GIT RE-BASIN: MERGING MODELS MODULO PERMUTATION SYMMETRIES

Anonymous authors
Paper under double-blind review

ABSTRACT

The success of deep learning is due in large part to our ability to solve certain massive non-convex optimization problems with relative ease. Though non-convex optimization is NP-hard, simple algorithms – often variants of stochastic gradient descent – exhibit surprising effectiveness in fitting large neural networks in practice. We argue that neural network loss landscapes often contain (nearly) a single basin after accounting for all possible permutation symmetries of hidden units a la Entezari et al. (2021). We introduce three algorithms to permute the units of one model to bring them into alignment with a reference model in order to merge the two models in weight space. This transformation produces a functionally equivalent set of weights that lie in an approximately convex basin near the reference model. Experimentally, we demonstrate the single basin phenomenon across a variety of model architectures and datasets, including the first (to our knowledge) demonstration of zero-barrier linear mode connectivity between independently trained ResNet models on CIFAR-10 and CIFAR-100. Additionally, we investigate intriguing phenomena relating model width and training time to mode connectivity. Finally, we discuss shortcomings of the linear mode connectivity hypothesis, including a counterexample to the single basin theory.

1 INTRODUCTION

We investigate the unreasonable effectiveness of stochastic gradient descent (SGD) algorithms on the high-dimensional non-convex optimization problems of deep learning. In particular,

1. Why does SGD thrive in optimizing high-dimensional non-convex deep learning loss landscapes despite being noticeably less robust in other non-convex optimization settings, like policy learning (Ainsworth et al., 2021), trajectory optimization (Kelly, 2017), and recommender systems (Kang et al., 2016)?
2. What are all the local minima? When linearly interpolating between initialization and final trained weights, why does the loss smoothly and monotonically decrease (Goodfellow & Vinyals, 2015; Frankle, 2020; Lucas et al., 2021; Vlaar & Frankle, 2021)?
3. How can two independently trained models with different random initializations and data batch orders inevitably achieve nearly identical performance? Furthermore, why do their training loss curves often look identical?

We posit that the final phenomenon referenced in item 3 points to the existence of some yet uncharacterized invariance(s) in the training dynamics that cause independent training runs to exhibit almost identical characteristics. Hecht-Nielsen (1990) noted the permutation symmetries of hidden units in neural networks; briefly, one can swap any two units of a hidden layer in a network and – assuming weights are adjusted accordingly – network functionality will not change. Recently, Entezari et al. (2021) conjectured that these permutation symmetries may let us linearly connect points in weight space with no detriment to the loss.

Conjecture 1 (Permutation invariance, informal (Entezari et al., 2021))**.** Most SGD solutions belong to a set whose elements can be permuted so that no barrier (as in Definition 2.2) exists on the linear interpolation between any two permuted elements.

ARCHITECTURE	NUM. PERMUTATION SYMMETRIES
MLP (3 layers, 512 width)	$10 \wedge 3498$
VGG16	$10 \wedge 35160$
ResNet50	$10 \wedge 55109$
Atoms in the observable universe	$10 \wedge 82$

Table 1: **Permutation symmetries of deep learning models vs. an upper estimate on the number of atoms in the known, observable universe.** Deep learning loss landscapes contain incomprehensible amounts of geometric repetition.

We refer to such solutions as being *linearly mode connected* (LMC) (Frankle et al., 2020). If true, Conjecture 1 will both materially expand our understanding of how SGD works in the context of deep learning and offer a credible explanation for the preceding phenomena, in particular.

Contributions. In this paper, we attempt to uncover what invariances may be responsible for the phenomena cited above and the unreasonable effectiveness of SGD in deep learning. We make the following contributions:

1. **Matching methods.** We propose three new algorithms, grounded in concepts and techniques from combinatorial optimization, to align the weights of two independently trained models. Where appropriate, we prove hardness results for these problems and propose approximation algorithms. Our fastest method identifies permutations in mere seconds on current hardware.
2. **Relationship to optimization algorithms.** We demonstrate by means of counterexample that linear mode connectivity is an emergent property of training procedures, not of model architectures. We connect this result to prior work on the implicit biases of SGD.
3. **Experiments, including zero-barrier LMC for ResNets.** Empirically, we explore the existence of linear mode connectivity modulo permutation symmetries in experiments across MLPs, CNNs, and ResNets trained on MNIST, CIFAR-10, and CIFAR-100. We contribute the first-ever demonstration of zero-barrier LMC between two independently trained ResNets. We explore the relationship between LMC and model width as well as training time. Finally, we show evidence of our methods' ability to combine models trained on independent datasets into a merged model that outperforms both input models in terms of test loss (but not accuracy) and is no more expensive in compute or memory than either input model.

2 Background

Although our methods can be applied to arbitrary model architectures, we proceed with the multilayer perceptron (MLP) for its ease of presentation (Bishop, 2007). Consider an L-layer MLP,

$$f(\boldsymbol{x}; \Theta) = \boldsymbol{z}_{L+1}, \quad \boldsymbol{z}_{\ell+1} = \sigma(\boldsymbol{W}_\ell \boldsymbol{z}_\ell + \boldsymbol{b}_\ell), \quad \boldsymbol{z}_1 = \boldsymbol{x},$$

where σ denotes an element-wise nonlinear activation function. Furthermore, consider a loss, $\mathcal{L}(\Theta)$, that measures the suitability of a particular set of weights Θ towards some goal, e.g., fitting to a training dataset.

Central to our investigation is the phenomenon of *permutation symmetries* of weight space. Given Θ, we can apply some permutation to the output features of any intermediate layer, ℓ, of the model, denoted by a permutation matrix $\boldsymbol{P} \in S_d$,[1]

$$\boldsymbol{z}_{\ell+1} = \boldsymbol{P}^\top \boldsymbol{P} \boldsymbol{z}_{\ell+1} = \boldsymbol{P}^\top \boldsymbol{P} \sigma(\boldsymbol{W}_\ell \boldsymbol{z}_\ell + \boldsymbol{b}_\ell) = \boldsymbol{P}^\top \sigma(\boldsymbol{P}\boldsymbol{W}_\ell \boldsymbol{z}_\ell + \boldsymbol{P}\boldsymbol{b}_\ell)$$

for σ, an element-wise operator. It follows that as long as we reorder the input weights of layer $\ell+1$ according to $\boldsymbol{P}^\top$, we will have a functionally equivalent model. To be precise, if we define Θ' to be identical to Θ with the exception of

$$\boldsymbol{W}'_\ell = \boldsymbol{P}\boldsymbol{W}_\ell, \quad \boldsymbol{b}'_\ell = \boldsymbol{P}\boldsymbol{b}_\ell, \quad \boldsymbol{W}'_{\ell+1} = \boldsymbol{W}_{\ell+1}\boldsymbol{P}^\top,$$

[1]We denote the set of all $d \times d$ permutation matrices – isomorphic to the symmetric group – as S_d, to the possible chagrin of pure mathematicians.

then the two models are functionally equivalent: $f(\boldsymbol{x};\Theta) = f(\boldsymbol{x};\Theta')$ for all inputs $\boldsymbol{x}$. This implies that for any trained weights Θ, there is an entire equivalence class of functionally equivalent weight assignments, not just one such Θ, and convergence to any one specific element of this equivalence class, as opposed to any others, is determined only by random seed. We denote a functionality-preserving permutation of weights as $\pi(\Theta)$.

Consider the task of reconciling the weights of two, independently trained models, A and B, with weights Θ_A and Θ_B, respectively, such that we can linearly interpolate between them. We assume that models A and B were trained with equivalent architectures but different random initializations, data orders, and potentially different hyperparameters or datasets, as well. Our central question is: Given Θ_A and Θ_B, can we identify some π such that when linearly interpolating between Θ_A and $\pi(\Theta_B)$, all intermediate models enjoy performance similar to Θ_A and Θ_B?

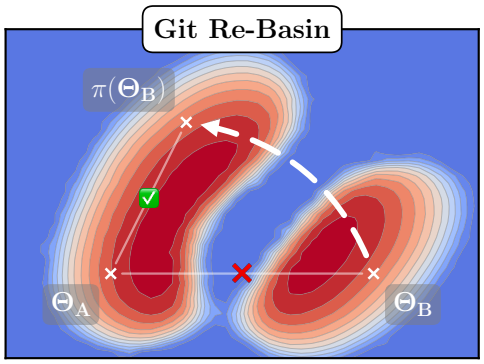

Figure 1: **Git Re-Basin merges models by teleporting solutions into a single basin.** Θ_B is permuted into $\pi(\Theta_B)$ so that it lies in the same basin as Θ_A.

We base any claims of loss landscape convexity on the usual definition of multi-dimensional convexity in terms of one-dimensional convexity per

Definition 2.1 (Convexity)**.** A function $f : \mathbb{R}^D \to \mathbb{R}$ is convex if every one-dimensional slice is convex, i.e., for all $x, y \in \mathbb{R}^D$, the function $g(\lambda) = f((1-\lambda)x + \lambda y)$ is convex in λ.

Due to Definition 2.1, it suffices to show that arbitrary one-dimensional slices of a function are convex in order to reason about the convexity of complex, high-dimensional functions. In practice, we rarely observe perfect convexity but instead hope to approximate it as closely as possible. Following Draxler et al. (2018), Entezari et al. (2021), Frankle et al. (2020) and others, we measure approximations to convexity via "barriers."

Definition 2.2 (Loss barrier (Frankle et al., 2020))**.** Given two points Θ_A, Θ_B such that $\mathcal{L}(\Theta_A) \approx \mathcal{L}(\Theta_B)$, the *loss barrier* is defined as $\max_{\lambda\in[0,1]} \mathcal{L}((1-\lambda)\Theta_A + \lambda\Theta_B) - \frac{1}{2}(\mathcal{L}(\Theta_A) + \mathcal{L}(\Theta_B))$.

Loss barriers are non-negative, with zero indicating an interpolation of flat or positive curvature.

3 Permutation Selection Methods

We introduce three methods of matching units between model A and model B. Further, we present an extension to simultaneously merging multiple models in Appendix A.10 and an appealing but failed method in Appendix A.11.

3.1 Matching Activations

Following the classic Hebbian mantra, "[neural network units] that fire together, wire together" (Hebb, 2005), we propose associating units across two models by performing regression between their activations. Matching activations between models is compelling since it captures the intuitive notion that two models must learn similar features to accomplish the same task (Li et al., 2016). Provided activations for each model, we aim to associate each unit in A with a unit in B. It stands to reason that a linear relationship may exist between the activations of the two models. We fit this into the regression framework by constraining ordinary least squares (OLS) to solutions in the set of permutation matrices, S_d. For activations of the ℓ'th layer, let $\boldsymbol{Z}^{(A)}, \boldsymbol{Z}^{(B)} \in \mathbb{R}^{d\times n}$ denote the d-dim. activations for all n training data points in models A and B, respectively. Then,

$$\boldsymbol{P}_\ell = \underset{\boldsymbol{P}\in S_d}{\arg\min} \sum_{i=1}^{n} \|\boldsymbol{Z}^{(A)}_{:,i} - \boldsymbol{P}\boldsymbol{Z}^{(B)}_{:,i}\|^2 = \underset{\boldsymbol{P}\in S_d}{\arg\max} \langle \boldsymbol{P}, \boldsymbol{Z}^{(A)}(\boldsymbol{Z}^{(B)})^\top \rangle_F, \tag{1}$$

where $\langle \boldsymbol{A}, \boldsymbol{B} \rangle_F = \sum_{i,j} A_{i,j} B_{i,j}$ denotes the Frobenius inner product between real-valued matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. Conveniently, eq. (1) constitutes a "linear assignment problem" (LAP) (Bertsekas, 1998) for which efficient, practical algorithms are known. Having solved this assignment problem on each layer, we can then permute the weights of model B to match model A as closely as possible

$$\boldsymbol{W}'_\ell = \boldsymbol{P}_\ell \boldsymbol{W}_\ell^{(B)} \boldsymbol{P}_{\ell-1}^\top, \quad \boldsymbol{b}'_\ell = \boldsymbol{P}_\ell \boldsymbol{b}_\ell^{(B)}$$

for each layer ℓ, producing weights Θ' with activations that align as closely possible with Θ_A.

