LINEAR MODE CONNECTIVITY OF DEEP NEURAL NETWORKS VIA PERMUTATION INVARIANCE AND RENORMALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper we empirically investigate the conjecture from Entezari et al. (2021) which states that if permutation invariance is taken into account, then there should be no loss barrier to the linear interpolation between SGD solutions. We conduct our investigation using standard computer vision architectures trained on CIFAR-10 and ImageNet. First, we observe a general phenomenon in which interpolated deep networks suffer a collapse in the variance of their activations. We demonstrate that an appropriate rescaling of the pre-activations of the interpolated networks ameliorates this problem and significantly reduces the barrier. Second, by combining this with an algorithm for finding permutations based on maximizing correlations between the activations of matched neurons, we are able to reduce the interpolation barrier for a standard ResNet18 trained on CIFAR-10 to 1.5% absolute test error. We explore the interaction between our method and the choice of normalization layer, and demonstrate its robustness across a variety of architectures and training sets.

1 INTRODUCTION

Training a neural network corresponds to optimizing a highly non-linear function by navigating a complex loss landscape with numerous minima, symmetries and saddles (Zhang et al., 2017; Keskar et al., 2017; Draxler et al., 2018; Şimşek et al., 2021). Overparameterization is one of the reasons behind the abundance of minima leading to different functions behave similarly on the training data but different on other inputs (Neyshabur et al., 2017; Nguyen et al., 2018; Li et al., 2018; Liu et al., 2020). Another reason are permutation and scaling invariances which lead to functionally identical minima that differ in the weight space (Brea et al., 2019; Entezari et al., 2021). Due to the relationship of loss landscape with generalization and optimization, a large body of recent work (Li et al., 2017; Mei et al., 2018; Geiger et al., 2019; Nguyen et al., 2018; Fort et al., 2019; Şimşek et al., 2021; Juneja et al., 2022) study the properties of the loss landscape of DNNs with the goal to navigate the optimizer to a good region with desired properties, *e.g.*, with respect to flatness around the SGD solution (Baldassi et al., 2020; Pittorino et al., 2020).

Early works conjectured the existence of a *non-linear* path of non-increasing loss between solutions found by SGD (Freeman & Bruna, 2016; Draxler et al., 2018) and empirically showed how to find it (Garipov et al., 2018; Pittorino et al., 2022). Recently, Entezari et al. (2021) conjectured the existence of a *linear* path between SGD solutions if permutation invariance of neural networks function space is taken into account. The conjecture is important from both theoretical and empirical perspectives. Theoretically, it leads to a drastic simplification of the loss landscape, reducing the complexity obstacle for theoretically analyzing deep neural networks. Empirically, linear interpolation between network weights has become an important tool, having recently been used to set state of the art accuracy on ImageNet (Wortsman et al., 2022a), improve the robust accuracy of finetuned models (Wortsman et al., 2022b; Ilharco et al., 2022), and build effective weight-space model ensembles (Izmailov et al., 2019; Frankle et al., 2020; Guo et al., 2022). New insights about the behavior of interpolated networks may lead to further empirical progress in utilizing linear interpolation to achieve better results.

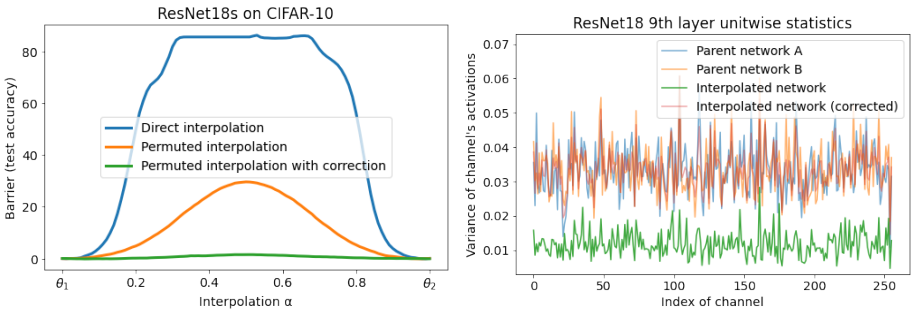


Figure 1: **Neuron-aligned deep networks are nearly mode-connected after correcting the statistics of interpolated hidden units. (left):** We plot the increase in test error when moving along the linear path between two SGD solutions for the ResNet18 architecture trained on CIFAR-10. We compare direct interpolation (blue line) and interpolation after neuron-aligning the two networks (Li et al., 2015; Ainsworth et al., 2022). We find that neuron-alignment is not able to remove the barrier for a standard ResNet18. Investigating this, **(right):** we find that interpolated networks suffer from a phenomenon of *variance collapse* in which the activations of their hidden units have reduced variance (green line) relative to those of trained networks (blue and orange lines). We observe that by rescaling the activations of the interpolated network such that their statistics roughly match those of the parent networks (red line), we can correct the variance collapse, leading to a significant increase in the performance of the interpolated networks (green line of left figure), and the reduction of the barrier from 29.6% to 1.5%.

