Direct Manifold Capacity Optimization

Satchel Grant
Department of Psychology
Stanford University
Stanford, CA 94305
grantsrb@stanford.edu

Chi-Ning Chou
Flatiron Institute
New York, NY 10010
cchou@flatironinstitute.org

Thomas Edward Yerxa
New York University
New York, NY 10010
tey214@nyu.edu

SueYeon Chung
Flatiron Institute
New York, NY 10010
EMAIL

Abstract

Manifold capacity is a tool for interpreting artificial and biological neural representations. Although the technique has shown utility in many analyses, an open question remains about whether the theory can also be used as a training objective for useful/robust neural representations. Previous work has made progress towards this goal in self-supervised learning settings by making assumptions about the shape of the manifolds. In this work, we use differentiable quadratic programming to maximize manifold capacity directly, without using simplifying assumptions. We show that our technique can match the overall performance of the pre-existing baselines with the ability to tune a hyperparameter to minimize the cumulative gradient steps or the total training samples. Our results show promise for exploring domains less suited to pre-existing simplifying assumptions, and our results add to the mounting evidence of manifold capacity as a powerful tool for characterizing neural representations.

1 Introduction

Manifold Capacity Theory (MCT) has developed over the past decade as a tool for characterizing neural representations. The technique asks the simple question of how easily a set of neural manifolds (or point clouds) are to be linearly separated by a randomly sampled hyperplane. The theory originated as a tool for biological neural networks [Froudarakis et al., 2020, Yao et al., 2023, Paraouty et al., 2023, Chou et al., 2024] and has since been recognized as a useful tool for artificial neural data as well [Cohen et al., 2020, Stephenson et al., 2021, Dapello et al., 2021, Kuoch et al., 2023].

Given the successes of MCT for neural data analysis, an immediate question arises as to whether it can also be used as a tool for representation learning. Can maximizing manifold capacity be a meaningful objective for useful neural representations? And if so, how can we practically incorporate manifold capacity into a training objective? Answering these questions would not only provide a new way to train neural networks, but would also add to the mounting evidence of the utility of MCT as a tool for neural analysis.

The prior work of Yerxa et al. [2023] made progress towards this goal in the domain of computer vision by making a simplifying assumption that the neural manifolds are elliptical. They called the technique Maximum Manifold Capacity Representations (MMCR). Their elliptical assumption allowed them to efficiently pre-train convolutional neural networks in a contrastive, self-supervised learning (SSL) setting. Although the elliptical assumption worked well for creating useful features

Preprint. Under review.

in vision, it is less clear that this assumption is suited for less smooth domains such as language modeling or higher level cognitive processing. Can we move beyond the simplifying assumptions used in MMCR to directly optimize for manifold capacity?

In this work, we use the latest formulation of MCT [Chou et al., 2024] combined with relatively recent developments in differentiable quadratic programming [Agrawal et al., 2019] to demonstrate that it is possible to perform direct manifold capacity optimization (DMCO). We show that using DMCO, we can match the performance of MMCR in representation learning on Cifar-10 [Krizhevsky et al.] with the ability to minimize the cumulative number of gradient steps or overall training samples depending on hyperparameter choices. Although these results are from toy settings, they show promise for the future of DMCO in SSL, and they serve to further validate the legitimacy of MCT as a tool for characterizing neural representations.

2 Related Work

Many SSL methods for learning vision and multimodal representations operate by encouraging related data points ("positive pairs“) to be nearby in the representation space while employing some mechanism to prevent trivial collapsed solutions. Instance and dimension contrastive methods [Balestriero and LeCun, 2022] require diversity in the representation over unrelated data points and feature decorrelation respectively. While these examples are most relevant to the current work many other approaches exist, see [Balestriero et al., 2023] for a comprehensive review.

