KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Machine Learning

Second cycle, 30 credits

On Linear Mode Connectivity up to Permutation of Hidden Neurons in Neural Networks

When does Weight Averaging work?

ADHITHYAN KALAIVANAN

Stockholm, Sweden, 2023

On Linear Mode Connectivity up to Permutation of Hidden Neurons in Neural Networks

When does Weight Averaging work?

ADHITHYAN KALAIVANAN

Master's Programme, Machine Learning, 120 credits
Date: September 28, 2023

Supervisor: Josephine Sullivan
Examiner: Hossein Azizpour
School of Electrical Engineering and Computer Science
Swedish title: Anslutning i linjärt läge upp till permutation av dolda neuroner i neurala nätverk
Swedish subtitle: När fungerar Viktmedelvärde?

Abstract

Neural networks trained using gradient-based optimization methods exhibit a surprising phenomenon known as mode connectivity, where two independently trained network weights are not isolated low loss minima in the parameter space. Instead, they can be connected by simple curves along which the loss remains low. In case of linear mode connectivity up to permutation, even linear interpolations of the trained weights incur low loss when networks that differ by permutation of their hidden neurons are considered equivalent. While some recent research suggest that this implies existence of a single near-convex loss basin to which the parameters converge, others have empirically shown distinct basins corresponding to different strategies to solve the task. In some settings, averaging multiple network weights naively, without explicitly accounting for permutation invariance still results in a network with improved generalization. In this thesis, linear mode connectivity among a set of neural networks independently trained on labelled datasets, both naively and upon reparameterization to account for permutation invariance is studied. Specifically, the effect of hidden layer width on the connectivity is empirically evaluated. The experiments are conducted on a two dimensional toy classification problem, and the insights are extended to deeper networks trained on handwritten digits and images. It is argued that accounting for permutation of hidden neurons either explicitly or implicitly is necessary for weight averaging to improve test performance. Furthermore, the results indicate that the training dynamics induced by the optimization plays a significant role, and large model width alone may not be a sufficient condition for linear model connectivity.

Keywords

Mode Connectivity, Representation Learning, Loss Landscape, Network Symmetry

Sammanfattning

Neurala nätverk som tränats med gradientbaserade optimeringsmetoder uppvisar ett överraskande fenomen som kallas modeconnectivity, där två oberoende tränade nätverksvikter inte är isolerade lågförlustminima i parameterutrymmet. Istället kan de kopplas samman med enkla kurvor längs vilka förlusten förblir låg. I händelse av linjär mode-anslutning upp till permutation medför även linjära interpolationer av de tränade vikterna låga förluster när nätverk som skiljer sig åt genom permutation av deras dolda neuroner anses vara likvärdiga. Medan en del nyare undersökningar tyder på att detta innebär att det finns en enda nära-konvex förlustbassäng till vilken parametrarna konvergerar, har andra empiriskt visat distinkta bassänger som motsvarar olika strategier för att lösa uppgiften. I vissa inställningar resulterar ett naivt medelvärde av flera nätverksvikter, utan att uttryckligen ta hänsyn till permutationsinvarians, fortfarande i ett nätverk med förbättrad generalisering. I den här avhandlingen studeras linjärmodsanslutningar mellan en uppsättning neurala nätverk som är oberoende tränade på märkta datamängder, både naivt och vid omparameterisering för att ta hänsyn till permutationsinvarians. Specifikt utvärderas effekten av dold lagerbredd på anslutningen empiriskt. Experimenten utförs på ett tvådimensionellt leksaksklassificeringsproblem, och insikterna utökas till djupare nätverk som tränas på handskrivna siffror och bilder. Det hävdas att redogörelse för permutation av dolda neuroner antingen explicit eller implicit är nödvändigt för viktgenomsnitt för att förbättra testprestanda. Dessutom indikerar resultaten att träningsdynamiken som induceras av optimeringen spelar en betydande roll, och enbart stor modellbredd kanske inte är ett tillräckligt villkor för linjär modellanslutning.

