GEODESIC MODE CONNECTIVITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Mode connectivity is a phenomenon where trained models are connected by a path of low loss. We reframe this in the context of Information Geometry, where neural networks are studied as spaces of parameterized distributions with *curved geometry*. We hypothesize that shortest paths in these spaces, known as *geodesics*, correspond to mode-connecting paths in the loss landscape. We propose an algorithm to approximate geodesics and demonstrate that they achieve mode connectivity.

1 INTRODUCTION

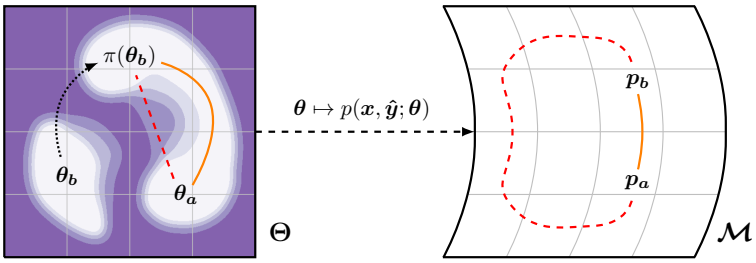

Figure 1: **Geodesics are shortest paths in the space of parameterized distributions $\mathcal{M}$.** For narrow architectures linear interpolation (dashed) fails to achieve mode connectivity, passing through a region of high loss, despite using a permutation π to 'shift' $\boldsymbol{\theta_b}$ closer to $\boldsymbol{\theta_a}$ (Ainsworth et al., 2022). If we instead follow the geodesic (shortest) path (solid) in the curved distribution space, this *does* achieve mode connectivity, appearing as a curved path in the loss landscape.

Garipov et al. (2018) demonstrated the existence of **mode connectivity**, wherein stochastic gradient descent (SGD) solutions are connected by low-loss paths in the loss landscape. This result challenged the prevailing perspective of isolated minima in favour of large connected valleys. More recently, Ainsworth et al. (2022) achieved **linear mode connectivity** (LMC), finding *linear* paths between two minima with no increase in loss. They achieved this by exploiting the natural permutation symmetries of neural network layers (Entezari et al., 2022) to 'shift' one of the models into the same loss basin as the other, enabling LMC (see Figure 1, LHS).

Information Geometry interprets neural networks as parameterized distributions $p(\boldsymbol{x}, \hat{\boldsymbol{y}}; \boldsymbol{\theta})$, studying them using tools from differential geometry (Amari, 2012). In this setting, the distribution space $\mathcal{M}$ is endowed with a *metric* which determines lengths and gives the space a curvature, much like Einstein's theory of general relativity defines a metric describing curved spacetime. One can then define **geodesics** as the *paths of shortest length* between two points in this space.

Our work demonstrates a connection between mode connectivity and geodesics in the curved distribution space. In particular, **we hypothesize that all geodesics between SGD solutions are mode-connecting paths**. Our specific contributions include:

1. A novel algorithm for approximating geodesic paths through the loss landscape.
2. Using our algorithm to find mode-connecting paths between narrow ResNets trained on CIFAR-10, where existing methods require wider architectures.

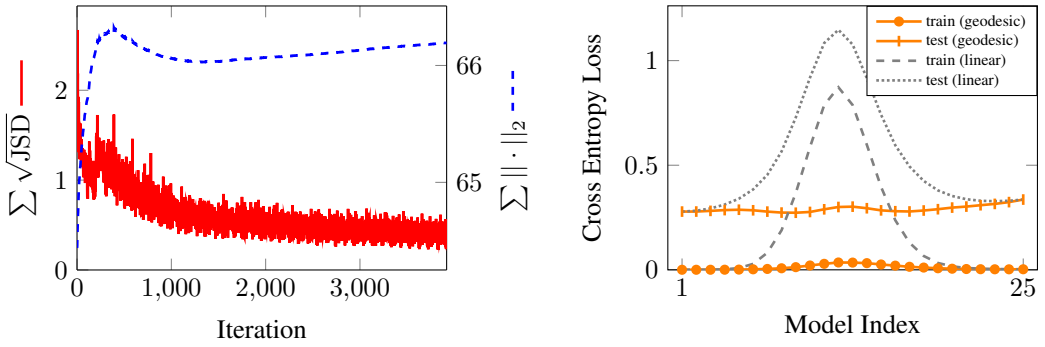

Figure 2: **Geodesic optimization achieves mode connectivity for ResNet20 on CIFAR-10. Left: Geodesic optimization minimizes discretized length functional**. Our algorithm finds a curved path in parameter space; although this curvature increases the Euclidean length (dashed) compared to the linear path, it decreases the distribution space length (solid). **Right: Geodesic optimization identifies low-loss paths where linear mode connectivity has failed**. Linear interpolation between $\boldsymbol{\theta_1}$ and $\boldsymbol{\theta_N}$ shows a large increase in loss on both train and test datasets. After geodesic optimization, all points on the path have low loss, which we refer to as **geodesic mode connectivity**.