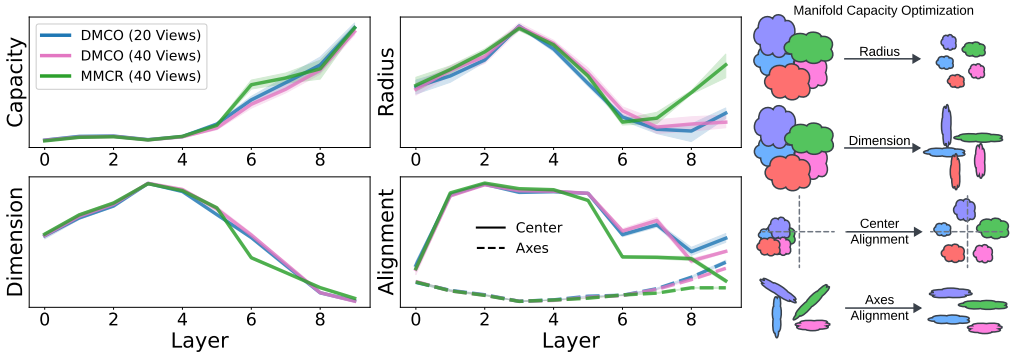

Figure 1: A comparative analysis of manifold geometry at different layers of the model trained using DMCO and MMCR.

3 Methods

3.1 SSL in Computer Vision

All of our experiments are performed in a self-supervised learning (SSL) setting on images from Cifar-10 [Krizhevsky et al.]. We follow the same procedure as that of Yerxa et al. [2023], where we sample a batch of b images and create k random augmentations of each image. This results in a matrix of inputs which are then processed by an ANN, $f_\theta(x)$, that is parameterized by θ. The outputs of $f_\theta(x)$ are projected onto the unit sphere which results in a matrix of outputs $Z \in R^{m \times d}$ where $m = bk$ and d is the dimensionality of the representations. We treat the representations, $z_{i,j} \in \{z_{i,1}, z_{i,2}, ..., z_{i,k}\}$, resulting from a single input image, x_i, as a single manifold. In general, the goal of the SSL method is to make representations within a manifold more similar, and make representations between manifolds more dissimilar.

3.2 Direct Manifold Capacity Optimization (DMCO)

Manifold capacity is a metric that can be used to characterize the coding efficiency of neural representations. The original definition of manifold capacity is the critical number of manifolds per neuron that are linearly separable. This corresponds to how efficiently the neural state space packs

the manifolds. Recently, it was found that there are variational forms for manifold capacity [Chou et al., 2024]. In this work, we focus on the version related to decoding robustness. Intuitively, this form measures the quality of a randomly sampled hyperplane T for (linearly) separating the neural manifolds. Concretely, we can start with the primal problem:

$$\min_{v} \|v - T\|^2 \tag{1}$$

$$\text{s.t. } y_i \cdot \langle z_{i,j}, v \rangle \leq 0 \; \forall \; i, j \tag{2}$$

Where $T \in \mathbb{R}^N$ defines a sampled hyperplane from the normal distribution, $z \in \mathbb{R}^N$ is a single representation, $y_i \in \{-1, 1\}$ is a randomly sampled label assigned to manifold i, and $v \in \mathbb{R}^N$ is the vector to be minimized. Intuitively, v is the closest decision boundary to T that successfully separates the manifolds. Equation 1 can be solved using Langrange multipliers and quadratic programming.

Once the Lagrange multipliers are found, manifold capacity, denoted as α, can be calculated using the following formulation [Chou et al., 2024]:

$$\alpha^{-1} = \mathop{\mathbb{E}}_{\substack{T \sim N(0, I_d) \\ y \sim \{-1,1\}}} \left[\left\| \sum_{i=1}^{b} \sum_{j=1}^{k} y_i \lambda_{i,j}(Z, y, T) f_\theta(x_{i,j}) \right\|_2^2 \right] \tag{3}$$

Where i indexes the corresponding manifold, j indexes a point within the manifold i, y is a vector of all sampled y_i, $\lambda_{i,j}$ is a Lagrange multiplier that is dependent on T, y, and all data points Z, and $f_\theta(x_{i,j})$ is the neural network representation, $z_{i,j}$, from a data sample $x_{i,j}$.