Nyckelord

Lägesanslutning, representationsinlärning, förlustlandskap, nätverkssymmetri

Acknowledgments

I would like to express my sincere appreciation and gratitude to my supervisor, Prof. Josephine Sullivan, for her invaluable guidance throughout the thesis period. Her expertise and support have been instrumental in shaping this work. I am also deeply thankful to Prof. Hossein Azizpour for his encouragement, which inspired me to pursue this research question. Finally, I am indebted to my friends, Daniel Richards and Aishwarya Ganesan, and family, who have enriched both my academic and personal life, and have been a constant source of motivation.

Stockholm, September 2023
Adhithyan Kalaivanan

Contents

List of Figures

List of Tables

Chapter 1

Introduction

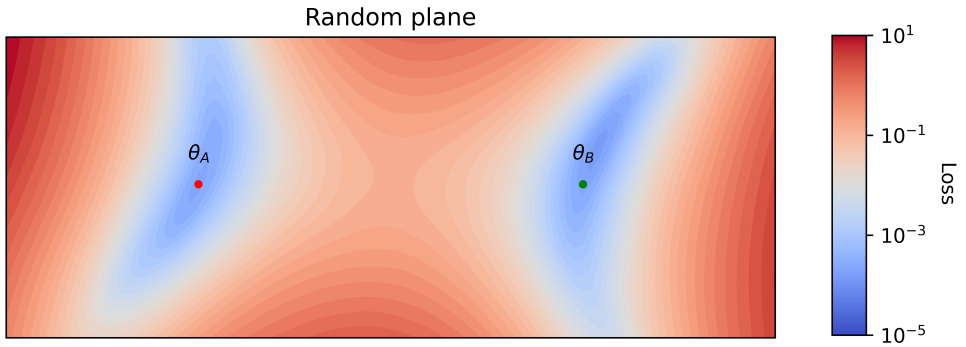
This chapter provides a brief introduction to the optimization landscape of neural networks, along with the main research questions and contributions of the thesis. After demarcating the scope of this work, it concludes with a general outline of the report.

1.1 Overview

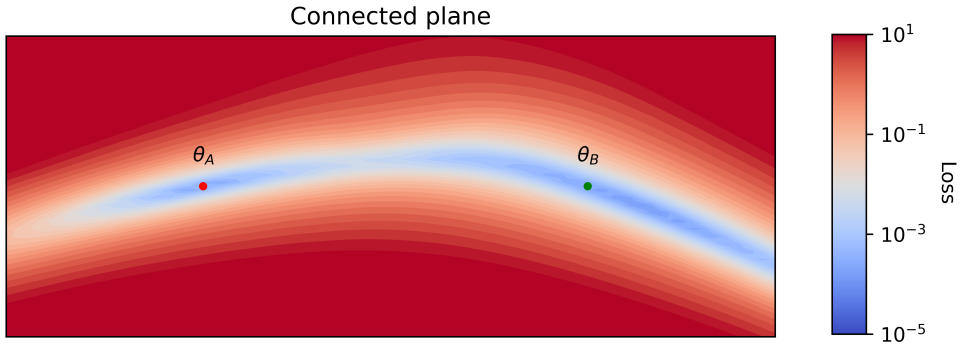
Neural networks continue to grow as a dominant paradigm of modelling both structured and unstructured data, yet their unreasonable efficacy remains unexplained. Despite often being overparameterized and trained to fully fit a set of noisy training samples, these models exhibit strong generalization performance that classical learning theory would not predict. Consequently, ongoing efforts are focused on understanding the generalization of neural networks from various perspectives. One approach involves studying the properties of their loss landscape.

Modern neural networks have millions of trainable parameters, resulting in a high-dimensional parameter space, also referred to as the weight space. The loss function that maps the model parameters to a real value measuring success on a task is often non-convex. Therefore, numerical methods involving local gradient-based optimization are used to estimate these parameters. This process has multiple sources of stochasticity, ranging from the choice of initial parameter values to the number of training samples used to estimate the gradient and the order of the training samples. Changing the random seed results in different network weights that are far apart in the weight space, yet they all share a high and near identical test accuracy. Probing the weights along a linear path between two networks shows that they are separated by a

large loss barrier, creating an image of these solutions being isolated pockets of low loss in the weight space, as depicted in Figure 1.1a. However, *mode connectivity* is a surprising phenomenon where solutions obtained through variants of gradient descent can be connected by simple curves in the weight space along which the loss remains low. This is entirely counterintuitive, as the predominant symmetry in neural networks is permutation invariance, where the neurons of a hidden layer can be permuted along with suitable rearrangement of their incoming and outgoing weights while preserving the function being computed exactly. Since the permutation group consists of discrete elements, this would typically inhibit any continuous transformation of a solution that could result in a different but equally good solution. Yet such continuous paths of low loss solutions are empirically shown to exist on the loss surface of neural networks, as illustrated in Figure 1.1b



