Neural Collapse in Quantized Neural Networks

Gabor Balazs Regely, ABX5LX

1 Introduction

During the terminal phase of training, when neural networks reach zero error, various architectures exhibit the Neural Collapse (NC) properties without explicitly defined structural constraints. NC represents a state at which the within-class variability of the final hidden layer outputs becomes infinitesimally small, class means form an equiangular tight frame and the last layer acts as a nearest-class center classifier [1, 2].

Quantization is a widely used technique to reduce the memory footprint and computational requirements of deep neural networks. However, quantization introduces noise and can potentially affect the training dynamics and final performance of neural networks.

Recent work has shown that, contrary to conventional deep learning wisdom, NC improves generalization, and studying it can lead to a better understanding of the inner workings of neural networks. [2] This project work aims to investigate the interplay between neural collapse and quantization. The project's source code is available at: `github.com/eRGiBi/QuantizedNeuralCollapse`.

2 Literature Review

2.1 Neural Collapse

Papyan et al. [1] introduced NC and described its four main properties across different architectures and datasets, focusing on the computer vision domain. NC is characterized as the co-occurrence of:

- NC1 - Within-class variability collapse: the within-class variation of the last hidden layer activations becomes negligible as they collapse to their class-means.
- NC2 - Class mean convergence to simplex ETF: vectors of class-means converge to an equiangular tight frame (ETF) after centering, maximizing pairwise angles and distances.

$$\cos(\theta_{c,c'}) = \frac{\langle \mu_c^t, \mu_{c'}^t \rangle}{|\mu_c^t| \cdot |\mu_{c'}^t|} \rightarrow \begin{cases} 1, & \text{if } c = c' \\ -\frac{1}{C-1}, & \text{if } c \neq c' \end{cases}$$

- NC3 - Self-dual alignment: columns of the last layer linear classifier matrix form a simplex ETF in their dual vector space and converge to the ETF with possible rescaling.

$$\left\| \frac{\mu_c^t}{||\mu_c^t||} - \frac{w_c^t}{||w_c^t||} \right\| \rightarrow 0$$

- NC4 - Nearest class center classification: the last-layer classifier acts with the nearest class mean decision rule on the penultimate layer features.

$$\hat{c}^t = \arg\min_{c'} ||h(x) - \mu_{c'}^t||$$

Kothapalli [2] gives a broad review on the principles behind NC and its counter-intuitive implications for generalization, transfer learning and adversarial robustness. They use the principles of Unconstrained Features Model (UFM) and Local Elasticity (LE) to model it. UFM is used to analyze the ideal values for perfect classification and their training dynamics, while LE is aimed to capture the separation of class features using stochastic differential equations and similarity kernels.

Linguistic Collapse Because Natural Language Processing (NLP) tasks are essentially classification problems, NC properties can also be identified in Language Models, as shown by Wu et al. [3] However, reaching NC in language models requires a new approach, as the conditions that give rise to it in computer vision differ in the NLP domain. Language models are typically undertrained, classes are imbalanced, the vocabulary size exceeds the embedding dimension, and contexts can be ambiguous to the next token prediction. The authors introduced modified metrics to measure NC in language models, show that scaling models makes the NC properties emerge, and that there is a correlation between NC and generalization.

Lu et al. [4] demonstrated that NC3 is minimized in debiasing methods. They add it as an auxiliary objective to be minimized in loss calculation, and conclude that this significantly improves fairness scores in language models without negatively affecting performance.

2.2 Quantization

Gholami et al. [5] provide a comprehensive survey of quantization methods for efficient neural network inference, covering the advantages and disadvantages of different methods.

Ashkboos et al. present EfQAT [6], an efficient framework for quantization-aware training that reduces the computational overhead of QAT while maintaining model accuracy.

3 Practical Results

3.1 Computer Vision

I ran experiments with multiple convolutional neural network architectures, including simple CNNs built from scratch, ResNet-18, MobileNetV3, but focued primarly on different ConvNeXt variants [7] and their customized versions with fewer parameters and removed stochasticity.

Using the NC metrics defined by Papyan et al. [1], I reproduced Neural Collapse accross several models trained on the CIFAR-10 and CIFAR-100 datasets. I was only able to reach Neural Collapse with carefully tuned learning rate scheduling and larger, overparameterized models.