Computationally, this entire process is relatively lightweight: the $\boldsymbol{Z}^{(A)}$ and $\boldsymbol{Z}^{(B)}$ matrices can be computed in a single pass over the training dataset, and, in practice, a full run through the training dataset may be unnecessary. Solving eq. (1) is possible due to well-established, polynomial-time algorithms for solving the linear assignment problem (Kuhn, 2010; Jonker & Volgenant, 1987; Crouse, 2016). Also, conveniently, the activation matching at each layer is independent of the matching at every other layer, resulting in a separable and straightforward optimization problem; this advantage will not be enjoyed by the following methods.

Dispensing with regression, one could similarly associate units by matching against a matrix of cross-correlation coefficients in place of $\boldsymbol{Z}^{(A)}(\boldsymbol{Z}^{(B)})^\top$. We observed correlation matching to work equally well but found OLS regression matching to be more principled and easier to implement.

Activation matching has been independently studied in Tatro et al. (2020); Singh & Jaggi (2020) albeit not from the perspective of OLS regression.

3.2 Matching Weights

Instead of associating units by their activations, we could alternatively inspect the weights of the model itself. Consider the first layer weights, $\boldsymbol{W}_1$; each row of $\boldsymbol{W}_1$ corresponds to a single feature. If two such rows were equal, they would compute exactly the same feature (ignoring bias terms for the time being). And, if $[\boldsymbol{W}_1^{(A)}]_{i,:} \approx [\boldsymbol{W}_1^{(B)}]_{j,:}$, it stands to reason that units i and j should be associated. Extending this idea to every layer, we are inspired to pursue the optimization

$$\arg\min_\pi \ \|\text{vec}(\Theta_A) - \text{vec}(\pi(\Theta_B))\|^2 \ = \ \arg\max_\pi \ \text{vec}(\Theta_A) \cdot \text{vec}(\pi(\Theta_B)).$$

We can re-express this in terms of the full weights,

$$\arg\max_{\pi=\{\boldsymbol{P}_i\}} \langle \boldsymbol{W}_1^{(A)}, \boldsymbol{P}_1 \boldsymbol{W}_1^{(B)} \rangle_F + \langle \boldsymbol{W}_2^{(A)}, \boldsymbol{P}_2 \boldsymbol{W}_2^{(B)} \boldsymbol{P}_1^\top \rangle_F + \cdots + \langle \boldsymbol{W}_L^{(A)}, \boldsymbol{W}_L^{(B)} \boldsymbol{P}_{L-1}^\top \rangle_F, \quad (2)$$

resulting in another matching problem. We term this formulation the "sum of bilinear assignments problem" (SOBLAP). Unfortunately, this matching problem is thornier than the classic linear assignment matching problem presented in eq. (1). Unlike LAP, we are interested in permuting *both* the rows and columns of $\boldsymbol{W}_\ell^{(B)}$ to match $\boldsymbol{W}_\ell^{(A)}$, which fundamentally differs from permuting only rows or only columns. We formalize this difficulty as follows.

Lemma 1. *The sum of a bilinear assignments problem (SOBLAP) is NP-hard and admits no polynomial-time constant-factor approximation scheme for $L > 2$.*

Lemma 1 contrasts starkly with classical LAP, for which polynomial-time algorithms are known.

Undeterred, we propose a approximation algorithm for SOBLAP. Looking at a single $\boldsymbol{P}_\ell$ while holding the others fixed, we observe that the problem can be reduced to a classic LAP,

$$\begin{aligned} &\arg\max_{\boldsymbol{P}_\ell} \langle \boldsymbol{W}_\ell^{(A)}, \boldsymbol{P}_\ell \boldsymbol{W}_\ell^{(B)} \boldsymbol{P}_{\ell-1}^\top \rangle_F + \langle \boldsymbol{W}_{\ell+1}^{(A)}, \boldsymbol{P}_{\ell+1} \boldsymbol{W}_{\ell+1}^{(B)} \boldsymbol{P}_\ell^\top \rangle_F \\ &\qquad = \arg\max_{\boldsymbol{P}_\ell} \langle \boldsymbol{P}_\ell, \boldsymbol{W}_\ell^{(A)} \boldsymbol{P}_{\ell-1} (\boldsymbol{W}_\ell^{(B)})^\top + (\boldsymbol{W}_{\ell+1}^{(A)})^\top \boldsymbol{P}_{\ell+1} \boldsymbol{W}_{\ell+1}^{(B)} \rangle_F. \end{aligned}$$

This leads to a convenient coordinate descent algorithm: go through each layer and greedily select its best $\boldsymbol{P}_\ell$. Repeat until convergence. We present this in Algorithm 1.

Although we present Algorithm 1 in terms of an MLP without bias terms, in practice our implementation can handle the weights of models of nearly arbitrary architectures, including bias terms, residual connections, convolutional layers, attention mechanisms, and so forth. We propose an extension of Algorithm 1 to merging more than two models at a time in Appendix A.10.

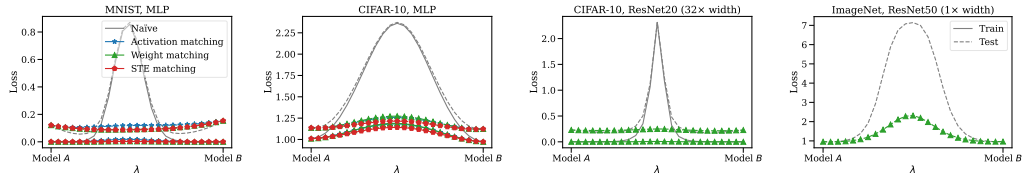

Figure 2: **Linear mode connectivity is possible after permuting.** Loss when interpolating between models trained on MNIST, CIFAR-10, and ImageNet. In all cases we can significantly improve over naïve interpolation. Straight-through estimator matching performs best but is very computationally expensive. Weight and activation matching perform similarly, although weight matching is orders of magnitude faster and does not rely on the input data distribution. We hypothesize that the ImageNet barrier could be reduced by increasing the model width as in Section 5.3.

Algorithm 1: PERMUTATIONCOORDINATEDESCENT

Given: Model weights $\Theta_A = \left\{\boldsymbol{W}_1^{(A)}, \ldots, \boldsymbol{W}_L^{(A)}\right\}$ and $\Theta_B = \left\{\boldsymbol{W}_1^{(B)}, \ldots, \boldsymbol{W}_L^{(B)}\right\}$

Result: A permutation $\pi = \{\boldsymbol{P}_1, \ldots, \boldsymbol{P}_{L-1}\}$ of Θ_B such that $\text{vec}(\Theta_A) \cdot \text{vec}(\pi(\Theta_B))$ is approximately maximized.

Initialize: $\boldsymbol{P}_1 \leftarrow \boldsymbol{I}, \ldots, \boldsymbol{P}_{L-1} \leftarrow \boldsymbol{I}$
repeat
  for $\ell \in$ RANDOMPERMUTATION$(1, \ldots, L-1)$ **do**
    $\boldsymbol{P}_\ell \leftarrow$ SOLVELAP $\left(\boldsymbol{W}_\ell^{(A)} \boldsymbol{P}_{\ell-1} (\boldsymbol{W}_\ell^{(B)})^\top + (\boldsymbol{W}_{\ell+1}^{(A)})^\top \boldsymbol{P}_{\ell+1} \boldsymbol{W}_{\ell+1}^{(B)}\right)$
  end
until *convergence*

Lemma 2. *Algorithm 1 terminates.*

Our experiments showed this algorithm to be fast in terms of both iterations necessary for convergence and wall-clock time, generally on the order of seconds to a few minutes.

Unlike the activation matching method presented in Section 3.1, weight matching ignores the data distribution entirely. Ignoring the input data distribution and therefore the loss landscape handicaps weight matching but allows it to be much faster. We therefore anticipate its potential application in fields such as finetuning (Devlin et al., 2019; Wortsman et al., 2022b;a), federated learning (McMahan et al., 2017; Konečný et al., 2016a;b), and model patching (Matena & Raffel, 2021; Sung et al., 2021; Raffel, 2021). In practice, we found weight matching to be surprisingly competitive with data-aware methods. Section 5 studies this trade-off.

3.3 LEARNING PERMUTATIONS WITH A STRAIGHT-THROUGH ESTIMATOR

Inspired by the success of straight-through estimators (STEs) in other discrete optimization problems (Bengio et al., 2013; Kusupati et al., 2021; Rastegari et al., 2016; Courbariaux & Bengio, 2016), we attempt here to "learn" the ideal permutation of weights $\pi(\Theta_B)$. Specifically, our goal is to optimize

$$\min_{\tilde{\Theta}_B} \mathcal{L}\left(\frac{1}{2}\left(\Theta_A + \text{proj}\left(\tilde{\Theta}_B\right)\right)\right), \qquad \text{proj}(\Theta) \triangleq \arg\max_{\pi} \text{vec}(\Theta) \cdot \text{vec}(\pi(\Theta_B)), \tag{3}$$

where $\tilde{\Theta}_B$ denotes an approximation of $\pi(\Theta_B)$, allowing us to implicitly optimize π. However, eq. (3) involves inconvenient non-differentiable projection operations, $\text{proj}(\cdot)$, complicating the optimization. We overcome this via a "straight-through" estimator: we parameterize the problem in terms of a set of weights $\tilde{\Theta}_B \approx \pi(\Theta_B)$. In the forward pass, we project $\tilde{\Theta}_B$ to the closest realizable $\pi(\Theta_B)$. In the backwards pass, we then switch back to the unrestricted weights $\tilde{\Theta}_B$. In this way, we

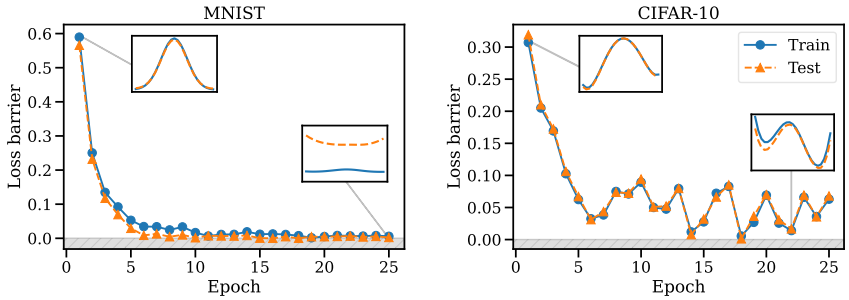

Figure 3: **Linear mode connectivity is challenging at initialization.** We show loss barriers per training time for MLPs trained on MNIST (left) and CIFAR-10 (right). Loss interpolation plots are inlaid to highlight results in initial and later epochs. LMC manifests gradually throughout training. We hypothesize that the variance in CIFAR-10 training is higher due to our MLP architecture being under-powered relative to the dataset. (Y-axis scales differ in each inlaid plot.)

are guaranteed to stay true to the projection constraints in evaluating the loss but can still compute usable gradients at our current parameters, $\tilde{\Theta}_B$.[2]

Conveniently, we can re-purpose Algorithm 1 to solve $\text{proj}(\tilde{\Theta}_B)$. Furthermore, we found that initializing $\tilde{\Theta}_B = \Theta_A$ performed better than random initialization. This is to be expected immediately at initialization since the initial matching will be equivalent to the weight matching method of Section 3.1. However, it is not immediately clear why these solutions continue to outperform a random initialization asymptotically.