Entezari et al. (2021)'s conjecture has been shown to hold true in special cases both theoretically and empirically. In particular, the loss barrier approaches zero for sufficiently wide overparameterized shallow networks. Şimşek et al. (2021) show that adding one extra neuron to each MLP layer is sufficient to connect previously discrete minima into a single manifold. Singh & Jaggi (2020) explore linear mode connectivity with soft-alignment between ResNet18s, achieving a barrier of 14%. Pittorino et al. (2022) normalize networks by projecting them onto a hyper-toroidal manifold, which they find reduces the barrier for VGG16 on CIFAR10 to 20%.

Earlier works are unable to establish low-barrier linear mode connectivity between permuted solutions for standard deep networks. In this work we investigate and resolve the cause of the typically poor performance of deep networks which have been interpolated from independently-trained weights. In summary, the contributions of this paper are as follows:

- We empirically investigate the behavior of interpolated deep networks in order to determine why permutation invariance alone is insufficient to allow low-barrier linear interpolation between SGD solutions to standard deep architectures. We find that interpolated networks suffer from a phenomenon in which their hidden units have significantly smaller activation variance relative to the corresponding units of the trained networks. We refer to this phenomenon as *variance collapse* (Section 3).
- We propose a method which corrects for variance collapse by rescaling hidden units in the interpolated network such that their statistics match those of the original trained networks (Section 4).
- We show that after applying this correction, it is possible to linearly interpolate between permuted SGD solutions to standard deep architectures with a low barrier. For example, we demonstrate (Figure 1) that when statistically corrected, an interpolated ResNet18 reaches 94.1% test accuracy on CIFAR-10, only 1.5% lower than the original trained networks. To our knowledge, this is a $10\times$ improvement relative to previous approaches in terms of the size of the barrier to linear interpolation for a standard ResNet18. We further demonstrate the utility of our method to a variety of other architectures and datasets (Section 5).

1.1 Related Work

Neuron alignment is a technique of finding a match between neurons in the same layer of two neural networks, so that these networks are linear mode connected. Early works (Li et al., 2015) investigated the extent to which representations learned by independently trained networks converge to a similar set of features. They use neuron alignment based on linear correlation and estimated mutual information metrics. Since brute force is ineffective when searching for winning permutations due

to a factorial time complexity, several recent works proposed heuristics for neuron alignment. Entezari et al. (2021) implements neuron alignment based on Simulated Annealing (Zhan et al., 2016) in order to remove the barrier between independently trained models. The method is successful only in a very limited number of cases. He et al. (2018) proposed a neuron alignment method in the context of multi-task model compression. The algorithm leverages layer-wise Hessian approximation to match neuron by computing a similarity measure based on their functional difference. Singh & Jaggi (2020) propose a model fusion algorithm which leverages optimal transport to perform neuron alignment in the Wasserstein space extensively studied by Agueh & Carlier (2011). The method allows building high-quality ensembles of models trained on heterogeneous datasets.

Tatro et al. (2020) point out that special care must be taken for networks that contain batch normalization layers when it comes to neuron alignment. They match neurons by optimizing a curve in the weight space between two models by aligning correlations of post-activations. The running statistics are normalized by training the model for one epoch, while freezing all learnable parameters of the model. However, the crucial role of statistics reset in the context of linear interpolation of the models is not investigated. A recent work by Pittorino et al. (2022) uses LayerNorm as part of symmetry removal, so that the network behavior remains unchanged. Finally, concurrently to this work, Ainsworth et al. (2022) develop a method which achieves zero-barrier interpolation between sufficiently wide ResNet models which have been modified to replace batch normalization with layer normalization. We note that this detail of the normalization layer was out of the scope of their study.

2 Preliminaries

2.1 Linear Interpolation of Neural Networks

Let θ be a weight vector which parameterizes a neural network, and $\mathcal{L}(\theta)$ be a given loss metric (e.g., train or test error). We consider the problem of interpolating between a pair of neural networks with parameters θ_0, θ_1. The loss barrier $B(\theta_0, \theta_1)$ is defined (Entezari et al., 2021) as the maximum increase in loss along the linear path from θ_0 to θ_1, relative to the linear interpolation of the loss values at each of them.

$$B(\theta_0, \theta_1) = \sup_{\alpha \in [0,1]} \Big[\mathcal{L}((1-\alpha)\theta_0 + \alpha\theta_1) \Big] - \left[(1-\alpha)\mathcal{L}(\theta_0) + \alpha\mathcal{L}(\theta_1) \right]. \tag{1}$$

The two networks θ_0 and θ_1 are called linear mode connected if the barrier between them is roughly zero (Frankle et al., 2020). When referring to weights θ_α produced by interpolating between the two weights θ_0 and θ_1, we refer to the network parameterized by θ_α as the interpolated network, and the networks parameterized by θ_0 and θ_1 as the trained or parent networks.