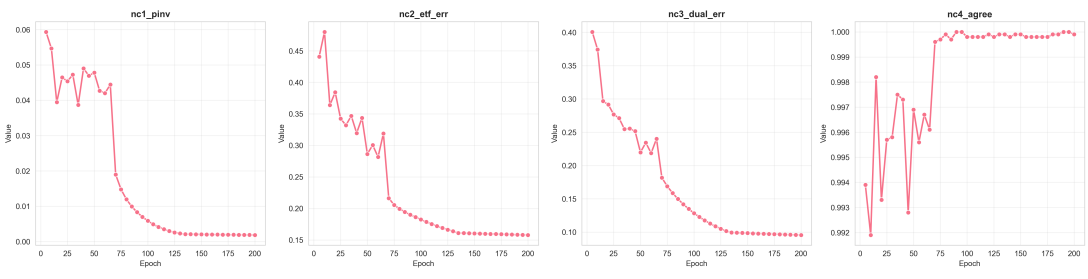

Figure 1: NC forming in a ResNet-18 model

I disabled all regularization techniques in the networks, such as stochastic depth, dropout, the auxiliary l_1 and l_2 norms of weights in the loss function, and all data augmentation.

3.2 Natural Language Processing

During my experimentation to reproduce NC in LMs, I expanded my codebase to cover multiple models and datasets. For quantization, I used the default 32-bit and half precision 16-bit floating point formats.

Empirical Neural Collapse metrics I measured NC properties with the empirical metric calculations provided by Wu et al. [3] from their GitHub repository.

NC1 Defined as:

$$NC_1 = \frac{\Sigma_W}{\Sigma_B + \epsilon},$$

where ϵ is a small constant,

$$\Sigma_W = \frac{1}{N \cdot D} \sum_{c=1}^{C} \sum_{i=1}^{N_c} \|h_{i,c} - \mu_c\|_2^2$$

is the within-class variability, and the between-class variability is:

$$\Sigma_B = \frac{1}{C \cdot D} \sum_{c=1}^{C} \|\mu_c - \mu_G\|_2^2$$

NC2 Let $\tilde{M} \in \mathbb{R}^{C \times D}$ be the centered means matrix with the c-th row being $(\mu_c - \mu_G)^\top$, the normalized Gram Matrix be

$$\bar{G} = \frac{\tilde{M}\tilde{M}^\top}{\|\tilde{M}\tilde{M}^\top\|_F + \epsilon},$$

and the ideal ETF Gram Matrix be

$$G^*_{i,j} = \begin{cases} \frac{\sqrt{C-1}}{C} & \text{if } i = j \\ \frac{-1}{C\sqrt{C-1}} & \text{if } i \neq j \end{cases}$$

So the ETF error using the Frobenius norm is:

$$NC_2 = \text{ETF}_{\text{err}} = \|\bar{G} - G^*\|_F$$

NC3 By $W \in \mathbb{R}^{C \times D}$ being the linear classifier weights and $\bar{W} = \frac{1}{C}\sum_j W_j$, the centered, normalized weights and means are

$$w_c = \frac{W_c - \bar{W}}{\|W_c - \bar{W}\|_2}, \quad z_c = \frac{\mu_c - \mu_G}{\|\mu_c - \mu_G\|_2},$$

and the cosine similarity for corresponding pairs being $S_c = w_c^\top z_c$, the dual error and uniformity metrics are:

$$\text{Error}_{\text{dual}} = \frac{1}{C}\sum_{c=1}^{C}(1 - S_c) \quad \text{and} \quad \text{Uniform}_{\text{dual}} = \text{Var}(S_1, S_2, \ldots, S_C)$$

NC4 Optionally implemented using an IndexFlatL2 index from FAISS [8], let the NCM prediction be $\tilde{y}_i$, and the learned linear prediction for sample i be $\hat{y}_i$:

$$\hat{y}_i = \underset{c}{\text{argmax}} \ (W_c^\top h_i + b_c)$$

By expanding the Euclidean distance $\|h_i - \mu_c\|_2^2$ as $\|h_i\|_2^2 + \|\mu_c\|_2^2 - 2h_i^\top \mu_c$:

$$\tilde{y}_i = \underset{c}{\text{argmin}} \ \|h_i - \mu_c\|_2^2,$$

the agreement proportion is:

$$\text{Agreement} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = \tilde{y}_i)$$

Datasets First, I ran my experiments on a dataset containing all of Shakespeare's works, modeled on the character level. Later, I reduced this to a subset of The Tragedy of Julius Caesar, which still proved to be too imbalanced in its distribution of target characters to form the required ETF. I adapted the TinyStories [9] dataset, which is designed to train and evaluate small, interpretable models, while still produce fluent language and even show reasoning abilities.