Unlike the aforementioned methods, Algorithm 2 attempts to explicitly "learn" the best permutation π using a conventional training loop. By initializing to the weight matching solution of Section 3.2 and leveraging the data distribution as in Section 3.1, it seeks to offer a best-of-both-worlds solution. However, this comes at a very steep computational cost relative to the other two methods.

4 Interlude: A Counterexample

Before arguing for the presence of linear mode connectivity, we step back to consider its limitations. In Appendix A.6 we present a counterexample proving that there exist models that do not enjoy LMC under any permutation of weights. Since our two networks have small width, we can simply inspect all four possible permutations of the intermediate units. We visualize this in Figure 6. Critically, no permutation results in LMC.

We present this counterexample in part to establish some basic intuition for Conjecture 1 but more importantly to highlight that any success interpolating between permuted networks is due to inherent bias in the optimization algorithms used and the types of basins they are attracted to, not the network architectures themselves. To the extent we achieve LMC in practice, it follows that it is an artifact of our optimization methods, not our architectures.

We also note that there are invariances beyond permutation symmetries: It is possible to move features between layers, re-scale layers, and so forth. Prior works noted the feature/layer association (Nguyen et al., 2021) and re-scaling invariances (Ainsworth et al., 2018). The importance of these other symmetries and their interplay with optimization algorithms remains unclear.

5 Experiments

Our base methodology is to separately train two models, A and B, starting from different random initializations and with different random batch orders, resulting in trained weights Θ_A and Θ_B,

[2]Note again that projecting according to inner product distance is equivalent to projecting according to the L_2 distance when parameterizing the estimator based on the B endpoint. We also experimented with learning the midpoint directly, $\tilde{\Theta} \approx \frac{1}{2}(\Theta_A + \pi(\Theta_B))$, in which case the L_2 and inner product projections diverge. In testing all possible variations, we found that optimizing the B endpoint had a slight advantage, but all possible variations performed admirably.

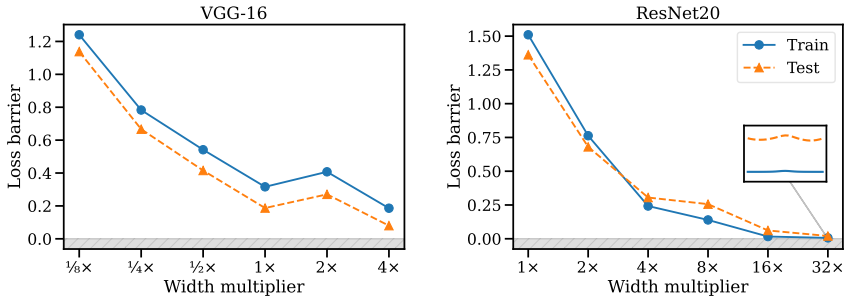

Figure 4: **Wider models exhibit better linear mode connectivity.** Training convolutional and ResNet architectures on CIFAR-10, we ablate their width and visualize loss barriers after weight matching. Notably, we achieve zero-barrier linear mode connectivity between ResNet models, the first such demonstration to the authors' knowledge.

respectively. We then evaluate slices through the loss landscape, $\mathcal{L}((1-\lambda)\Theta_A + \lambda\pi(\Theta_B))$ for $\lambda \in [0, 1]$, where π is selected according to the methods presented in Section 3.[3] Ideally, we seek a completely flat or even convex one-dimensional slice. As discussed in Section 2, the ability to exhibit this behavior for arbitrary Θ_A, Θ_B empirically suggests that the loss landscape contains only a single basin modulo permutation symmetries.

We remark that a failure to find a π such that linear mode connectivity holds cannot rule out the existence of a satisfactory permutation. Given the astronomical number of permutation symmetries, Conjecture 1 is essentially impossible to disprove for any realistically wide model architecture.

5.1 Loss Landscapes Before and After Matching

We present results for models trained on MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky, 2009), and ImageNet (Deng et al., 2009) in Figure 2. Naïve interpolation ($\pi(\Theta_B) = \Theta_B$) substantially degrades performance when interpolating. On the other hand, the methods introduced in Section 3 can achieve much better barriers. We achieve zero-barrier linear mode connectivity on MNIST with all three methods, although activation matching performs just slightly less favorably than weight matching and straight-through estimator (STE) matching. We especially note that the test loss landscape becomes convex after applying our weight matching and STE permutations! In other words, our interpolation actually yields a merged model that outperforms both models A and B. We elaborate on this phenomenon in Section 5.4 and Appendix A.10.

On ImageNet we fall short of zero-barrier connections, although we do see a 67% decrease in barrier relative to naïve interpolation. As we demonstrate in Section 5.3, we can achieve zero-barrier LMC on CIFAR-10 with large ResNet models. Therefore, we hypothesize that the presence of LMC depends on the model having sufficient capacity (esp. width) to capture the complexity of the input data distribution, and that ImageNet results could be improved by expanding model width.

STE matching, the most expensive method, produces the best solutions. Somewhat surprising, however, is that the gap between STE and the other two methods is relatively small. In particular, it is remarkable how well Algorithm 1 performs without access to the input data at all. We found that weight matching offered a compelling balance between computational cost and performance: It runs in mere seconds (on modern hardware) and produces high-quality solutions.

5.2 Onset of Mode Connectivity

Given the results of Section 5.1, it may be tempting to conclude that the entirety of weight space contains only a single basin modulo permutation symmetries. However, we found that linear mode connectivity is an emergent property of training, and we were unable to uncover it early in training. We explore the emergence of LMC in Figure 3. Concurrent to our work, Benzing et al. (2022) showed that LMC at initialization is possible using a permutation found at the end of training.

[3] We also experimented with spherical linear interpolation ("slerp") and found it to perform slightly better than linear interpolation; however, the difference was not sufficiently significant to warrant diverging from the pre-existing literature.

Note that the final inlaid interpolation plot in Figure 3(right) demonstrates an important shortcoming of the loss barrier metric, i.e., the interpolation includes points with lower loss than either of the two models. However, the loss barrier is still positive due to non-negativity, as mentioned in Section 2.

5.3 Effect of Model Width

Conventional wisdom maintains that wider architectures are easier to optimize (Jacot et al., 2018; Lee et al., 2019). We now investigate whether they are also easier to linearly mode connect. We train VGG-16 (Simonyan & Zisserman, 2015) and ResNet20 (He et al., 2016) architectures of varying widths on the CIFAR-10 dataset. Results are presented in Figure 4.[4]

A clear relationship emerges between model width and linear mode connectivity, as measured by the loss barrier between solutions. Although $1\times$-sized models did not seem to exhibit linear mode connectivity, we found that larger width models decreased loss barriers all the way to zero. In Figure 4(right), we show what is to our knowledge the premiere demonstration of zero-barrier linear mode connectivity between two large ResNet models trained on a non-trivial dataset.

We highlight that relatively thin models do not seem to obey linear mode connectivity yet still exhibit similarities in training dynamics. This suggests that either our permutation selection methods are failing to find satisfactory permutations on thinner models or that some form of invariance other than permutation symmetries must be at play in the thin model regime.

5.4 Model Patching, Split Data Training, and Improved Calibration

Inspired by work on finetuning (Wortsman et al., 2022a), model patching (Singh & Jaggi, 2020; Raffel, 2021), and federated learning (McMahan et al., 2017; Konečný et al., 2016a;b), we study whether it is possible to synergistically merge the weights of two models trained on disjoint datasets. Consider, for example, an organization with multiple (possibly biased) datasets separated for regulatory (e.g., GDPR) or privacy (e.g., on-device data) considerations. Models can be trained on each dataset individually, but training in aggregate is not feasible. Can we combine separately trained models so that the merged model performs well on the entirety of the data?

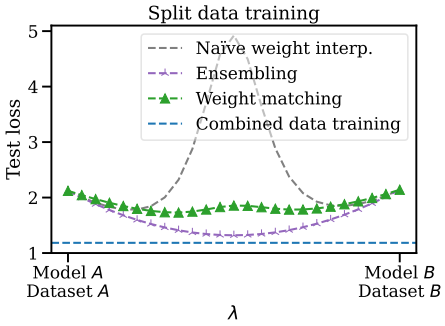

Figure 5: **Models trained on disjoint datasets can be merged constructively.** Algorithm 1 makes it possible for two ResNet models trained on disjoint, biased subsets of CIFAR-100 to be merged in weight space such that their combination outperforms both input models in terms of test loss on the combined dataset.

To address this question, we split the CIFAR-100 dataset (Krizhevsky, 2009) into two disjoint subsets: dataset A, containing 20% examples labelled 0-49 and 80% labelled 50-99, and dataset B, vice versa. ResNet20 models A and B were trained on their corresponding datasets. Privacy requirements mandate that we utilize a data-agnostic algorithm like Algorithm 1. Figure 5 shows the result of merging the two models with weight matching. For comparison, we benchmark naïve weight interpolation, ensembling of the model logits, and full-data training.

As expected, merging separately trained models did not match the performance of an omniscient model trained on the full dataset or an ensemble of the two models with twice the number of effective weights. On the other hand, we did manage to merge the two models in weight space, achieving an interpolated model that outperforms both input models in terms of test loss while using half the memory and compute required for ensembling. Furthermore, the merged model's probability estimates are better calibrated than either of the input models as demonstrated in Figure 11. Accuracy results are presented in Figure 10. Algorithm 1 also vastly outperformed naïve interpolation, the status quo for model combination in federated learning and distributed training.

[4]Unfortunately, $8\times$ width VGG-16 training was unattainable since it exhausted GPU memory on available hardware at the time of writing.

6 Related Work

(Linear) mode connectivity. Freeman & Bruna (2017) introduced the concept of mode connectivity, i.e., that SGD solutions in the loss landscape are connected by constant-loss curves in weight space. Further explorations were undertaken in Garipov et al. (2018); Draxler et al. (2018), among others. Tatro et al. (2020) explored non-linear mode connectivity modulo permutation symmetries. Frankle et al. (2020) demonstrated a connection between *linear* mode connectivity and the lottery ticket hypothesis. Juneja et al. (2022) demonstrated that LMC does not always hold, even when fine-tuning. Hecht-Nielsen (1990); Chen et al. (1993) noted the existence of permutation symmetries, and Brea et al. (2019) implicated them as a source of saddle points in the loss landscape. Recently, the prescient work of Entezari et al. (2021) conjectured that SGD solutions could be linear mode connected modulo permutation symmetries and offered a battery of experiments buttressing this claim. Unlike previous works on LMC we accomplish zero-barrier paths between two independently-trained models with an algorithm that runs on the order of seconds.