2.2 Permutation Invariance

The class of neural networks with ReLU activation satisfies two invariance properties. Namely, scale invariance or unit-rescaling due to positive homogeneity of ReLU activations (Neyshabur et al., 2015) and the other is permutation of hidden units. Unit-rescaling has been well-studied and empirical evidence suggests that implicit bias of SGD would make the solution converge to a stage where the weights are more balanced (Neyshabur et al., 2015; Wu et al., 2019). Since permutations are equally likely for all SGD solutions, we follow the line of work that focuses on permutation invariance. Permutation invariance is considered in form of permutations of hidden units in each layer of the network, *i.e.*, each layer i with parameters W_i is replaced with $P_i W_i P_{i-1}$ where P_i is a permutation matrix and $P_l = P_0$ is the identity matrix.

For the case of networks with skip connections such as ResNets, care must be taken to apply a permutation in such a manner that the function represented by the network does not change. In particular, the same permutation of hidden units must be applied to all convolutional layers which feed into a single residual stream.

Entezari et al. (2021) conjecture that if permutation invariance is taken into account, SGD solutions for a network trained on the same training data become linear mode connected, *i.e.*, there will be no barrier between them. This conjecture is the subject of the study performed in this paper.

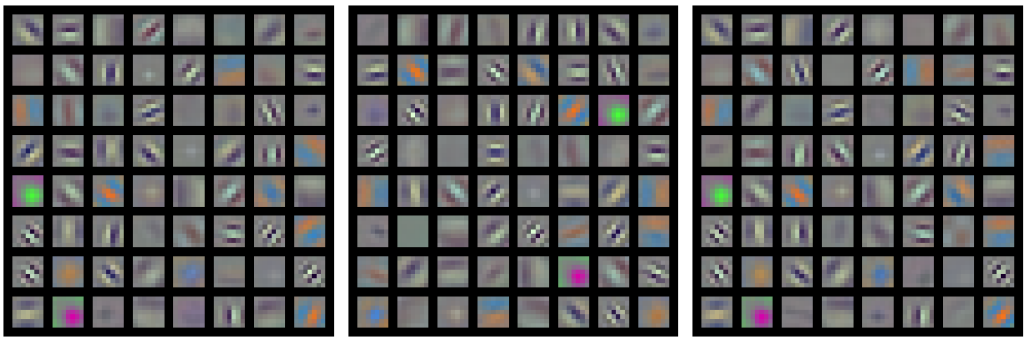


Figure 2: The first-layer convolutional filters of two ResNet50s trained independently on ImageNet with different random seeds. **(left):** The filters of model A. **(middle):** The filters of model B. **(right):** The filters of model B, having been permuted in order to maximize correlation to the corresponding filters of model A. Some paired filters appear nearly identical, while for others there was no close match.

2.3 Neuron Alignment Algorithms

(Li et al., 2015) explore the notion that neurons of one trained network might be aligned with those of another. To perform this alignment for a given layer, they compute the correlation matrix between all pairs of hidden units of the two networks, and then solve the linear assignment problem corresponding to finding a permutation which maximizes the diagonal of this matrix. In notation, let $X_{l,i}^{(0)}$ and $X_{l,i}^{(1)}$ be random variables corresponding to the i-th hidden units (or channels depending on terminology) of the l-th layers in the networks parameterized by θ_0 and θ_1, respectively, and where randomness is over the data distribution. Then we want to find the permutation P_l which maximizes the following the sum of correlations between matched units, *i.e.*, the following cost:

$$\sum_i \mathrm{corr}(X_{l,i}^{(1)}, X_{l,P_l(i)}^{(2)}). \tag{2}$$

In Figure 2, we display the outcome of applying this algorithm to the first layer of convolutional filters in two ResNet50s trained on ImageNet. The alighment results in two sets of 64 7×7 convolutional filters, in which many of the matched pairs appear visually similar.

In the view of the conjecture, this alignment method is a tool for increasing linear mode connectivity of two networks. Specifically, consider two parameterizations θ_0 and θ_1 of a network. Then applying the above alignment method will produce a modified version of θ_1 which has been permuted such that hidden units of the two networks are brought into alignment. The idea of the conjecture is that network weights which are interpolated between θ_0 and this modified θ_1 should have better performance than those which are directly interpolated between θ_0 and θ_1. We next move on to understanding the behavior of such interpolated networks.

3 Collapse of Variance in Interpolated Networks

In this section we empirically investigate the source of the poor performance of interpolated deep networks. We study two scenarios: multi-layer perceptrons (MLPs) (Rosenblatt, 1961) trained on MNIST, and VGG11 (Simonyan & Zisserman, 2014) trained on CIFAR-10 (Krizhevsky et al., 2009).

