the bottleneck of graph neural networks and its practical implications." *arXiv preprint arXiv:2006.05205*

Cai, Chen, Truong Son Hy, Rose Yu, and Yusu Wang. 2023. "On the connection between mpnn and graph transformer." In *International Conference on Machine Learning*. PMLR

Choromanski, Krzysztof, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser et al. 2020. "Rethinking attention with performers." *arXiv preprint arXiv:2009.14794*

Dwivedi, Vijay Prakash, and Xavier Bresson. 2020. "A generalization of transformer networks to graphs." *arXiv preprint arXiv:2012.09699*

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

Kothapalli, Vignesh, Tom Tirer, and Joan Bruna. 2023. "A Neural Collapse Perspective on Feature Evolution in Graph Neural Networks." [Link]

Li, Qimai, Zhichao Han, and Xiao-Ming Wu. 2018. "Deeper insights into graph convolutional networks for semi-supervised learning." In *Proceedings of the AAAI conference on artificial intelligence*.

Müller, Luis, Mikhail Galkin, Christopher Morris, and Ladislav Rampášek. 2023. "Attending to graph transformers." *arXiv preprint arXiv:2302.04181*

Papyan, Vardan, X. Y. Han, and David L. Donoho. 2020. "Prevalence of neural collapse during the terminal phase of deep learning training." *Proceedings of the National Academy of Sciences* 117 (40), p. 24652–24663. [Link]

Rampášek, Ladislav, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. 2022. "Recipe for a general, powerful, scalable graph transformer." *Advances in Neural Information Processing Systems* 35: 14501–14515

Topping, Jake, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. 2021. "Understanding over-squashing and bottlenecks on graphs via curvature." *arXiv preprint arXiv:2111.14522*

Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. "Graph attention networks." *arXiv preprint arXiv:1710.10903*

Xu, Keyulu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. "How powerful are graph neural networks?" *arXiv preprint arXiv:1810.00826*

Appendices

For more details, refer to the Overleaf project at this link.

- **Caleb Stam:** Led the implementation and debugging of various graph transformer architectures, focusing on positional and structural encodings. Analyzed and modified the paper's codebase for Neural Collapse (NC) metrics and experimented with different model configurations and training loops.
- **Bill Wang:** Focused on adapting and improving Graph Transformer models, achieving high accuracy on Stochastic Block Model (SBM) datasets. Added attention mechanisms like performer and multihead to explore their impact on Neural Collapse. Contributed to the website and project report.
- **Eduardo Spiegel:** Implemented the GraphGPS network and achieved strong initial results. Adapted models to the reference paper's repository and addressed performance issues during training. Assisted with debugging and refining implementations, contributed to report writing, and planned further experiments on datasets like Cora.

A.1 Training Details

A.2 Additional Figures

A.3 Additional Tables