Loss landscapes and training dynamics. Li et al. (2016); Yosinski et al. (2014) investigated whether independently trained networks learn similar features, and to what extent they transfer. Jiang et al. (2021) argued that independently trained networks meaningfully differ in the features they learn in certain scenarios. Zhang et al. (2019) studied the relative importance of layers. Benton et al. (2021) argued that SGD solutions form a connected volume of low loss. Pittorino et al. (2022) proposed a toroidal topology of solutions and a set of algorithms for symmetry removal. On the theoretical front, Kawaguchi (2016) proved that deep linear networks contain no local minima. Boursier et al. (2022); Chizat & Bach (2018); Mei et al. (2018) characterized the training dynamics of one-hidden layer networks, proving that they converge to zero loss. Godfrey et al. (2022); Simsek et al. (2021) investigated the algebraic structure of symmetries in neural networks and how this structure manifests in loss landscape geometry.

Federated learning and model merging. McMahan et al. (2017); Konečný et al. (2016a;b) introduced the concept of "federated learning," i.e., learning split across across multiple devices and datasets. Wang et al. (2020) proposed an exciting federated learning method in which model averaging is done after permuting units. Unlike this work, they merged smaller "child" models into a larger "main" model, and did so with a layer-wise algorithm that does not support residual connections or normalization layers. Raffel (2021); Matena & Raffel (2021); Sung et al. (2021) conceptualized the study of "model patching," i.e., the idea that models should be easy to modify and submit changes to. Ilharco et al. (2022) investigated model patching for the fine-tuning of open-vocabulary vision models. Ashmore & Gashler (2015) first explored the use of matching algorithms for the alignment of network units. Singh & Jaggi (2020) proposed merging models by soft-aligning associations weights, inspired by optimal transport. Liu et al. (2022a); Uriot & Izzo (2020) further explored merging models taking possible permutations into account. Wortsman et al. (2022a) demonstrated state-of-the-art ImageNet performance by averaging the weights of many fine-tuned models.

7 Discussion and Future Work

We explore the role of permutation symmetries in the linear mode connectivity of SGD solutions. We introduce three novel algorithms to canonicalize independent neural network weights in order to make the loss landscape between them as flat as possible. In contrast to prior work, we linearly mode connect large ResNet models with no barrier in seconds to minutes. Despite presenting successes across multiple architectures and datasets, linear mode connectivity between thin models remains elusive. Therefore, we conjecture that permutation symmetries are a necessary piece, though not a complete picture, of the fundamental invariances at play in neural network training dynamics. In particular, we hypothesize that linear, possibly non-permutation, relationships connect the layer-wise activations between models trained by SGD. In the infinite width limit, there exist satisfactory linear relationships that are also permutations.

An expanded theory and empirical exploration of other invariances – such as cross-layer scaling or general linear relationships between activations – presents an intriguing avenue for future work. Ultimately, we anticipate that a lucid understanding of loss landscape geometry will not only advance the theory of deep learning but will also promote the development of better optimization, federated learning, and ensembling techniques.

ETHICS STATEMENT

Merging models raises interesting ethical and technical questions about the resulting models. Do they inherit the same biases as their input models? Are rare examples forgotten when merging? Is it possible to gerrymander a subset of the dataset by splitting its elements across many shards?

Deployment of any form of model merging ought to be paired with thorough auditing of the resulting model, investigating in particular whether the merged model is representative of the entirety of the data distribution.

REPRODUCIBILITY STATEMENT

Our code is open sourced at `https://github.com/REDACTED/REDACTED`. Our experimental logs and downloadable model checkpoints are fully open source at `https://wandb.ai/REDACTED/REDACTED`.

REFERENCES

Samuel K. Ainsworth, Nicholas J. Foti, Adrian K. C. Lee, and Emily B. Fox. oi-vae: Output interpretable vaes for nonlinear group factor analysis. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 119–128. PMLR, 2018. URL http://proceedings.mlr.press/v80/ainsworth18a.html.

Samuel K. Ainsworth, Kendall Lowrey, John Thickstun, Zaïd Harchaoui, and Siddhartha S. Srinivasa. Faster policy learning with continuous-time gradients. In Ali Jadbabaie, John Lygeros, George J. Pappas, Pablo A. Parrilo, Benjamin Recht, Claire J. Tomlin, and Melanie N. Zeilinger (eds.), *Proceedings of the 3rd Annual Conference on Learning for Dynamics and Control, L4DC 2021, 7-8 June 2021, Virtual Event, Switzerland*, volume 144 of *Proceedings of Machine Learning Research*, pp. 1054–1067. PMLR, 2021. URL http://proceedings.mlr.press/v144/ainsworth21a.html.

Anonymous. Linear mode connectivity of deep neural networks via permutation invariance and renormalization. In *Submitted to The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gU5sJ6ZggcX. under review.

Stephen C. Ashmore and Michael S. Gashler. A method for finding similarity between multi-layer perceptrons by forward bipartite alignment. In *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-17, 2015*, pp. 1–7. IEEE, 2015. doi: 10.1109/IJCNN.2015.7280769. URL https://doi.org/10.1109/IJCNN.2015.7280769.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013. URL http://arxiv.org/abs/1308.3432.

Gregory W. Benton, Wesley J. Maddox, Sanae Lotfi, and Andrew Gordon Wilson. Loss surface simplexes for mode connecting volumes and fast ensembling. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 769–779. PMLR, 2021. URL http://proceedings.mlr.press/v139/benton21a.html.

Frederik Benzing, Simon Schug, Robert Meier, Johannes von Oswald, Yassir Akram, Nicolas Zucchet, Laurence Aitchison, and Angelika Steger. Random initialisations performing above chance and how to find them, 2022. URL https://arxiv.org/abs/2209.07509.

D.P. Bertsekas. *Network Optimization: Continuous and Discrete Methods*. Athena scientific optimization and computation series. Athena Scientific, 1998. ISBN 9788865290279. URL https://books.google.com/books?id=afYYAQAAIAAJ.

Christopher M. Bishop. *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer, 2007. ISBN 9780387310732. URL https://www.worldcat.org/oclc/71008143.

Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. Fast differentiable sorting and ranking. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 950–959. PMLR, 2020. URL http://proceedings.mlr.press/v119/blondel20a.html.

Etienne Boursier, Loucas Pillaud-Vivien, and Nicolas Flammarion. Gradient flow dynamics of shallow relu networks for square loss and orthogonal inputs. *CoRR*, abs/2206.00939, 2022. doi: 10.48550/arXiv.2206.00939. URL https://doi.org/10.48550/arXiv.2206.00939.

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. *CoRR*, abs/1907.02911, 2019. URL http://arxiv.org/abs/1907.02911.

E. Cela. *The Quadratic Assignment Problem: Theory and Algorithms*. Combinatorial Optimization. Springer US, 2013. ISBN 9781475727876. URL https://books.google.com/books?id=20QGCAAAQBAJ.

An Mei Chen, Haw-minn Lu, and Robert Hecht-Nielsen. On the geometry of feedforward neural network error surfaces. *Neural Comput.*, 5(6):910–927, 1993. doi: 10.1162/neco.1993.5.6.910. URL https://doi.org/10.1162/neco.1993.5.6.910.

Lénaïc Chizat and Francis R. Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 3040–3050, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/a1afc58c6ca9540d057299ec3016d726-Abstract.html.

Matthieu Courbariaux and Yoshua Bengio. Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1. *CoRR*, abs/1602.02830, 2016. URL http://arxiv.org/abs/1602.02830.

David Frederic Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Trans. Aerosp. Electron. Syst.*, 52(4):1679–1696, 2016. doi: 10.1109/TAES.2016.140952. URL https://doi.org/10.1109/TAES.2016.140952.

Marco Cuturi, Olivier Teboul, and Jean-Philippe Vert. Differentiable ranks and sorting using optimal transport. *CoRR*, abs/1905.11885, 2019. URL http://arxiv.org/abs/1905.11885.

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 Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009. doi: 10.1109/CVPR.2009.5206848. URL https://doi.org/10.1109/CVPR.2009.5206848.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred A. Hamprecht. Essentially no barriers in neural network energy landscape. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1308–1317. PMLR, 2018. URL http://proceedings.mlr.press/v80/draxler18a.html.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *CoRR*, abs/2110.06296, 2021. URL https://arxiv.org/abs/2110.06296.

Fajwel Fogel, Rodolphe Jenatton, Francis R. Bach, and Alexandre d'Aspremont. Convex relaxations for permutation problems. *SIAM J. Matrix Anal. Appl.*, 36(4):1465–1488, 2015. doi: 10.1137/130947362. URL https://doi.org/10.1137/130947362.

Jonathan Frankle. Revisiting "qualitatively characterizing neural network optimization problems". *CoRR*, abs/2012.06898, 2020. URL https://arxiv.org/abs/2012.06898.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3259–3269. PMLR, 2020. URL http://proceedings.mlr.press/v119/frankle20a.html.

C. Daniel Freeman and Joan Bruna. Topology and geometry of half-rectified network optimization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Bk0FWVcgx.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P. Vetrov, and Andrew Gordon Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 8803–8812, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/be3087e74e9100d4bc4c6268cdbe8456-Abstract.html.

Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. On the symmetries of deep learning models and their internal representations. *CoRR*, abs/2205.14258, 2022. doi: 10.48550/arXiv.2205.14258. URL https://doi.org/10.48550/arXiv.2205.14258.

Ian J. Goodfellow and Oriol Vinyals. Qualitatively characterizing neural network optimization problems. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6544.

Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic optimization of sorting networks via continuous relaxations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=H1eSS3CcKX.

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 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

Robert Hecht-Nielsen. On the algebraic structure of feedforward network weight spaces. 1990.

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. *CoRR*, abs/2208.05592, 2022. doi: 10.48550/arXiv.2208.05592. URL https://doi.org/10.48550/arXiv.2208.05592.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In Amir Globerson and Ricardo Silva (eds.), *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pp. 876–885. AUAI Press, 2018. URL http://auai.org/uai2018/proceedings/papers/313.pdf.

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In Samy Bengio, Hanna M. Wallach, Hugo

Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 8580–8589, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/5a4be1fa34e62bb8a6ec6b91d2462f5a-Abstract.html.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J. Zico Kolter. Assessing generalization of SGD via disagreement. *CoRR*, abs/2106.13799, 2021. URL https://arxiv.org/abs/2106.13799.

Roy Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987. doi: 10.1007/BF02278710. URL https://doi.org/10.1007/BF02278710.

Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. Linear connectivity reveals generalization strategies. *CoRR*, abs/2205.12411, 2022. doi: 10.48550/arXiv.2205.12411. URL https://doi.org/10.48550/arXiv.2205.12411.

Zhao Kang, Chong Peng, and Qiang Cheng. Top-n recommender system via matrix completion. In Dale Schuurmans and Michael P. Wellman (eds.), *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pp. 179–185. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11824.

Kenji Kawaguchi. Deep learning without poor local minima. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 586–594, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/f2fc990265c712c49d51a18a32b39f0c-Abstract.html.

Matthew Kelly. An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Rev.*, 59(4):849–904, 2017. doi: 10.1137/16M1062569. URL https://doi.org/10.1137/16M1062569.

Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *CoRR*, abs/1610.02527, 2016a. URL http://arxiv.org/abs/1610.02527.

Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016b. URL http://arxiv.org/abs/1610.05492.

Tjalling C. Koopmans and Martin Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25(1):53–76, 1957. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1907742.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Rohith Kuditipudi, Xiang Wang, Holden Lee, Yi Zhang, Zhiyuan Li, Wei Hu, Rong Ge, and Sanjeev Arora. Explaining landscape connectivity of low-cost solutions for multilayer nets. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 14574–14583, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/46a4378f835dc8040c8057beb6a2da52-Abstract.html.