In Figure 3, we evaluate the effect of depth on barrier. We observe that although applying a suitable permutation to the weights is sufficient to achieve nearly zero-barrier mode connectivity for shallower networks, past a certain depth even the permuted networks are no longer connected. This reproduces the findings of Entezari et al. (2021).

To investigate the behavior of interpolated deep MLPs, we focus on the statistics of the hidden units. For each hidden unit, we measure the variance of its activation across the MNIST training set, see Figure 4. For each layer, we take the average of the variances of all hidden units of that layer, and call this the layerwise variance. We plot in Figure 4 the ratio between the layerwise variances of the interpolated and parent networks for MLP and VGG architectures on MNIST and CIFAR-10 respectively. For the parent network values, we take the layerwise variances to be the average of

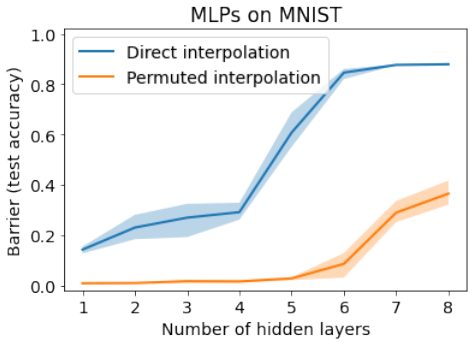


Figure 3: Test accuracy barriers between MLPs trained on MNIST as a function of the number of hidden layers. Each layer has width 512. Barrier size is evaluated over three runs. **Direct interpolation** refers to direct interpolation between two SGD solutions. When MLP depth exceeds six hidden layers, the barrier rapidly increases. This reproduces the findings of Entezari et al. (2021). **permuted interpolation** refers to the barrier between two SGD solutions when we permute one of them such that the barrier is reduced. Neuron alignment is used to find such a permutation.

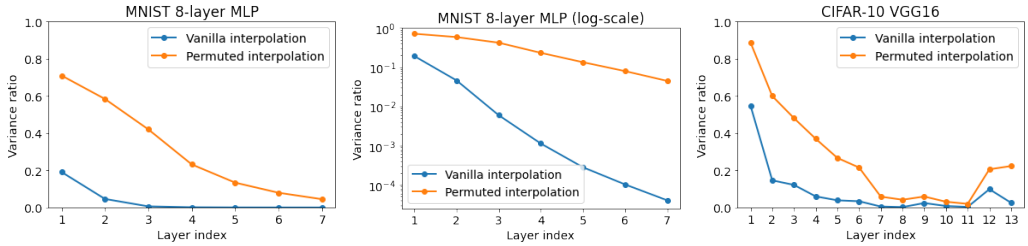


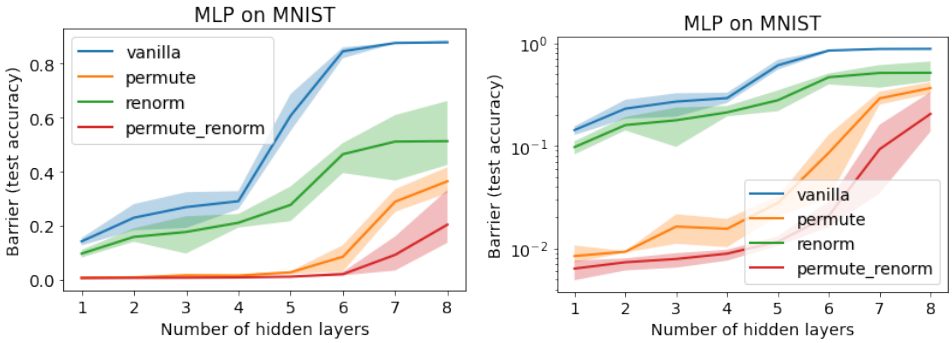
Figure 4: **Variance collapse phenomenon observed in MLP and VGG architectures.** We compute the variance across training data of the activations of each hidden unit in the network. For each layer, we take the mean of variances of the hidden units of that layer, and call this the layerwise variance. Finally, we take the ratios of the calculated layerwise variances in the interpolated network to that in the parent networks. For networks interpolated both with and without a permutation applied to the parent, we observe a progressive collapse in variance which proceeds roughly exponentially in the depth of the layer.

those of the two parents. To be precise, if we suppose that the l-th layers of the two parent networks have layerwise variances σ_0^2 and σ_1^2, and that the l-th layer of the interpolated network has layerwise variance σ_α^2, then we report the ratio $\frac{\sigma_\alpha^2}{(\sigma_0^2+\sigma_1^2)/2}$.