We use Differentiable Convex Optimization Layers [Agrawal et al., 2019] to calculate each λ in a differentiable way. This allows us to use α^{-1} as a loss function and directly maximize α with respect to the weights θ using gradient descent.

Similar to results found in MMCR, we found empirically that directly minimizing α^{-1} yielded poor results, likely due to a dimensionality collapse of the features [Yerxa et al., 2023, Jing et al., 2022]. To prevent this collapse, we introduce two different capacity terms. The first is the local manifold capacity loss, α_ℓ^{-1}, which is calculated and minimized as described in Equation 3. The other is the global manifold capacity reward, α_g^{-1}, which we seek to maximize. To calculate α_g^{-1}, we start with Equation 3 but we replace the $f_\theta(x_{i,j})$ terms with the centroids, $c_{\theta,i}$, of each manifold and then use all the centroids as points within a single manifold.

$$c_{\theta,i} = \frac{1}{m} \sum_{j=1}^{m} f_\theta(x_{i,j}) \tag{4}$$

$$\alpha_g^{-1} = \mathop{\mathbb{E}}_{T \sim N(0, I_d)} \left[\left\| \sum_{i=1}^{b} \lambda_i(C, T) c_{\theta,i} \right\|_2^2 \right] \tag{5}$$

Where $C \in R^{b \times d}$ is a matrix of all centroids included in Equation 5.

We then combine the global and local objectives using a weighting term, ϵ, for a single loss function that can be minimized using stochastic gradient descent.

$$\mathcal{L} = \epsilon \alpha_\ell^{-1} - (1 - \epsilon) \alpha_g^{-1} \tag{6}$$

See Appendix A.2 for hyperparameter settings.

4 Results

We turn our attention to Figure 2 where we compare the *k*-Nearest Neighbor (kNN) performance of our Direct Manifold Capacity Optimization (DMCO) to Maximum Manifold Capacity Representations (MMCR) on ResNet-18 architectures trained on CIFAR-10. We look at performance over the course of training both in terms of model updates and cumulative data samples during training. We see that DMCO matches the performance of MMCR with the ability to minimize the cumulative number of image augmentations used for training or the cumulative number of gradient steps. We emphasize that the elliptical manifold assumption used in MMCR is likely well suited to manifolds in early

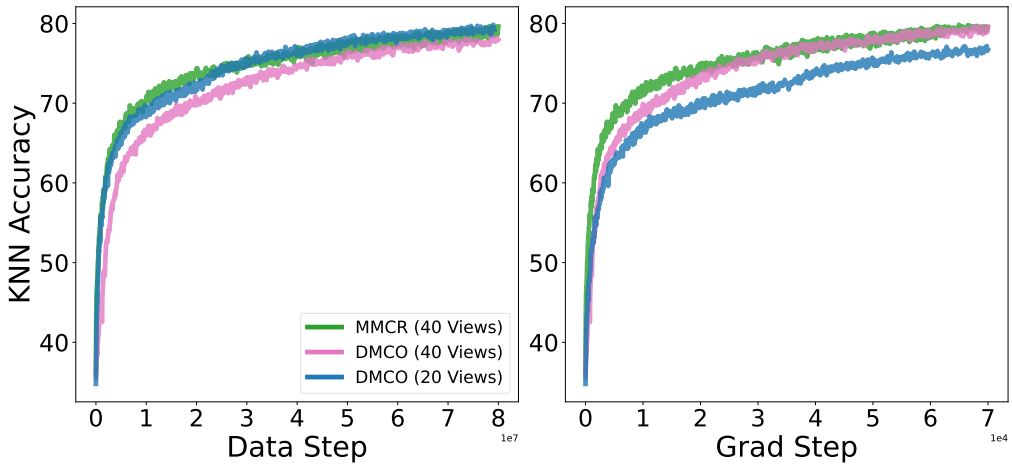

Figure 2: A performance comparison of different training methods showing the kNN classification accuracy on held out data using the learned model features. We can see that DMCO matches the performance of MMCR with the ability to minimize data consumption or number of model updates. Data steps are measured as total image augmentations. Gradient steps are measured as number of model updates.