Harold W. Kuhn. The hungarian method for the assignment problem. In Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey (eds.), *50 Years of Integer Programming 1958-2008 - From the Early Years to the State-of-the-Art*, pp. 29–47. Springer, 2010. doi: 10.1007/978-3-540-68279-0_2. URL https://doi.org/10.1007/978-3-540-68279-0_2.

Aditya Kusupati, Matthew Wallingford, Vivek Ramanujan, Raghav Somani, Jae Sung Park, Krishna Pillutla, Prateek Jain, Sham M. Kakade, and Ali Farhadi. LLC: accurate, multi-purpose learnt low-dimensional binary codes. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 23900–23913, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/c88d8d0a6097754525e02c2246d8d27f-Abstract.html.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791. URL https://doi.org/10.1109/5.726791.

Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 8570–8581, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/0d1a9651497a38d8b1c3871c84528bd4-Abstract.html.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John E. Hopcroft. Convergent learning: Do different neural networks learn the same representations? In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.07543.

Chang Liu, Chenfei Lou, Runzhong Wang, Alan Yuhan Xi, Li Shen, and Junchi Yan. Deep neural network fusion via graph matching with applications to model ensemble and federated learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 13857–13869. PMLR, 2022a. URL https://proceedings.mlr.press/v162/liu22k.html.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 11966–11976. IEEE, 2022b. doi: 10.1109/CVPR52688.2022.01167. URL https://doi.org/10.1109/CVPR52688.2022.01167.

Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

James Lucas, Juhan Bae, Michael R. Zhang, Stanislav Fort, Richard S. Zemel, and Roger B. Grosse. On monotonic linear interpolation of neural network parameters. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7168–7179. PMLR, 2021. URL http://proceedings.mlr.press/v139/lucas21a.html.

Wesley J. Maddox, Pavel Izmailov, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 13132–13143, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/118921efba23fc329e6560b27861f0c2-Abstract.html.

Konstantin Makarychev, Rajsekar Manokaran, and Maxim Sviridenko. Maximum quadratic assignment problem: Reduction from maximum label cover and lp-based approximation algorithm. *ACM Trans. Algorithms*, 10(4):18:1–18:18, 2014. doi: 10.1145/2629672. URL https://doi.org/10.1145/2629672.

Michael Matena and Colin Raffel. Merging models with fisher-weighted averaging. *CoRR*, abs/2111.09832, 2021. URL https://arxiv.org/abs/2111.09832.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In Aarti Singh and Xiaojin (Jerry) Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017. URL http://proceedings.mlr.press/v54/mcmahan17a.html.

Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layers neural networks. *CoRR*, abs/1804.06561, 2018. URL http://arxiv.org/abs/1804.06561.

Gonzalo E. Mena, David Belanger, Scott W. Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=Byt3oJ-0W.

Thao Nguyen, Maithra Raghu, and Simon Kornblith. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=KJNcAkY8tY4.

Felix Petersen, Christian Borgelt, Hilde Kuehne, and Oliver Deussen. Differentiable sorting networks for scalable sorting and ranking supervision. *CoRR*, abs/2105.04019, 2021. URL https://arxiv.org/abs/2105.04019.

Felix Petersen, Christian Borgelt, Hilde Kuehne, and Oliver Deussen. Monotonic differentiable sorting networks. *CoRR*, abs/2203.09630, 2022. doi: 10.48550/arXiv.2203.09630. URL https://doi.org/10.48550/arXiv.2203.09630.

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. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17759–17781. PMLR, 2022. URL https://proceedings.mlr.press/v162/pittorino22a.html.

Sebastian Prillo and Julian Eisenschlos. Softsort: A continuous relaxation for the argsort operator. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7793–7802. PMLR, 2020. URL http://proceedings.mlr.press/v119/prillo20a.html.

Colin Raffel. A call to build models like we build open-source software. https://colinraffel.com/blog/a-call-to-build-models-like-we-build-open-source-software.html, 2021. Accessed: 2022-06-17.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pp. 525–542. Springer, 2016. doi: 10.1007/978-3-319-46493-0_32. URL https://doi.org/10.1007/978-3-319-46493-0_32.

Sartaj Sahni and Teofilo F. Gonzalez. P-complete approximation problems. *J. ACM*, 23(3):555–565, 1976. doi: 10.1145/321958.321975. URL https://doi.org/10.1145/321958.321975.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.1556.

Berfin Simsek, 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. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9722–9732. PMLR, 2021. URL http://proceedings.mlr.press/v139/simsek21a.html.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/fb2697869f56484404c8ceee2985b01d-Abstract.html.

Yi-Lin Sung, Varun Nair, and Colin Raffel. Training neural networks with fixed sparse masks. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 24193–24205, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/cb2653f548f8709598e8b5156738cc51-Abstract.html.

N. Joseph Tatro, Pin-Yu Chen, Payel Das, Igor Melnyk, Prasanna Sattigeri, and Rongjie Lai. Optimizing mode connectivity via neuron alignment. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/aecad42329922dfc97eee948606e1f8e-Abstract.html.

Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016. URL http://arxiv.org/abs/1607.08022.

Thomas Uriot and Dario Izzo. Safe crossover of neural networks through neuron alignment. In Carlos Artemio Coello Coello (ed.), *GECCO '20: Genetic and Evolutionary Computation Conference, Cancún Mexico, July 8-12, 2020*, pp. 435–443. ACM, 2020. doi: 10.1145/3377930.3390197. URL https://doi.org/10.1145/3377930.3390197.

Tiffany Vlaar and Jonathan Frankle. What can linear interpolation of neural network loss landscapes tell us? *CoRR*, abs/2106.16004, 2021. URL https://arxiv.org/abs/2106.16004.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=uXl3bZLkr3c.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris S. Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BkluqlSFDS.

Mitchell Wortsman, Maxwell Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. Learning neural network subspaces. In Marina Meila and Tong Zhang (eds.), *Proceedings of*

the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, volume 139 of *Proceedings of Machine Learning Research*, pp. 11217–11227. PMLR, 2021. URL http://proceedings.mlr.press/v139/wortsman21a.html.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak 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. *Proceedings of the 39th International Conference on Machine Learning, ICML 2022*, 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.

Yuxin Wu and Kaiming He. Group normalization. *Int. J. Comput. Vis.*, 128(3):742–755, 2020. doi: 10.1007/s11263-019-01198-w. URL https://doi.org/10.1007/s11263-019-01198-w.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 3320–3328, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/375c71349b295fbe2dcdca9206f20a06-Abstract.html.

Chiyuan Zhang, Samy Bengio, and Yoram Singer. Are all layers created equal? *CoRR*, abs/1902.01996, 2019. URL http://arxiv.org/abs/1902.01996.

A APPENDIX

A.1 KNOWN FAILURE MODES

We emphasize that none of the techniques presented in this paper are silver bullets. Here we list the failure cases that the authors are presently aware of,

1. Models of insufficient width
2. Models in the initial stages of training
3. VGGs on MNIST
4. MNIST MLPs trained with SGD and too low of a learning rate, or Adam and too high of a learning rate
5. ConvNeXt architectures (Liu et al., 2022b), which have surprisingly few permutation symmetries due to extensive use of depth-wise convolutions

Furthermore, we believe other failure modes certainly exist but have yet to be discovered.

We are excited by the prospect of future work investigating these failure modes and improving our understanding of when and why model merging modulo permutation symmetries is feasible.

A.2 EXTENDED RELATED WORK

Non-linear mode connectivity. A flourishing set of literature exists studying non-linear mode connectivity, including but not limited to Garipov et al. (2018); Draxler et al. (2018); Kuditipudi et al. (2019). This insightful line of work is inspirational to our own, however we take a strictly linear approach to mode connectivity as in Frankle et al. (2020); Juneja et al. (2022). Restricting ourselves to linear trajectories comes with the advantage of having direct implications for a single-basin theory. However, it comes at the cost of a more challenging, discrete optimization problem. In particular, we found that – in contrast to non-linear mode connectivity – linear mode connectivity becomes drastically harder with smaller width models. Note additionally that most pre-existing mode connectivity work does not account for permutation symmetries of weight space, a linchpin element of our work. A notable exception to this trend can be found in Tatro et al. (2020).

To summarize: Freeman & Bruna (2017) introduced the notion of mode connectivity and proved that loss landscapes for single-hidden layer ReLU models contain only a single basin in the infinite width limit. Garipov et al. (2018) and Draxler et al. (2018) concurrently demonstrated that simple zero-barrier curves can be learned to connect the optima in weight space, thus reshaping our understanding of practical loss landscape geometries. Kuditipudi et al. (2019) proposes a theoretical explanation for the mode connectivity phenomenon. Benton et al. (2021) extends mode connectivity from one-dimensional paths to entire manifolds of low-loss, and show that these manifolds can be leveraged for state-of-the-art Bayesian ensembling of models.

Relationship with Tatro et al. (2020). The impact of permutation symmetries on the non-linear mode connectivity of models is considered in Tatro et al. (2020). In particular, they independently propose an algorithm more-or-less equivalent to Section 3.1 but use it in conjunction with learned non-linear mode connecting curves. In contrast, we show that linear mode connectivity can be achieved without the need for learning non-linear paths between the aligned weights. Our derivation of Section 3.1 from the principle of least-squares regression is novel, to the best of our knowledge.

Relationship with Singh & Jaggi (2020). Singh & Jaggi (2020) studies model merging with "soft matchings" between units from the perspective of optimal transport. We emphasize the following commonalities/differences with their work:

- We focus on linear mode connectivity modulo permutation symmetries and its implications for a single-basin theory. On the other hand, Singh & Jaggi (2020) emphasizes "soft" (ie., non-permutation) matching of units via optimal transport.
- The activation matching method of Singh & Jaggi (2020) reduces to that of Section 3.1 when the optimal transport regularization term is set to zero and the unit "importance" values are set to uniform across all units on all layers.

- Our weight matching and straight-through estimator methods solve for an alignment across all layers jointly. In contrast, Singh & Jaggi (2020) executes greedy, single-pass matching looking only at weight information from the immediately previous layer when selecting permutations. Singh & Jaggi (2020) mentions jointly solving for alignments as an avenue for future work.
- The "wts" method of Singh & Jaggi (2020) is not run on models including bias terms, skip connections, or normalization layers. In contrast, our Algorithm 1 works with models of nearly arbitrary architecture.
- Our weight matching method (Algorithm 1) outperforms the "wts" method of Singh & Jaggi (2020). See Appendix A.7 for more information.
- Singh & Jaggi (2020) introduces a method for merging multiple models simultaneously, but only demonstrates results on at most 8 models at a time and performs continued training after merging. In contrast, our Algorithm 3 has been shown to work with as many as 32 models at a time and does not require continued training after merging. Our analysis of the calibration of the resulting merged models has no parallel in Singh & Jaggi (2020).

Relationship with Entezari et al. (2021). Entezari et al. (2021) introduces the single-basin conjecture and provides the following evidence towards it:

- Entezari et al. (2021) provides a statistical test which fails to detect a difference in barrier statistics between independently trained models and random permutations of the same model (Fig. 5 of Entezari et al. (2021)). Our work provides stronger support for the conjecture in that we give methods that can directly "unscramble" these permutations, proving that LMC can be found (Figures 4 and 5).