(a) Visualization of the high dimensional test loss surface along a random plane which passes through two independently trained network weights. They are isolated from one another by a large loss barrier



(b) Visualization of the same loss surface but now through a plane on which the two network weights are found to be connected by a low-loss valley

Figure 1.1: Illustration of mode connectivity on the loss surface

1.2 Problem

A special case known as *linear mode connectivity* (LMC) is observed when accounting for the permutation invariance of sufficiently wide neural networks optimized through gradient based methods. Reparameterizing the networks by a suitable permutation of hidden neurons along with their weights results in a low loss along the linear path connecting them in the weight space. See figure 1.2 for an illustration. This has led to speculations that the loss landscape, accounting for permutation, contains one near-convex basin to which the solutions found through conventional training methods converge. Exploring this may help better understand the geometry of the loss landscape and shed insights on neural network generalization.

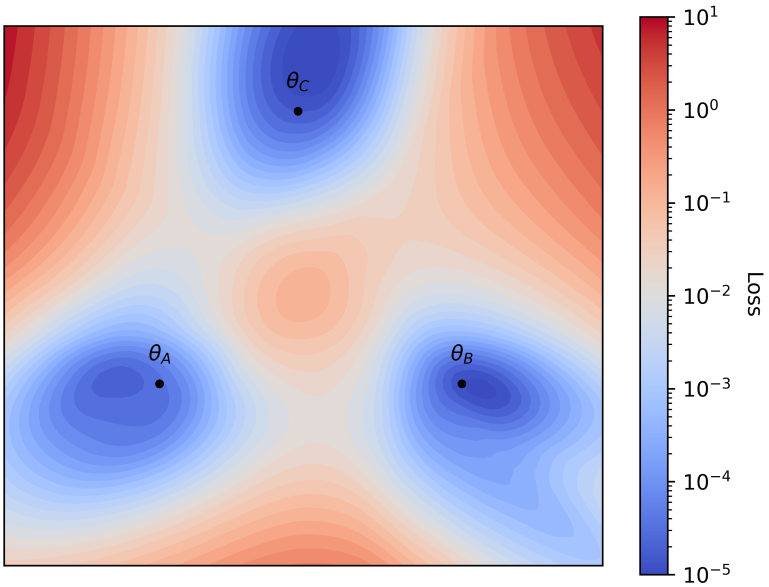
1.2.1 Research questions

In this work, we analyze the influence of reparameterization for permutation alignment on the loss neighbourhood of network weights. Specifically, we are interested in how the loss varies along the line connecting different trained network weights before and after reparameterization. These are presented in Section 4.1 and Section 4.2 respectively. By quantifying the increase in loss along the path by ϵ, as defined in Section 2.1.2, we investigate if such reparameterization is necessary and sufficient for the average network to incur low loss, in Section 5.1 and Section 5.2. Lastly, we relate linear mode connectivity up to permutation and the weight averaging methods used in practice, by explaining the success of Stochastic Weight Averaging (SWA), a popular method where no explicit permutation alignment is done. This is discussed in Section 4.3.