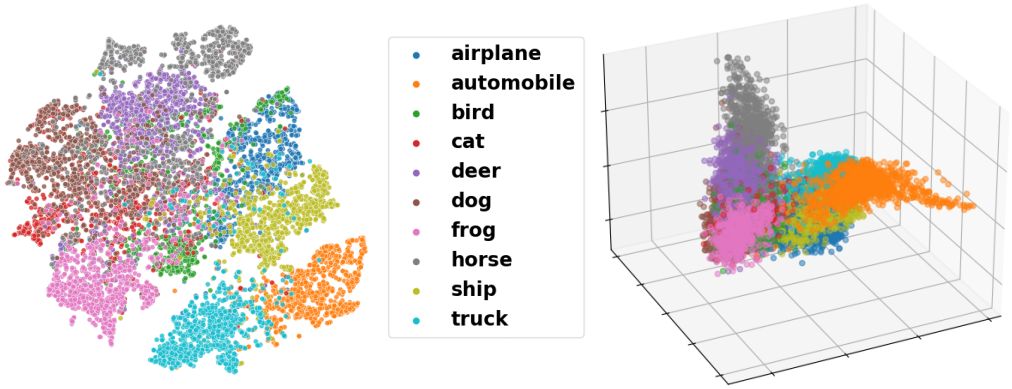

Figure 3: Low dimensional visualizations of the learned model features trained using DMCO with 20 views. *Left:* 2 dimensional t-SNE. *Right:* 3 dimensional PCA.

visual processing. As such, it is reasonable to expect that MMCR will learn faster in visual domains than DMCO and that the two methods will have similar final performance.

A qualitative visualization of the DMCO (20 View) learned features can be found in Figure 3. We can see that reasonable clustering emerges from the training. Automobiles, ships, and trucks are clustered near one another; cats and dogs are clustered together with limited separation; deer and horses are nearby one another; birds and airplanes have a large degree of overlap; and ships and airplanes are clustered near one another.

Lastly, we examine the geometric analysis in Figure 1. This analysis was performed using manifolds formed over class labels from the Cifar-10 test set. In general, manifold capacity is maximized when radius, dimension, and center alignment are small and when axes alignment is high. We can see that MMCR and DMCO produce relatively similar geometries at the final layers except that DMCO produces a more optimal radius and axes alignment whereas MMCR produces more optimal center alignment.

5 Conclusion

This work has demonstrated how to directly optimize manifold capacity in vision models. We have matched the performance of MMCR while maintaining data consumption and gradient steps. We have shown that the two techniques produce relatively similar manifold capacity with slightly different geometric measures. This work serves as a proof of principle that manifold capacity can be used as an optimization objective for useful visual features. This serves as a new method for the field of SSL, and it serves to further validate manifold geometric analyses as relevant methods for neural analysis.

Acknowledgments and Disclosure of Funding

We wish to acknowledge the Simon's foundation for research funding and the PDP Lab at Stanford for providing extra computational resources.

References

Emmanouil Froudarakis, Uri Cohen, Maria Diamantaki, Edgar Y Walker, Jacob Reimer, Philipp Berens, Haim Sompolinsky, and Andreas S Tolias. Object manifold geometry across the mouse cortical visual hierarchy. *BioRxiv*, 2020.

Justin D Yao, Klavdia O Zemlianova, David L Hocker, Cristina Savin, Christine M Constantinople, SueYeon Chung, and Dan H Sanes. Transformation of acoustic information to sensory decision variables in the parietal cortex. *Proceedings of the National Academy of Sciences*, 120(2): e2212120120, 2023.

Nihaad Paraouty, Justin D Yao, Léo Varnet, Chi-Ning Chou, SueYeon Chung, and Dan H Sanes. Sensory cortex plasticity supports auditory social learning. *Nature communications*, 14(1):5828, 2023.