  Entezari et al. (2021)'s experimental protocol does not provide evidence for linear mode connectivity. Rather, their experimental results suggest that barriers resulting from independent training look like the barriers resulting from random permutations. But this result is consistent with a world in which all solutions have barriers between them – both between members of the same permutation equivalence class and between solutions in separate equivalence classes! In other words, there may still exist multiple equivalence classes of solutions. In contrast, we provide concrete evidence for a single-basin theory by developing algorithms that directly place permuted solutions into the same basin (Figure 1).
- Although Entezari et al. (2021)'s conjecture is an important intellectual ancestor to our work, their demonstration of linear mode connectivity is limited to a single hidden-layer MLP on MNIST (Fig. 2 of Entezari et al. (2021)). However, this result for single hidden-layer MLP models is preceded by Freeman & Bruna (2017); Uriot & Izzo (2020). On the other hand, we focus on larger models and datasets that are more closely aligned with models used in practice at the time of writing.
- Entezari et al. (2021) proposes a simulated annealing algorithm that yields modest reductions in barrier between independently trained models, yet requires multiple days to run.

  On the other hand, our weight matching algorithm (Algorithm 1) completely removes barriers between models for more challenging models and datasets (Figures 2 and 5), and runs in seconds (Appendix A.5). Moreover, our weight matching method does not require access to the training data, enabling its potential application in domains like federated learning and distributed training.

In short, the work of Entezari et al. (2021) first proposed the "single-basin" conjecture. Our work is the first (to the best of our knowledge) to demonstrate that linear mode connectivity can be achieved between large models independently trained on challenging datasets.

Differentiating through permutations. Akin to differentiable permutation learning, many prior works have studied differentiable sorting (Grover et al., 2019; Prillo & Eisenschlos, 2020; Cuturi et al., 2019; Petersen et al., 2022; 2021; Mena et al., 2018). Blondel et al. (2020) studied differentiable sorting and ranking with asymptotics that correspond to their non-differentiable versions. Fogel et al. (2015) explored recovering the linear orderings of items based on pairwise information, another form of permutation optimization. Bengio et al. (2013) introduced the straight-through estimator for differentiating through discrete projections that we utilize in Section 3.3.

A.3 EXPERIMENTAL DETAILS

A.3.1 MULTI-LAYER PERCEPTRON MODELS

In these experiments we utilized networks with 3 hidden layers of 512 units each. ReLU activations were used between layers and no normalization was performed. Optimization was done with Adam and a learning rate of $1e-3$.

A.3.2 VGG-16 AND RESNET MODELS ON CIFAR DATASETS

We utilized the VGG-16 architecture of Simonyan & Zisserman (2015) with the exception that we used LayerNorm normalization in place of BatchNorm. Similarly we used the ResNet20 architecture of He et al. (2016) but with LayerNorms in place of BatchNorms.

The following data augmentation was performed during training

- Random resizes of the image between 0.8×-1.2×
- Random 32×32 pixel crops
- Random horizontal flips
- Random rotations between $\pm 30°$

Optimization was done with SGD with momentum (momentum set to 0.9). A weight decay regularization term of $5e-4$ was applied. A single cosine decay schedule with linear warm-up was used. Learning rates were initialized at $1e-6$ and linearly increased to $1e-1$ over the span of an epoch. After that point a single cosine decay schedule (Loshchilov & Hutter, 2017) was used for the remainder of training.

A.3.3 RESNET50 MODELS ON IMAGENET-1K

In this experiment we utilized pre-trained ResNet50 model available for download online, and one trained ourselves. These were standard ResNet50 models, including the use of BatchNorm. In line with prior work (Izmailov et al., 2018; Wortsman et al., 2021; Maddox et al., 2019; Wang et al., 2021; Anonymous, 2023), we recalculate BatchNorm statistics after performing weight interpolation.

A note of chronology. Our work was concurrent to that of Anonymous (2023). However after our initial publication, the recalculation of BatchNorm statistics was suggested to us by the authors of Anonymous (2023). Therefore, the ImageNet experiments presented in Section 5.1 ought to be viewed as following the work of Anonymous (2023).

A.4 THE RELATIONSHIP BETWEEN PERMUTATION MATCHING AND NORMALIZATION LAYERS

In this section we discuss the impact that different types of common normalization layers can have on the feasibility of model merging.

- **BatchNorm** (Ioffe & Szegedy, 2015) generally breaks after interpolating between weights due to the so-called "variance collapse" problem (Anonymous, 2023). Therefore, following Anonymous (2023), we recommend the recalculation of batch statistics after merging models (Izmailov et al., 2018; Wortsman et al., 2021; Maddox et al., 2019; Wang et al., 2021).
- **LayerNorm** (Ba et al., 2016) is invariant to permutations of units and we found that architectures with LayerNorm can be merged without issue.
- **InstanceNorm** (Ulyanov et al., 2016) also places no restrictions on unit order, and in principle does not present any issues, although we have not run any experiments with it.
- **GroupNorm** (Wu & He, 2020) relies on unit indexes to organize units into groups, and therefore is not invariant to permutations of units. In principle, permutation alignment methods would not work on architectures with GroupNorm, though we have not tested this.

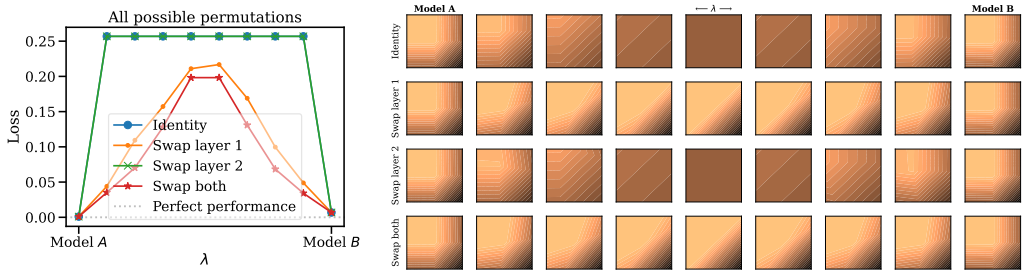

Figure 6: **A counterexample.** There exist models such that no possible permutation of weights allows for linear mode connectivity. *Left:* performance of all possible linear interpolations between the two models. *Right:* A visualization of the prediction functions $f(\boldsymbol{x})$ through each linear sweep. Each row corresponds to one of the four possible permutations, and each column corresponds to a value of λ, the linear interpolant. The existence of such cases suggests that linear mode connectivity is an artifact of SGD.

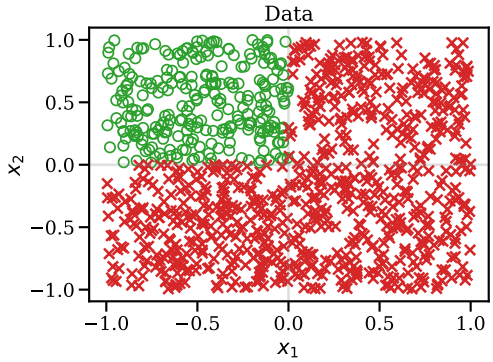

Figure 7: The counterexample classification problem data.

A.5 Additional Information on Algorithm 1

On currently available hardware (p3.2xlarge AWS instance with an NVIDIA V100 GPU), we observed the following timing results with Algorithm 1,

1. MLP (3 layers, 512 units each): 3 seconds
2. ResNet50 ($1\times$ width): 33 seconds
3. ResNet20 ($32\times$ width): 194 seconds

In addition, we tested the ability of Algorithm 1 to recover a known, randomly selected permutation. In a handful of experiments we found that Algorithm 1 was able to exactly recover the known, random permutation in just 3-4 of passes over the layers.

A.6 Counterexample Details

Consider a simple 2-dimensional classification task. Our data points are drawn $\boldsymbol{x} \sim \text{Uniform}([-1,1]^2)$ and $y = \mathbf{1}_{x_1<0 \text{ and } x_2>0}$. Figure 7 provides a visualization of a sample of such data.

Method	Test accuracy (↑)	Run-time (↓)
OT-Fusion	85.98%	2.86s
Weight matching (ours)	**86.57%**	**0.64s**

Table 2: **VGG11/CIFAR-10 performance relative to Singh & Jaggi (2020).** We found that our weight matching method outperforms the "wts" method of Singh & Jaggi (2020) in both implementation speed and model performance when reproducing one of their experiments. Our implementation is 4.5× faster, and produces a solution with better model performance.

We utilize an MLP architecture consisting of two hidden layers, with two units each, and ReLU nonlinearities. Consider two weight assignments that both achieve a perfect fit to the data:

$$f_A(\boldsymbol{x}) = \begin{bmatrix}-1 & -1\end{bmatrix} \sigma\left(\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix} \sigma\left(\begin{bmatrix}-1 & 0\\ 0 & -1\end{bmatrix} \boldsymbol{x} + \begin{bmatrix}1\\ 0\end{bmatrix}\right) + \begin{bmatrix}1\\ 0\end{bmatrix}\right) \tag{4}$$

$$f_B(\boldsymbol{x}) = \begin{bmatrix}-1 & -1\end{bmatrix} \sigma\left(\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix} \sigma\left(\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \boldsymbol{x} + \begin{bmatrix}0\\ 1\end{bmatrix}\right) + \begin{bmatrix}0\\ 1\end{bmatrix}\right). \tag{5}$$

We predict a positive label when $f(\boldsymbol{x}) \geq 0$ and a negative label otherwise. Intuitively, these networks are organized such that each layer makes a classification whether $\boldsymbol{x}_1 < 0$ or $\boldsymbol{x}_2 > 0$. In model A, the first layer tests whether $\boldsymbol{x}_2 > 0$, and the second layer tests whether $\boldsymbol{x}_1 < 0$, whereas in model B the order is reversed. With a bit of algebra, it is possible to see that both f_A and f_B achieve perfect performance.

A.7 On the Failures of Greedy Uni-directional Matching

In contrast to prior work (Singh & Jaggi, 2020; Wang et al., 2020), we eschew greedy uni-directional, single-pass matching between models. Instead we derive a weight matching algorithm from a principled, yet computationally infeasible optimization problem. In contrast to prior work, our method can be viewed as "bi-directional": it selects unit associations based on weights in all relevant layers, not just in the immediately previous layer. In this section, we describe benefits of our holistic approach, including an example problem showing the failure modes of greedy uni-directional matching. We find that matching across all layers simultaneously allows our weight matching algorithm (Algorithm 1) to exploit units' relationships with downstream weights in a way that greedy uni-directional matching cannot.

Concretely, greedy uni-directional matching begins at the first layer and computes a matching, $\boldsymbol{P}_1$, considering only $\boldsymbol{W}_1^{(A)}, \boldsymbol{W}_1^{(B)}$. After matching, $\boldsymbol{P}_1$ is applied throughout the remainder of the network. Then, we proceed through the layers in order repeating this process.

A.7.1 Experimental Comparison

In order to further evaluate or performance relative to prior work, and Singh & Jaggi (2020) in particular, we explored experimental comparisons with VGG11 models trained on CIFAR-10 and on ResNet50 models trained on ImageNet.

VGG11 models trained on CIFAR-10. Following an exact reproduction of experiment from Table 1 of Singh & Jaggi (2020), we merged the model weights released along with their paper. We present results in Table 2. We found that our weight matching method outperforms the "wts" method of Singh & Jaggi (2020) in both implementation speed and model performance when reproducing one of their experiments.

ResNet50 models trained on ImageNet. We applied OT-Fusion to the ImageNet experiment that we consider in Section 5.1. We present results in Figure 8. We found the OT-Fusion method resulted in models with 1.38% top-1 accuracy on ImageNet, only marginally improving over naïve averaging. On the other hand, we achieve 51.01%.