We observe a striking trend in the statistics of the interpolated networks: by the final layer, the activation variance within the directly interpolated network has become 10^4 times smaller than the activation variance of the parent networks. We refer to this phenomenon as *variance collapse*. This progressive decrease in variance seems to occur at a roughly exponential rate, in that the relative variances within each layer of the interpolated networks are roughly a constant factor smaller than those of the previous layer. In the case of the VGG network, the trend is not quite so precise, but many layers still have their variance reduced by more than $10\times$. In all cases, the application of the permutation to align hidden units before interpolating decreases but does not prevent the rate of collapse.

4 Correcting the Activation Statistics

To correct for this statistical phenomenon (variance collapse), we develop the following method. Let θ_0, θ_1 be the weights we wish to interpolate between, with a suitable permutation already applied to θ_1 in order to align the hidden units of the two networks. Let $\alpha \in (0, 1)$ be the desired interpolation coefficient, such that the directly interpolated weights are $\theta_\alpha = (1-\alpha)\theta_0 + \alpha\theta_1$. As discussed in the previous section, we typically observe that the directly interpolated networks have poor performance, with hidden units in later layers having "collapsed" variance.



(a) normal (b) log-scale

Figure 5: **Test accuracy barrier before and after neuron alignment and renormalization.** Test accuracy barriers between MLPs trained on MNIST as a function of number of hidden layers. Each layer has width 512. Barrier size is evaluated over three runs. **vanilla** refers to direct interpolation between two SGD solutions. When MLP depth exceeds six hidden layers, the barrier rapidly increases. This reproduces the findings of Entezari et al. (2021). **permute** refers to the barrier between two SGD solutions when we permute one of them such that the barrier is reduced. Neuron alignment is used to find such a permutation. **renorm** refers to the renormalization correction of variance collapse discussed in Section 4, and **permute-renorm** refers to using neuron alignment and renormalization together. We observe that both permutation and renormalization lead to barrier reduction separately and using them together significantly reduces the barrier leading to almost zero barrier size for a long range of network depths.

We want to compute a rescaled set of weights θ'_α in which this phenomenon is corrected. In particular, let σ_0^2 be the pre-activation variance (w.r.t. the training data) of some particular hidden unit in the network which is parameterized by θ_0. Let σ_1^2 be the variance of the same hidden unit in the network given by θ_1. Then we want the same unit in the network parameterized by θ'_α to have pre-activation variance which is equal to the square of the interpolated standard deviation: $\sigma'^2_\alpha = ((1-\alpha)\sigma_0 + \alpha\sigma_1)^2$.

We propose the following practical method for achieving this. First, we compute the mean and variance of every hidden unit in the parent networks θ_0, θ_1. Next, we modify the interpolated network by adding a new BatchNorm layer after every hidden convolutional or linear layer. In these added BatchNorm layers we also include a channelwise affine transformation (this is the default setting in common learning libraries such as PyTorch and TensorFlow). We set the biases and weights of this transformation to be equal to the interpolated means $(1-\alpha)\mu_0 + \alpha\mu_1$ and standard deviations $(1-\alpha)\sigma_0 + \alpha\sigma_1$. We then recompute the tracked normalization statistics of our added BatchNorm layers; this can be done by a forward pass over the training data in "train mode" in the standard libraries. Finally, we perform "fusion" of the BatchNorm layer back into the previous hidden convolutional or linear layer. This amounts to extracting rescaling and shift coefficients from the BatchNorm statistics and affine parameters, which are respectively multiplied into the weights and added to the bias of the previous layer. We note that the technique of recomputing BatchNorm statistics in order to boost the accuracy of interpolated networks goes back to (Izmailov et al., 2018).

After this process, we are left with a new set of parameters θ'_α in which weights have been rescaled and biases shifted such that hidden units have activation statistics which match the desired interpolated values.

5 Results

In this section we apply our correction method to various scenarios. Specifically we look into MLP, ResNet and VGG architectures and cover MNIST, CIFAR-10, and ImageNet datasets. We investigate the effect of depth and width, as well as studying the choice of ResNet architecture normalization layer between BatchNorm (Ioffe & Szegedy, 2015) and LayerNorm (Ba et al., 2016)), as well as no normalization layers via Fixup (Zhang et al., 2019) initialization.