Chi-Ning Chou, Luke Arend, Albert J. Wakhloo, Royoung Kim, Will Slatton, and SueYeon Chung. Neural manifold capacity captures representation geometry, correlations, and task-efficiency across species and behaviors. *bioRxiv*, 2024. doi: 10.1101/2024.02.26.582157. URL `https://www.biorxiv.org/content/early/2024/02/28/2024.02.26.582157`.

Uri Cohen, SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature communications*, 2020.

Cory Stephenson, Suchismita Padhy, Abhinav Ganesh, Yue Hui, Hanlin Tang, and SueYeon Chung. On the geometry of generalization and memorization in deep neural networks. *arXiv preprint arXiv:2105.14602*, 2021.

Joel Dapello, Jenelle Feather, Hang Le, Tiago Marques, David Cox, Josh McDermott, James J DiCarlo, and SueYeon Chung. Neural population geometry reveals the role of stochasticity in robust perception. *Advances in Neural Information Processing Systems*, 34:15595–15607, 2021.

Michael Kuoch, Chi-Ning Chou, Nikhil Parthasarathy, Joel Dapello, James J DiCarlo, Haim Sompolinsky, and SueYeon Chung. Probing biological and artificial neural networks with task-dependent neural manifolds. In *Conference on Parsimony and Learning (Proceedings Track)*, 2023.

Thomas Yerxa, Yilun Kuang, Eero Simoncelli, and SueYeon Chung. Learning efficient coding of natural images with maximum manifold capacity representations, 2023. URL `https://arxiv.org/abs/2303.03307`.

A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and Z. Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, 2019.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

Randall Balestriero and Yann LeCun. Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods. *Advances in Neural Information Processing Systems*, 35:26671–26685, 2022.

Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, Avi Schwarzschild, Andrew Gordon Wilson, Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A cookbook of self-supervised learning, 2023. URL https://arxiv.org/abs/2304.12210.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning, 2022. URL https://arxiv.org/abs/2110.09348.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703, 2019. URL http://arxiv.org/abs/1912.01703.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. URL https://arxiv.org/abs/1502.03167.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html.

Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901. doi: 10.1080/14786440109462720.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

A Appendix / supplemental material

A.1 Model Architecture

The code for all model trainings was implemented using PyTorch [Paszke et al., 2019]. We import the ResNet-18 [He et al., 2015] architecture from the torchvision package without pre-trained weights and remove the max-pooling layers for generation of the visual features. We then feed the flattened visual features into a 2 layer multi-layer perceptron with a first layer of 1024 units, and a second layer of 32, with a batch normalization [Ioffe and Szegedy, 2015] followed by a ReLU nonlinearity at the hidden layer.

A.2 Additional Settings

In the DMCO framework, we have the ability to choose the number of manifolds to include in a single forward pass of Equation 3 for both the local and global objectives. It seemed to be helpful to have a number of image augmentations, k, that was greater than the number of manifolds for the local objective. We believe this is due to the randomly sampled binary labeling within Equation 3 that can lead to inefficient manifold credit assignment if the number of manifolds is large. For the results in this paper, we used 10 manifolds with the reported number of augmentations (or views). The global objective, however, appeared to benefit from larger numbers of samples. In practice, we summed 3 independent calculations of α_ℓ^{-1} and then combined each of their respective 10 centroids for the global calculation of α_g^{-1}. We used an ϵ of 0.99 to weight the summed local objective from Equation 3.

We also have the choice of the number of samples of T to use to approximate the expectation in Equation 3. We used 100 samples, additionally sampling the labels, y, uniformly for each sampled T. We found that increasing the number of samples improved results, although it comes with a cost of computational efficiency. We used CVXpy Layers with the PyTorch backend with default settings

[Agrawal et al., 2019] to differentiate through Equation 3. For all trainings, we used a learning rate of 0.001. We used Nvidia A40 GPUs for accelerated computation.

The t-SNE [van der Maaten and Hinton, 2008] and PCA [F.R.S., 1901] were performed using the SciKit-Learn package [Pedregosa et al., 2011].