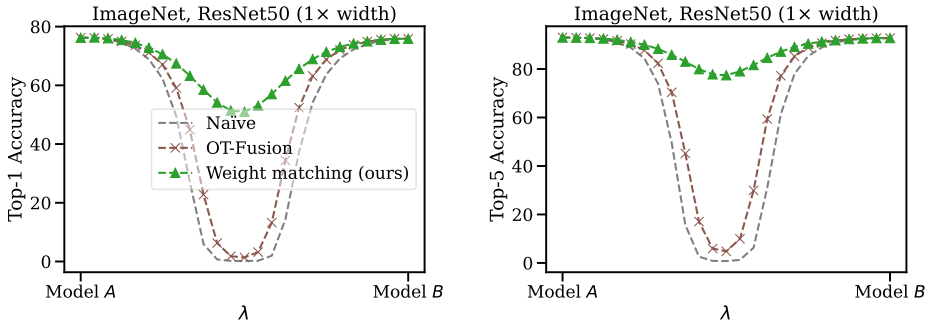

Figure 8: **ResNet50/ImageNet performance relative to Singh & Jaggi (2020).** We found the OT-Fusion method resulted in models with 1.38% top-1 accuracy on ImageNet, only marginally improving over naïve averaging. On the other hand, we achieve 51.01%.

A.7.2 An Example Failure Case

Consider two networks, A and B, with the objective that they capture the identity function $f(x) = x$,

$$f_{\Theta_A}(x) = \begin{bmatrix}1 & 0\end{bmatrix} \begin{bmatrix}1 & 0 \\ 0 & \epsilon\end{bmatrix} \begin{bmatrix}1 \\ 1+\epsilon\end{bmatrix} x \tag{6}$$

$$f_{\Theta_B}(x) = \begin{bmatrix}0 & 1\end{bmatrix} \begin{bmatrix}0 & 0 \\ 0 & 1\end{bmatrix} \begin{bmatrix}1 \\ 1+\epsilon\end{bmatrix} x \tag{7}$$

where $\epsilon > 0$ is some negligible constant. It can be seen that these reduce to $f_{\Theta_A}(x) = x$ and $f_{\Theta_B}(x) = (1+\epsilon)x$.

When aligning these models there are two possible opportunities for permutation, $\pi = \{\boldsymbol{P}_1, \boldsymbol{P}_2\}$. The permuted model B then has the form

$$f_{\pi(\Theta_B)}(x) = \left(\begin{bmatrix}0 & 1\end{bmatrix} \boldsymbol{P}_2^\top\right) \left(\boldsymbol{P}_2 \begin{bmatrix}0 & 0 \\ 0 & 1\end{bmatrix} \boldsymbol{P}_1^\top\right) \left(\boldsymbol{P}_1 \begin{bmatrix}1 \\ 1+\epsilon\end{bmatrix}\right) x \tag{8}$$

Now, aligning with greedy uni-directional matching will result in the alignment $\pi_{gud} = \{\boldsymbol{P}_1 = \boldsymbol{I}, \boldsymbol{P}_2 = \boldsymbol{I}\}$. On the other hand, our weight matching method (Algorithm 1) results in $\pi_{wm} = \left\{\boldsymbol{P}_1 = \begin{bmatrix}0 & 1 \\ 1 & 0\end{bmatrix}, \boldsymbol{P}_2 = \begin{bmatrix}0 & 1 \\ 1 & 0\end{bmatrix}\right\}$, regardless of the algorithm's execution order.

Interpolating these matched models at $\lambda = 0.5$, we have

$$f_{\frac{1}{2}(\Theta_A + \pi_{gud}(\Theta_B))}(x) = \begin{bmatrix}0.5 & 0.5\end{bmatrix} \begin{bmatrix}0.5 & 0 \\ 0 & (1+\epsilon)/2\end{bmatrix} \begin{bmatrix}1 \\ 1+\epsilon\end{bmatrix} x \qquad = (0.5 + O(\epsilon))x \tag{9}$$

$$f_{\frac{1}{2}(\Theta_A + \pi_{wm}(\Theta_B))}(x) = \begin{bmatrix}1 & 0\end{bmatrix} \begin{bmatrix}1 & 0 \\ 0 & \epsilon/2\end{bmatrix} \begin{bmatrix}1+\epsilon/2 \\ 1+\epsilon/2\end{bmatrix} x \qquad = (1+\epsilon/2)x \tag{10}$$

Here we can see that the greedy uni-directional matching (Equation (9)) results in a merged model that fails to represent the input, identity-function models. On the other hand, our weight matching algorithm (Algorithm 1, Equation (10)) produces a merged model that accurately reflects both of the input models, and even improves performance over the B model. [5]

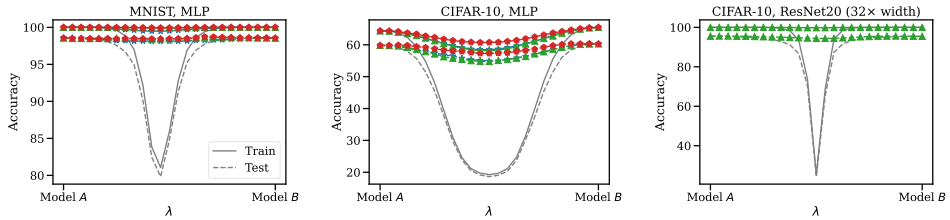

Figure 9: Top-1 accuracy results for the MNIST and CIFAR-10 models of Figure 2.

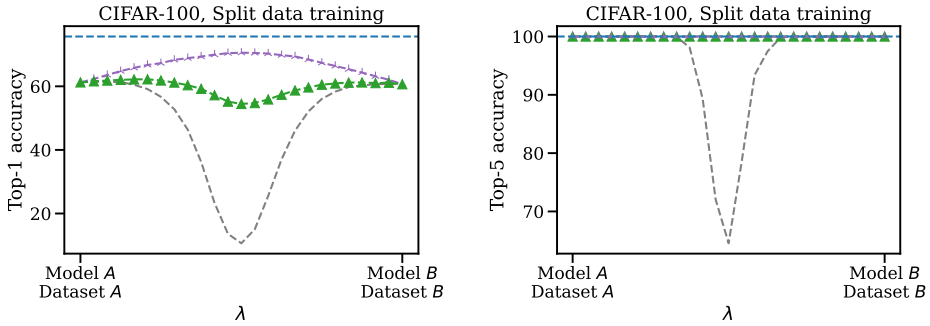

Figure 10: Accuracy results for the CIFAR-100 split data experiment.

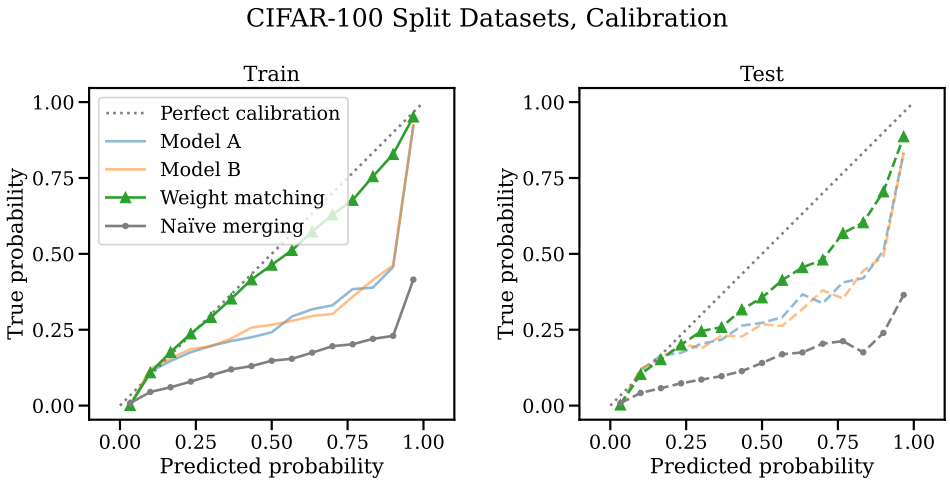

Figure 11: **Merging CIFAR-100 split data models results in superior probability calibration.** Although our merged model is not competitive in terms of top-1 accuracy in the CIFAR-100 split data experiment, we find that it has far better calibrated probability estimates than either of the input models.

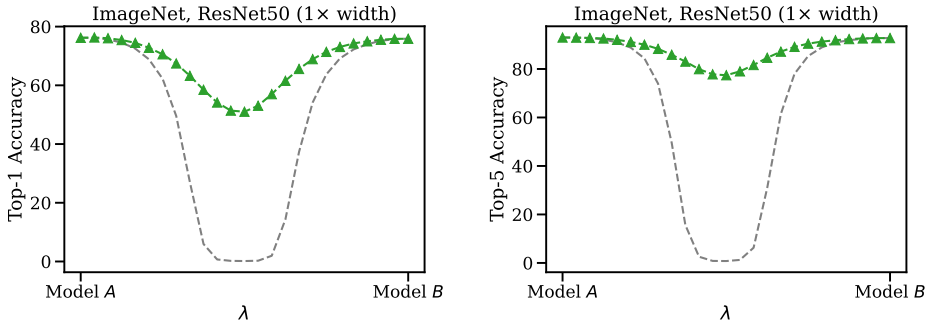

Figure 12: Accuracy results for merged ResNet50 (1× width) models on ImageNet.

A.8 Auxiliary Plots

A.9 Straight-through Estimator Details

See Algorithm 2 for a complete description of the straight-through estimator algorithm.

Algorithm 2: Straight-through estimator training

Given: Model weights Θ_A, Θ_B, and a learning rate η.

Result: A permutation π of Θ_B such that $\mathcal{L}(\frac{1}{2}(\Theta_A + \pi(\Theta_B)))$ is approximately minimized.

Initialize: $\tilde{\Theta}_B \leftarrow \Theta_A$

repeat

- $\pi(\Theta_B) \leftarrow \text{proj}(\tilde{\Theta}_B)$ using Algorithm 1.
- Evaluate the loss of the midpoint, $\mathcal{L}(\frac{1}{2}(\Theta_A + \pi(\Theta_B)))$.
- Evaluate the gradient, $\nabla\mathcal{L}$, using $\tilde{\Theta}_B$ in place of $\pi(\Theta_B)$ in the backwards pass.
- Update parameters, $\tilde{\Theta}_B \leftarrow \tilde{\Theta}_B - \eta\nabla\mathcal{L}$.

until *convergence*

A.10 Merging Many Models

We propose Algorithm 3 to merge the weights of more than two models at a time.

Following an argument similar to Lemma 2, it can be seen that Algorithm 3 terminates.

In our limited testing, we found that this algorithm converges quickly to solutions that extrapolate better than individual models and results in a merged model with better probability estimate calibration than any of the input models. For example, we present the results of this algorithm on MLPs trained on MNIST in Table A.10.

In addition, we found that merging multiple models helps to calibrate the resulting model predictions. We present this effect in Figure 13.

A.11 Failed Idea: A Method for Steepest Descent

Imagine standing in weight space at Θ_A and trying to decide in which immediate direction to move in order to approach a Θ_B-equivalent point. There are many, many possible permutations of Θ_B –

[5]We note that this example can be extended to the more conventional presentation, including non-linear activation functions, by inserting ReLU activations in each layer and considering x in the positive domain $\mathbb{R}^+$. The positive domain restriction can also be lifted by instead considering the task of absolute value estimation and adding layers $\sigma \circ \begin{bmatrix}1 & 1\end{bmatrix} \circ \sigma \circ \begin{bmatrix}1\\-1\end{bmatrix}$ to the beginning of the network.