To lessen the compute requirements but still realize a model that is capable of reaching NC, I subsampled the tokens to be predicted and balanced them to be of equal amount. I used context lengths of 128 and 256 characters.

Modeling I modified a simple transformer implementation from the nanoGPT [10] repository. I set the parameter range to 1 to 10 million parameters with a hidden size of 4 times the embedding dimension, and generally used GELU as the activation function, Kaiming or normal weight initialization, cross-entropy for loss, AdamW as the optimizer with 0.9 momentum, 0.01 weight decay and (0.9, 0.95) betas.

I experimented with cosine, exponential and multistep learning rate scheduling.

I also started to investigate pretrained models and fine-tuning them, such as a tiny-llama model from HuggingFace. [11]

Weight decay In principle, weight decay should act as a regularizer, but it was shown that it does not prevent neural networks from fully memorizing training data, as I also confirmed experimentally. D'Angelo et al. [12] studied the loss stabilizer effect of weight decay in deep learning and how it differs from regularization.

Weight tying Weight tying reduces the number of parameters in a transformer by sharing the input token embeddings directly with the output classifier layer weights. [13] For NC, this acts as a severe structural constraint because the input and output geometries are forced to be identical. The classifier layer cannot freely collapse into an ETF and align properly without also constraining the input embeddings to conform to the same geometry.

4 Future Work

The goal of the project, reproducing Neural Collapse in language models and studying its interaction with quantization techniques, is still a work in progress. To achieve this, more compute, larger and more carefully tuned models, and a systematic comparison of NC metrics across quantization precisions and methods will be required.

References

[1] Vardan Papyan, X. Y. Han, and David L. Donoho. "Prevalence of neural collapse during the terminal phase of deep learning training". In: *Proceedings of the National Academy of Sciences* 117.40 (2020), 24652–24663. DOI: 10.1073/pnas.2015509117. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.2015509117.

[2] Vignesh Kothapalli. *Neural Collapse: A Review on Modelling Principles and Generalization*. 2023. arXiv: 2206.04041 [cs.LG]. URL: https://arxiv.org/abs/2206.04041.

[3] Robert Wu and Vardan Papyan. *Linguistic Collapse: Neural Collapse in (Large) Language Models*. 2024. arXiv: 2405.17767 [cs.LG]. URL: https://arxiv.org/abs/2405.17767.

[4] Jingxuan Xu et al. *Collapsed Language Models Promote Fairness*. 2025. arXiv: 2410.04472 [cs.CL]. URL: https://arxiv.org/abs/2410.04472.

[5] Amir Gholami et al. *A Survey of Quantization Methods for Efficient Neural Network Inference*. 2021. arXiv: 2103.13630 [cs.CV]. URL: https://arxiv.org/abs/2103.13630.

[6] Saleh Ashkboos et al. *EfQAT: An Efficient Framework for Quantization-Aware Training*. 2024. arXiv: 2411.11038 [cs.LG]. URL: https://arxiv.org/abs/2411.11038.

[7] Zhuang Liu et al. *A ConvNet for the 2020s*. 2022. arXiv: 2201.03545 [cs.CV]. URL: https://arxiv.org/abs/2201.03545.

[8] Matthijs Douze et al. "The Faiss library". In: (2024). arXiv: 2401.08281 [cs.LG].

[9] Ronen Eldan and Yuanzhi Li. *TinyStories: How Small Can Language Models Be and Still Speak Coherent English?* 2023. arXiv: 2305.07759 [cs.CL]. URL: https://arxiv.org/abs/2305.07759.

[10] Andrej Karpathy et al. *nanoGPT: The simplest, fastest repository for training/finetuning medium-sized GPTs*. https://github.com/karpathy/nanoGPT. 2025.

[11] Yaowei Zheng et al. "LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models". In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations). Association for Computational Linguistics, 2024. URL: https://arxiv.org/abs/2403.13372.

[12] Francesco D'Angelo et al. *Why Do We Need Weight Decay in Modern Deep Learning?* 2024. arXiv: 2310.04415 [cs.LG]. URL: https://arxiv.org/abs/2310.04415.

[13] Ofir Press and Lior Wolf. *Using the Output Embedding to Improve Language Models*. 2017. arXiv: 1608.05859 [cs.CL]. URL: https://arxiv.org/abs/1608.05859.