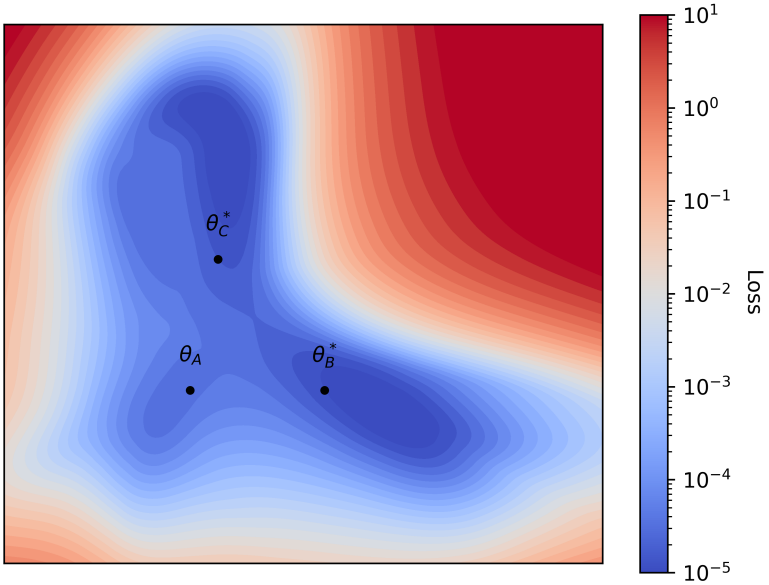
1.2.2 Contributions

Through experiments done on fully connected networks trained using the two-Moons, MNIST and CIFAR-10 datasets and reparameterizing them for permutation alignment, the following empirical results are presented by the thesis.

- The loss incurred by naive averaging of network weights decreases as the hidden layer width increases, but this is only due to the feature redundancy in the hidden neurons.
- In narrow networks, the loss incurred on averaging the weights after



(a) Visualization of the test loss surface along a plane passing through three network weights independently trained on the Moons data. They appear to be isolated minima



(b) Network weights θ_B and θ_C are reparameterized to be permutation aligned to θ_A. Now the visualization of the test loss surface passing through the three new network weights reveal the solutions to be linearly mode connected

Figure 1.2: Illustration of linear mode connectivity up to permutation

reparameterization for permutation still remains high, indicating that the network weights converge to diverse loss basins up to permutation.

- In wide networks, the loss incurred on averaging the weights after reparameterization significantly reduces, but this is dependent on the choice of optimizer, indicating that such reparameterization alone is not sufficient to observe LMC up to permutation.
- Weight averaging methods improve the test accuracy of the final average network as the sample networks are implicitly permutation aligned and linearly mode connected from sharing parts of their training trajectory.

1.3 Delimitations

Understanding linear mode connectivity in neural networks is a broad task, and this work presents a limited exploration. Various network architectures are developed to suit the natural structure of different data modalities, but here the experiments are limited to simple fully connected networks trained on synthetic and vision datasets. A thorough investigation on how well the conclusions extend to other architectures and domains remains open. Additionally, only networks trained in a supervised manner are considered, and the results may not necessarily extend to unsupervised learning. Finally, reparameterization for permutation alignment uses heuristic algorithms that do not guarantee an optimal solution. Thus, the failure to find a low-loss connecting path between networks does not disprove the existence of such a path, only that the current state-of-the-art methods are unable to obtain them. In fact, making conclusive remarks requires a brute force search through all possible arrangements of the hidden neurons, which is computationally prohibitive for large models.

1.4 Structure of the thesis

Having provided a short overview of the research area, problem statements and main contributions, the rest of the thesis is structured as below:

- Chapter 2 introduces the preliminaries and related literature.
- Chapter 3 details the data, training procedure and the reparameterization method to account for permutation invariance in networks.

- Chapter 4 motivates the experiments and presents their results.
- Chapter 5 presents further analysis on some of the intriguing results.
- Chapter 6 provides the summary and final remarks.

Chapter 2

Background

This chapter presents an introduction to neural networks and how they are trained in supervised settings. Following this, the literature related to mode connectivity, the special case of linear mode connectivity, and studies on network symmetries are discussed.