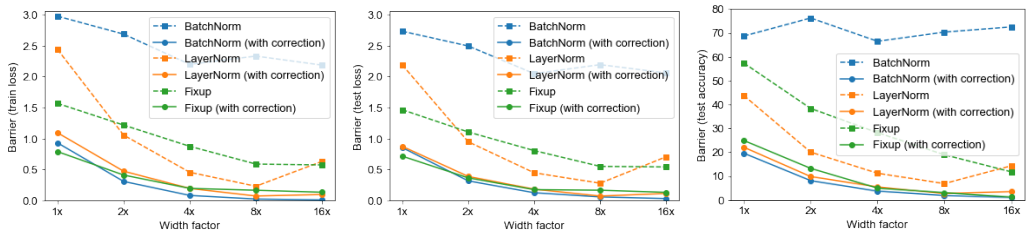


Figure 6: **Effect of width:** We investigate the effect of our proposed correction on barrier between permuted interpolated networks for ResNet20 trained on CIFAR-10, measured across varying choice of width multiplier. **Effect of normalization layer:** We also compare ResNets trained with standard BatchNorm layers to those trained without normalization using Fixup initialization. In each case, we vary the width multiplier from 1 to 16, giving models whose final block ranges from 64 to 512 channels. Solid indicates after the correction and dashed before. **(left):** Barrier is measured in terms of train loss. **(middle):** Barrier is measured in terms of test loss. **(right):** Barrier is measured in terms of test accuracy. We observe that no matter which performance metric we choose for measuring barrier, our correction method significantly reduces barrier between permuted interpolated networks across varying choice of width. Moreover, it works well for any choice of normalization layer.

5.1 MLPs

First, we establish the effect of our correction on the MLPs which were discussed in Section 3. In Figure 5, we plot the barrier as a function of number of layers (depth) for four different interpolation scenarios. **vanilla** refers to direct interpolation between two SGD solutions. **permute** refers to the barrier between two SGD solutions when we permute one of them such that the barrier is reduced; these first two scenarios reproduce Figure 3. **renorm** refers to the renormalization correction of variance collapse discussed in Section 4 as applied to the directly interpolated networks, and **permute-renorm** refers to using neuron alignment and renormalization together. We observe that both permutation and renormalization lead to barrier reduction separately, and using them together reduces the barrier leading to almost zero barrier interpolation for a greater range of network depths.

5.2 Effect of Normalization Layer in ResNets

Next, we consider ResNet networks trained on CIFAR-10 across three different types of normalization layer used during training: BatchNorm, LayerNorm, and Fixup initialization which does not use a normalization layer. We combine this ablation with a study of the effect of width – for each class of networks, we vary the width multiplier from $1\times$, where the final block has 64 channels, to $16\times$ which gives the final block 1024 channels. In Figure 6 we report the permuted interpolation barriers for three metrics: the training set (cross-entropy) loss, test set loss, and test set accuracy. We find that our correction method is able to significantly reduce the barrier in all cases.

For the case of LayerNorm, we find that the interpolated networks have reasonably high performance even without applying our statistical correction. We hypothesize that his has to do with the fact that LayerNorm applies renormalization at evaluation-time, which can partially correct the variance collapse. In contrast, BatchNorm layers do not compute fresh normalization statistics at evaluation time, instead using running mean and variance statistics that are computed during train time.

(Ainsworth et al., 2022) studied interpolation for the case of ResNet and VGG networks which use LayerNorm as their normalization layer. We note that our reported barriers for (uncorrected) LayerNorm interpolations are higher than theirs. This may be due to a difference in LayerNorm implementation – we normalize with a single mean and variance computed across each entire layer of activations, but it is also common to normalize each spatial location separately. Regardless, we note that with our proposed correction, the barriers for standard BatchNorm-based ResNets are lower across all widths than those reported by (Ainsworth et al., 2022) for uncorrected LayerNorm-based networks.

5.3 Effect of width and depth

In Figure 7, we report the results of applying our proposed correction to VGG networks of varying depth. In the left plot, the barrier is measured in terms of train and test loss. In the right plot, the

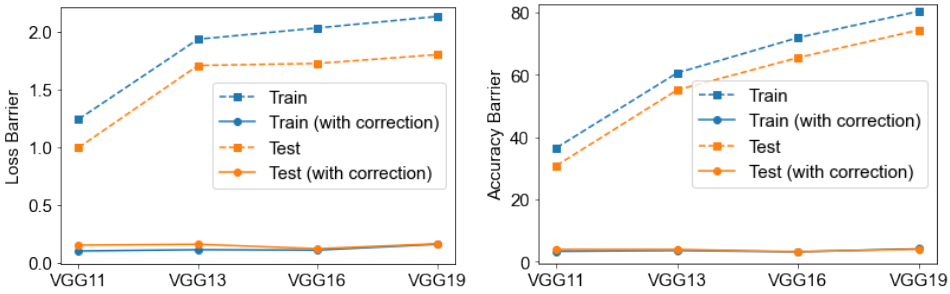


Figure 7: **Effect of depth:** We investigate the effect of our proposed correction on the interpolation barrier between neuron-aligned VGG networks which are trained on CIFAR-10. We use varying depth VGG networks, from 11 to 19 layers. **(left):** Barrier is measured in terms of train and test loss (blue and orange lines respectively). **(right):** Barrier is measured in terms of respective accuracies. We observe that for all barrier metrics and depths, our correction method significantly reduces the interpolation barrier. We also note that without the correction, the barrier between networks increases with depth, but with the correction, the barrier appears relatively flat with depth.