2 Geodesic Optimization

Given two trained neural networks, $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_b}$, we seek to approximate the geodesic path γ between the distributions $\boldsymbol{p_a} = \boldsymbol{p}(\boldsymbol{x}, \hat{\boldsymbol{y}}; \boldsymbol{\theta_a})$ and $\boldsymbol{p_b} = \boldsymbol{p}(\boldsymbol{x}, \hat{\boldsymbol{y}}; \boldsymbol{\theta_b})$. This path minimizes the length functional defined in Equation 1, where g_{ij} is the Fisher-Rao metric (see Appendix A.1). This functional is equivalent to the integral of the infinitesimal Jensen-Shannon Divergence (JSD) (Crooks, 2007). Our algorithm uses a discrete version of Equation 1. We first initialize a sequence of N models $\{\boldsymbol{\theta_a} = \boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \ldots, \boldsymbol{\theta_N} = \boldsymbol{\theta_b}\}$ along a linear path. Keeping the endpoints fixed, we optimize parameters $\{\boldsymbol{\theta_i}\}$ to minimize the loss defined in Equation 2. Note two differences with Equation 1: (i) We approximate the integral as a discrete sum, measuring the JSD between distributions defined by each $\boldsymbol{\theta_i}$, similar to (Carter et al., 2008) (ii) We do not take the square root of JSD; this is equivalent to minimizing the *energy* functional which gives the *unique* geodesic of constant velocity.

$$L(\gamma) = \int_t \sqrt{\frac{d\gamma^i}{dt} g_{ij} \frac{d\gamma^j}{dt}} dt = \sqrt{8} \int_\gamma \sqrt{d\text{JSD}} \tag{1}$$

$$\mathcal{L}(\{\boldsymbol{\theta_i}\}_{i=1}^N) = \sum_{i=1}^{N-1} \text{JSD}(\boldsymbol{p_i} || \boldsymbol{p_{i+1}}) \tag{2}$$

3 Experiments

We conduct experiments with a ResNet20 architecture (He et al., 2016) on the CIFAR-10 dataset (Krizhevsky, 2009). We independently train two networks, $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_b}$, and use the weight matching algorithm of Ainsworth et al. (2022) to permute $\boldsymbol{\theta_b}$ closer to $\boldsymbol{\theta_a}$. Even after permuting, the linear path between the two models still shows a large increase in loss, as seen in Figure 2 (RHS). We then use the algorithm defined in Section 2 with $N = 25$. Note this only requires training images, no labels or test data are employed. See Appendix A.2 for further details. The results in Figure 2 demonstrate that our algorithm achieves mode connectivity, where LMC has failed.

4 Conclusion

Geodesic optimization succeeds in identifying paths of low-loss between narrow ResNets, where linear mode connectivity is not present. Future work could compare geodesic paths with those found by the method of Garipov et al. (2018), or theoretically investigate these geodesic paths.

URM STATEMENT

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2023 Tiny Papers Track.

REFERENCES

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git Re-Basin: Merging Models modulo Permutation Symmetries, December 2022. URL http://arxiv.org/abs/2209.04836. arXiv:2209.04836 [cs].

Shun-ichi Amari. *Differential-Geometrical Methods in Statistics*. Springer Science & Business Media, December 2012. ISBN 978-1-4612-5056-2. Google-Books-ID: XiDnBwAAQBAJ.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. 2016. doi: 10.48550/ARXIV.1607.06450. URL https://arxiv.org/abs/1607.06450.

Kevin Carter, Raviv Raich, and Alfred Hero. Learning on statistical manifolds for clustering and visualization. December 2008.

Gavin E. Crooks. Measuring Thermodynamic Length. *Physical Review Letters*, 99(10):100602, September 2007. doi: 10.1103/PhysRevLett.99.100602. URL https://link.aps.org/doi/10.1103/PhysRevLett.99.100602. Publisher: American Physical Society.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The Role of Permutation Invariance in Linear Mode Connectivity of Neural Networks, July 2022. URL http://arxiv.org/abs/2110.06296. arXiv:2110.06296 [cs].

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry Vetrov, and Andrew Gordon Wilson. Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs, October 2018. URL http://arxiv.org/abs/1802.10026. arXiv:1802.10026 [cs, stat].