Algorithm 3: MERGEMANY

Given: Model weights $\Theta_1, \ldots, \Theta_N$

Result: A merged set of parameters $\tilde{\Theta}$.

repeat
  for $i \in$ RANDOMPERMUTATION$(1, \ldots, N)$ **do**
    $\Theta' \leftarrow \frac{1}{N-1} \sum_{j \in \{1,\ldots,N\} \setminus \{i\}} \Theta_j$
    $\pi \leftarrow$ PERMUTATIONCOORDINATEDESCENT(Θ', Θ_i)
    $\Theta_i \leftarrow \pi(\Theta_i)$
  end
until *convergence*
return $\frac{1}{N} \sum_{j=1}^{N} \Theta_j$

	Train Loss	Train Acc.	Test Loss	Test Acc.
Seed 1	0.0000	1.0000	0.1153	0.9856
Seed 2	0.0000	1.0000	0.1531	0.9854
Seed 3	0.0000	1.0000	0.1229	0.9855
Seed 4	0.0000	1.0000	0.1108	0.9865
Seed 5	0.0000	1.0000	0.1443	0.9871
MERGEMANY	0.0141	0.9952	**0.0727**	0.9831

Table 3: **Merging multiple models decreases test loss by 43%.** We train five separate MLPs on MNIST. Using Algorithm 3 we merge all these models together simultaneously. This produces a model that appears to be better calibrated than any of the input models, with superior test loss performance. We are excited by potential applications of this methodology in federated learning and ensembling, esp. along the lines of "model soups" (Wortsman et al., 2022a).

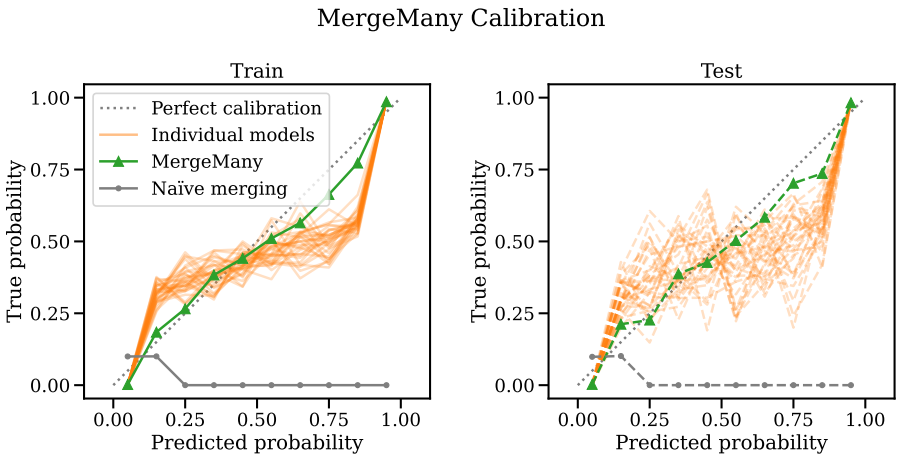

Figure 13: **Merging multiple models results in superior calibration.** Here we show the results of running Algorithm 3 on 32 MLP models trained on MNIST, with each model given access to a random 50% of the training dataset. The resulting merged model demonstrates substantively improved calibration of probability estimates on both the training and test datasets.

call them $\pi^{(1)}(\Theta_B), \pi^{(2)}(\Theta_B), \ldots$ – to aim for in the distance. Assuming that the loss landscape is in fact (quasi-)convex modulo these permutation symmetries, a natural choice would be to pick the $\pi^{(i)}(\Theta_B)$ that corresponds to the direction of steepest descent starting from Θ_A since we expect $\pi^{(i)}(\Theta_B)$ to lie in the same basin as Θ_A. In other words,

$$\min_{\pi} \left. \frac{d\,\mathcal{L}(\Theta_A + \lambda(\pi(\Theta_B) - \Theta_A))}{d\,\lambda} \right|_{\lambda=0} = \min_{\pi} \nabla\mathcal{L}(\Theta_A)^\top (\pi(\Theta_B) - \Theta_A) \tag{11}$$

$$= -\nabla\mathcal{L}(\Theta_A)^\top \Theta_A + \min_{\pi} \nabla\mathcal{L}(\Theta_A)^\top \pi(\Theta_B) \tag{12}$$

Now, we are tenuously in a favorable situation: $\nabla\mathcal{L}(\Theta_A)$ is straightforward to compute, and picking the best π reduces to a matching problem. In particular it is a SOBLAP matching problem of the same form as in Section 3.2. In addition, there is a fast, exact solution for the single intermediate layer case ($L = 2$).

In practice, we found that this method can certainly find directions of steepest descent, but that they are accompanied by high barriers in between the initial dip and $\pi(\Theta_B)$.

A.12 Proof of Lemma 1

To lighten notation we use $\langle \cdot, \cdot \rangle = \langle \cdot, \cdot \rangle_F$ in this section.

Lemma. *Given $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{d \times d}$,*

$$\min_{\boldsymbol{P}, \boldsymbol{Q} \text{ perm. matrices}} \langle \boldsymbol{P}\boldsymbol{A}\boldsymbol{Q}^\top, \boldsymbol{B} \rangle$$

is strongly NP-hard and has no PTAS.

Proof. We proceed by reduction from the quadratic assignment problem (QAP) (Koopmans & Beckmann, 1957; Cela, 2013). Consider a QAP,

$$\min_{\boldsymbol{P} \text{ perm. matrix}} \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{P}^\top, \boldsymbol{D} \rangle$$

for $\boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{d \times d}$.

Now, pick $\boldsymbol{A} = \boldsymbol{C} + \lambda \boldsymbol{I}$, $\boldsymbol{B} = \boldsymbol{D} - \lambda \boldsymbol{I}$. The we have,

$$\min_{\boldsymbol{P}, \boldsymbol{Q}} \langle \boldsymbol{P}(\boldsymbol{C} + \lambda \boldsymbol{I})\boldsymbol{Q}^\top, \boldsymbol{D} - \lambda \boldsymbol{I} \rangle = \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{Q}^\top + \lambda \boldsymbol{P}\boldsymbol{Q}^\top, \boldsymbol{D} - \lambda \boldsymbol{I} \rangle \tag{13}$$

$$= \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{Q}^\top, \boldsymbol{D} \rangle - \lambda \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{Q}^\top, \boldsymbol{I} \rangle + \lambda \langle \boldsymbol{P}\boldsymbol{Q}^\top, \boldsymbol{D} \rangle - \lambda^2 \langle \boldsymbol{P}\boldsymbol{Q}^\top, \boldsymbol{I} \rangle \tag{14}$$

$$= \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{Q}^\top, \boldsymbol{D} \rangle - \lambda \langle \boldsymbol{P}^\top \boldsymbol{Q}, \boldsymbol{C} \rangle + \lambda \langle \boldsymbol{P}\boldsymbol{Q}^\top, \boldsymbol{D} \rangle - \lambda^2 \operatorname{tr}(\boldsymbol{P}\boldsymbol{Q}^\top) \tag{15}$$

For sufficiently large λ, the $\operatorname{tr}(\boldsymbol{P}\boldsymbol{Q}^\top)$ term will dominate. Letting $\alpha = \max\left(\max_{i,j} |C_{i,j}|, \max_{i,j} |D_{i,j}|\right)$, we can bound the other terms,

$$-d^2\alpha^2 \leq \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{Q}^\top, \boldsymbol{D} \rangle \leq d^2\alpha^2 \tag{16}$$

$$-\lambda d\alpha \leq -\lambda \langle \boldsymbol{P}^\top \boldsymbol{Q}, \boldsymbol{C} \rangle \leq \lambda d\alpha \tag{17}$$

$$-\lambda d\alpha \leq \lambda \langle \boldsymbol{P}\boldsymbol{Q}^\top, \boldsymbol{D} \rangle \leq \lambda d\alpha \tag{18}$$

Now there are two classes of solutions: those where $\boldsymbol{P} = \boldsymbol{Q}$ and those where $\boldsymbol{P} \neq \boldsymbol{Q}$. We seek to make the best (lowest) possible $\boldsymbol{P} \neq \boldsymbol{Q}$ solution to have worse (higher) objective value than the worst (highest) $\boldsymbol{P} = \boldsymbol{Q}$ solution. When $\boldsymbol{P} = \boldsymbol{Q}$, the highest possible objective value is

$$d^2\alpha^2 + \lambda d\alpha + \lambda d\alpha - \lambda^2 d$$

and similarly, the lowest possible objective value when $\boldsymbol{P} \neq \boldsymbol{Q}$ is

$$-d^2\alpha^2 - \lambda d\alpha - \lambda d\alpha - \lambda^2 d + \lambda^2$$

where the final term is due to the fact that at least one entry of $\boldsymbol{P}\boldsymbol{Q}^\top$ must be 0. With some algebra, it can be seen that $\lambda > 5d\alpha$ is sufficient to guarantee that all $\boldsymbol{P} = \boldsymbol{Q}$ solutions are superior to all $\boldsymbol{P} \neq \boldsymbol{Q}$ solutions.

Now when $\boldsymbol{P} = \boldsymbol{Q}$, all frivolous terms reduce to constants and we are left with the QAP objective:

$$\begin{aligned}\min_{\boldsymbol{P}} \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{P}^\top, \boldsymbol{D}\rangle - \lambda\langle \boldsymbol{P}^\top\boldsymbol{P}, \boldsymbol{C}\rangle + \lambda\langle \boldsymbol{P}\boldsymbol{P}^\top, \boldsymbol{D}\rangle - \lambda^2 \operatorname{tr}(\boldsymbol{P}\boldsymbol{P}^\top) \\ = -\lambda \operatorname{tr}(\boldsymbol{C}) + \lambda \operatorname{tr}(\boldsymbol{D}) - \lambda^2 d + \min_{\boldsymbol{P}} \langle \boldsymbol{P}\boldsymbol{C}\boldsymbol{P}^\top, \boldsymbol{D}\rangle\end{aligned}$$

completing the reduction. QAP is known to be strongly NP-hard (Koopmans & Beckmann, 1957; Sahni & Gonzalez, 1976) and MaxQAP is known to not admit any PTAS (Makarychev et al., 2014), thus completing the proof. □

A.13 Proof of Lemma 2

Lemma. *Algorithm 1 terminates.*

Proof. We proceed by contradiction.

Consider a graph with each possible permutation $\pi_i = \{\boldsymbol{P}_1, \ldots, \boldsymbol{P}_{L-1}\}$ as a vertex and directed edges $\pi_i \to \pi_j$ if π_j can be reached from π_i with a single $\boldsymbol{P}_\ell$ update, as in Algorithm 1. (Ignore those updates that result in no change to $\boldsymbol{P}_\ell$ in order to avoid $\pi_i \to \pi_i$ cycles.) Let $\rho(\pi) = \text{vec}(\Theta_A) \cdot \text{vec}(\pi(\Theta_B))$ denote the utility of a particular π. Note that $\pi_i \to \pi_j$ implies $\rho(\pi_i) < \rho(\pi_j)$. There exist finitely many possible permutations π_i, meaning that a failure to terminate must involve a cycle in the graph $\pi_1 \to \cdots \to \pi_n \to \pi_1$. However ρ forms a total order on the vertices and therefore we have a contradiction. □