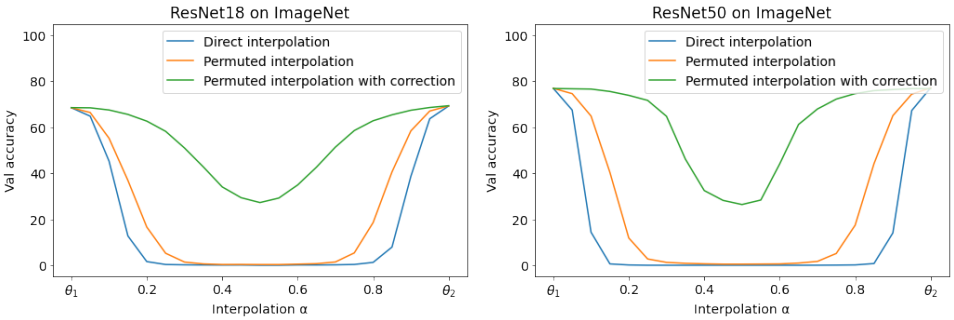


Figure 8: ImageNet validation accuracy curves for interpolated ResNets. **Direct interpolation** refers to direct interpolation between two SGD solutions. **Permuted interpolation** refers to the barrier between two SGD solutions after aligning neurons (Li et al., 2015; Ainsworth et al., 2022). **Permuted interpolation with correction** refers to our proposed method. We note that our method leads to significantly improved barrier reduction between the two networks.

barrier is measured in terms of the respective accuracies. We observe that the uncorrected interpolations have a large barrier which increases with the depth. In contrast, the barrier for interpolated networks with our correction applied stays roughly constant across all depths we consider.

5.4 IMAGENET

Next, we explore the interpolation barriers of ResNet models trained on ImageNet (Deng et al., 2009). We present interpolated accuracy curves of ResNet18 and ResNet50 architectures in Figure 8. We observe that both the direct and permuted interpolations appear to have maximum barrier in that the midpoints do not perform better than random chance. With the correction applied, performance at the midpoint increases to 26.4% and 27.3% respectively. To the best of our knowledge this is the first method that leads to nonzero barrier reduction for the ImageNet dataset. We note however that the barrier in this scenario is still high at 41.7% and 50.4% for ResNet18 and ResNet50 respectively. As a future direction, it is of interest to look for extra interventions which may lead to this barrier being further reduced.

5.5 PUTTING THINGS TOGETHER

In Figure 9, we visualize the relationship between test error and (corrected) barrier for all of our models which were trained on CIFAR-10. There is generally a relationship between the test error and size of the barrier in that our models with smaller test error also tend to have smaller barrier. Comparing this to Figure 4(c) of (Entezari et al., 2021), we note the same relationship between

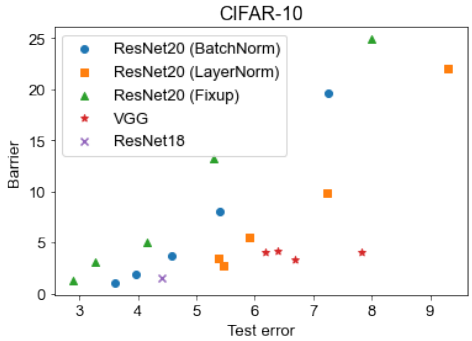


Figure 9: X-axis: test accuracy of parent networks. Y-axis: barrier as measured in test accuracy between parent network and interpolated network (with correction) for all the models trained on CIFAR-10 in this paper.

barrier values to test error. It is therefore reasonable to conclude that the high barrier of ImageNet models stems in part simply from the relatively higher test error of said models, as compared to our models which were trained on CIFAR-10.

6 Conclusion and Future Work

In this paper we investigated the source of the typically high barrier when interpolating between neuron-aligned deep networks. We observed a phenomenon of *variance collapse* in which the hidden units of interpolated networks have significantly reduced variance relative to those of their trained parent networks. We showed that this phenomenon is general and scales with depth, and proposed a method of correcting for it by rescaling the activations of the interpolated network. We demonstrated experimentally that this correction significantly improves the performance of intepolated networks, including reducing the interpolation barrier for a pair of ResNet18s trained on CIFAR-10 to 1.5% absolute test error.

Our work provides support for the conjecture of Entezari et al. (2021) by showing that once permutation invariance and the statistical correction of variance collapse are taken into account, it is possible to move all SGD solutions into a single basin with a low barrier between all pairs. On the empirical side, we hope that a better understanding of the neural network loss landscape and in particular linear interpolation will lead to improvements in applications including weight-space ensembling, checkpoint averaging, and robust finetuning.