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016. doi: 10.1109/CVPR.2016.90. ISSN: 1063-6919.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015. doi: 10.48550/ARXIV.1502.03167. URL https://arxiv.org/abs/1502.03167.

A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009. URL https://www.semanticscholar.org/paper/Learning-Multiple-Layers-of-Features-from-Tiny-Krizhevsky/5d90f06bb70a0a3dced62413346235c02b1aa086.

A APPENDIX

A.1 NEURAL NETWORK INFORMATION GEOMETRY

In the context of supervised learning for classification tasks, a neural network with parameters $\boldsymbol{\theta}$ can be interpreted as a conditional distribution $p(\hat{\boldsymbol{y}}|\boldsymbol{x};\boldsymbol{\theta})$, which maps an input $\boldsymbol{x}$ to a probability distribution over a set of class labels $\hat{\boldsymbol{y}}$. Coupled with a distribution over the inputs $p(\boldsymbol{x})$, we get a parameterized joint probability distribution $p(\boldsymbol{x}, \hat{\boldsymbol{y}}; \boldsymbol{\theta}) = p(\boldsymbol{x})p(\hat{\boldsymbol{y}}|\boldsymbol{x};\boldsymbol{\theta})$. In practice, we simply take $p(\boldsymbol{x})$ to be the empirical distribution of our input data. These are the joint probabilities we work with in the main text.

This space of joint distributions, under certain assumptions, is a Riemannian manifold equipped with a Riemannian metric, known as the **Fisher-Rao metric** g. This is also known, in broader contexts, as the Fisher information matrix g: given a parameter $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_L)$, where L is the total number of neural network parameters, the (i, j)-coordinate of the Fisher-Rao metric is given by Equation 3.

$$g_{ij}(\boldsymbol{\theta}) = \mathbb{E}\left[\frac{\partial \log p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})}{\partial \theta_i} \frac{\partial \log p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})}{\partial \theta_j}\right], \tag{3}$$

where the expectation is taken with respect to the joint distribution $p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = p(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta})$. For more details on Information Geometry and its applications to neural networks, see Amari (2012).

A.2 Implementation Details

A.2.1 Geodesic Algorithm

We begin by training two separate models $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_b}$ with different random seeds using SGD. We then use the weight matching algorithm of Ainsworth et al. (2022) to permute $\boldsymbol{\theta_b}$ to be close in the loss landscape to $\boldsymbol{\theta_a}$. Note this does not change the outputs of the network due to the permutation symmetries of common neural network layers. We found that applying this permutation meaningfully improves the performance of the geodesic algorithm. This is likely because different points in the parameter space $\boldsymbol{\Theta}$ can correspond to the *same* distribution $\boldsymbol{p}$, leading to undesirable behavior. Applying the permutation allows the path to travel through a smaller area of the loss landscape where the output distributions are closer to being uniquely identified.

As stated in Section 2, we linearly interpolate between $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_b}$ to obtain N models $\{\boldsymbol{\theta}_i\}_{i=1}^N$, where $N = 25$ for the experiments in this paper. We optimize the parameters $\boldsymbol{\theta}_i$ using SGD with a learning rate of 0.1 and a batch size of 256, with the loss as given by Equation 1. Note that JSD is given by an expectation over the joint distribution parameterized by $\boldsymbol{\theta}$, but we do not have access to the entire input space distribution. Thus, we approximate this with the empirical input distribution, $\hat{p}(\boldsymbol{x})$, and $p(\hat{\boldsymbol{y}}|\boldsymbol{x}; \boldsymbol{\theta})$ is given by the softmax of the logits outputted by the last ResNet layer. We do not use any observed class labels $\boldsymbol{y}$ at any point in this algorithm and are only minimizing path action via JSD, yet we observe lower loss barriers observed along the resulting approximate geodesic when compared lines in parameter space. This contrasts with previous work done by Garipov et al. (2018) where simple paths of low loss are found explicitly taking into account class labels.

A.3 A note on architecture

One key difference in the architectures used in Ainsworth et al. (2022) and this work from standard ResNets was the use of LayerNorm (Ba et al., 2016) instead of BatchNorm (Ioffe & Szegedy, 2015). This is due to the fact that BatchNorm layers are not invariant to permutations, and require retraining the BatchNorm layers with permuted statistics, while LayerNorm is invariant to permutations. The use of LayerNorm was observed to meaningfully change the optimal hyperparameters and decreased the test accuracy of the models by a few percentage points compared to common hyperparameter settings found in the literature for BatchNorm versions of the same model.