7 Reproducibility and Ethics Statement

All the datasets in this paper are public data and cited in the paper. The code can be found at this anonymous link `https://anonymous.4open.science/r/Network-Permutations-2BEC/README.md`

We believe that our submission raises no questions regarding the Code of Ethics as we use open public datasets and model. Our study also does not involve human subjects, potentially harmful insights, methodologies and applications, potential conflicts of interest and sponsorship, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues. We also referred to close related works and compare our results wherever applicable.

References

Martial Agueh and Guillaume Carlier. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924, 2011.

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Carlo Baldassi, Fabrizio Pittorino, and Riccardo Zecchina. Shaping the learning landscape in neural networks around wide flat minima. *Proceedings of the National Academy of Sciences*, 117(1): 161–170, 2020.

Johanni Brea, Berfin Simsek, Bernd Illing, and Wulfram Gerstner. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape. *arXiv preprint arXiv:1907.02911*, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. IEEE, 2009.

Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht. Essentially no barriers in neural network energy landscape. In *International conference on machine learning*, pp. 1309–1318. PMLR, 2018.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020.

C Daniel Freeman and Joan Bruna. Topology and geometry of half-rectified network optimization. *arXiv preprint arXiv:1611.01540*, 2016.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. *Advances in neural information processing systems*, 31, 2018.

Mario Geiger, Stefano Spigler, Stéphane d'Ascoli, Levent Sagun, Marco Baity-Jesi, Giulio Biroli, and Matthieu Wyart. Jamming transition as a paradigm to understand the loss landscape of deep neural networks. *Physical Review E*, 100(1):012115, 2019.

Hao Guo, Jiyong Jin, and Bin Liu. Stochastic weight averaging revisited, 2022.

Xiaoxi He, Zimu Zhou, and Lothar Thiele. Multi-task zipping via layer-wise neuron sharing. *arXiv preprint arXiv:1805.09791*, 2018.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. Patching open-vocabulary models by interpolating weights. *arXiv preprint arXiv:2208.05592*, 2022.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. PMLR, 2015.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization, 2019.

Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. Linear connectivity reveals generalization strategies. *arXiv preprint arXiv:2205.12411*, 2022.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 and cifar-10 (canadian institute for advanced research), 2009. URL `http://www.cs.toronto.edu/~kriz/cifar.html`. MIT License.

Dawei Li, Tian Ding, and Ruoyu Sun. Over-parameterized deep neural networks have no strict local minima for any continuous activations. *arXiv preprint arXiv:1812.11039*, 2018.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*, 2017.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. Convergent learning: Do different neural networks learn the same representations? *arXiv preprint arXiv:1511.07543*, 2015.

Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *arXiv preprint arXiv:2003.00307*, 2020.

Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. Path-SGD: Path-normalized optimization in deep neural networks. *arXiv preprint arXiv:1506.02617*, 2015.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 5949–5958, 2017.

Quynh Nguyen, Mahesh Chandra Mukkamala, and Matthias Hein. On the loss landscape of a class of deep neural networks with no bad local valleys. *arXiv preprint arXiv:1809.10749*, 2018.

Fabrizio Pittorino, Carlo Lucibello, Christoph Feinauer, Enrico M Malatesta, Gabriele Perugini, Carlo Baldassi, Matteo Negri, Elizaveta Demyanenko, and Riccardo Zecchina. Entropic gradient descent algorithms and wide flat minima. *arXiv preprint arXiv:2006.07897*, 2020.

Fabrizio Pittorino, Antonio Ferraro, Gabriele Perugini, Christoph Feinauer, Carlo Baldassi, and Riccardo Zecchina. Deep networks on toroids: Removing symmetries reveals the structure of flat regions in the landscape geometry. *arXiv preprint arXiv:2202.03038*, 2022.

Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY, 1961.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Berfin Şimşek, François Ged, Arthur Jacot, Francesco Spadaro, Clément Hongler, Wulfram Gerstner, and Johanni Brea. Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances. *arXiv preprint arXiv:2105.12221*, 2021.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33:22045–22055, 2020.

Norman Tatro, Pin-Yu Chen, Payel Das, Igor Melnyk, Prasanna Sattigeri, and Rongjie Lai. Optimizing mode connectivity via neuron alignment. *Advances in Neural Information Processing Systems*, 33:15300–15311, 2020.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pp. 23965–23998. PMLR, 2022a.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971, 2022b.

Xiaoxia Wu, Edgar Dobriban, Tongzheng Ren, Shanshan Wu, Zhiyuan Li, Suriya Gunasekar, Rachel Ward, and Qiang Liu. Implicit regularization and convergence for weight normalization. *arXiv preprint arXiv:1911.07956*, 2019.

Shi-hua Zhan, Juan Lin, Ze-jun Zhang, and Yi-wen Zhong. List-based simulated annealing algorithm for traveling salesman problem. *Intell. Neuroscience*, 2016:8, March 2016.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *Proceedings of the International Conference on Learning Representations*, 2017.

Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019.