2.1 Preliminaries

Suppose that the input space $\mathcal{X}$ and output space $\mathcal{Y}$ are such that a probability distribution P on their product space $\mathcal{X} \times \mathcal{Y}$ describes the data distribution. Say the goal is to create a classifier $f : \mathcal{X} \to \mathcal{Y}$ such that $f(X)$ gives a likely output. However, we do not have access to this underlying data distribution P, instead only samples $S = \{(x_i, y_i)\}_{i=1}^n$ which are assumed to be independent and identically distributed (i.i.d.) as P. This set S (or a subset) is referred to as the training data. The task of learning a classifier is often done through either empirical or regularized risk minimization. A sample wise loss function $\ell : \mathcal{X} \times \mathcal{Y} \times \mathcal{F} \to \mathbb{R}_{\geq 0}$ is defined as the error between an estimate $f(X)$ and the corresponding target output Y. Typical choices include the squared loss and cross-entropy loss. The population risk is defined as $\mathcal{L} := \mathbb{E}(\ell(f(X), Y))$ and is approximated by the empirical risk $\mathcal{L}_S := \frac{1}{n} \sum_{i=1}^n (\ell(f(x_i), y_i))$. Empirical risk minimization involves choosing a parameterized function $f_\theta \in \mathcal{F}$ and estimating θ which minimizes the empirical risk.

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^n (\ell(f_\theta(x_i), y_i))$$

In regularized risk minimization, an additional term $\Omega : \mathcal{F} \to \mathbb{R}_{\geq 0}$ that measures the classifier's complexity is also included in the objective to be minimized.

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (\ell(f_\theta(x_i), y_i)) + \Omega(f_\theta)$$

Consider input space $\mathbb{R}^{d_0}$ and output space $\mathbb{R}^{d_k}$, where $k > 1$. A fully connected neural network is defined as follows.

Definition 1. A fully connected network is a parameterised function $f_\theta : \mathbb{R}^{d_0} \to \mathbb{R}^{d_k}$ of the form,

$$\begin{aligned} f_\theta &:= l_k \circ \cdots \circ l_1 \\ l_i(x) &:= \sigma_i(W_i x + b_i) \quad \text{for} \quad 1 \leq i \leq k \end{aligned}$$

where $\sigma_i : \mathbb{R} \to \mathbb{R}$ is the activation function which is applied element-wise, and $l_i : \mathbb{R}^{d_{i-1}} \to \mathbb{R}^{d_i}$ denotes the operation at hidden layer i. The trainable parameters $\theta := \{W_i, b_i | 1 \leq i \leq k\}$ constitutes each layer's weights and biases. They are often estimated by variants of gradient descent, of which a simple example is stochastic gradient descent (SGD):

$$\theta^{t+1} = \theta^t - \alpha_t \nabla_{\theta^t} (\ell(f_{\theta^t}(x_i), y_i))$$

where α_t is the learning rate, the chosen step size for the update at step t. Starting from a random initialization θ^0, the update is repeated until convergence and the gradient at each step is evaluated at (x_i, y_i) which are uniformly sampled from the training set S.

Suppose a task is specified by the data distribution P and the choice of loss function ℓ. Given a sample set S, let θ_A and θ_B be the parameters of two neural networks trained by independent runs of stochastic gradient descent. Note that these networks may have different initialization and may have been trained on different ordering of the input data. For convenience, we use θ to refer to the function f_θ and $\mathcal{L}_S(\theta)$ to denote the empirical risk of the model evaluated on the sample set S. For a fixed network architecture, sample data, and loss function, mode connectivity can be defined, similar to that of Kuditipudi et al. [Kud+19]

Definition 2 (Mode connectivity). Two solutions θ_A and θ_B are said to be ϵ-

mode connected if for $\epsilon \geq 0$ there exists a continuous curve $\omega : [0, 1] \to \Theta$ such that,

$$\omega(0) = \theta_A, \quad \omega(1) = \theta_B,$$
$$\mathcal{L}_S(\omega(t)) \leq \max(\mathcal{L}_S(\theta_A), \mathcal{L}_S(\theta_B)) + \epsilon \quad \forall \quad 0 \leq t \leq 1$$

For ϵ close to zero, the solutions are simply said to be mode connected. In many practical settings, neural network solutions obtained through variants of gradient descent are shown to exhibit mode connectivity.

2.1.1 Model symmetries and invariance

A mapping $\mu : \Theta \to \mathcal{F}$ such that $\mu(\theta) := f_\theta$ is not injective, meaning that $f_{\theta_A} = f_{\theta_B}$ does not imply that $\theta_A = \theta_B$. This is because the same function can be parameterized in different ways due to the invariance induced by the network. A few examples that illustrate this are described below.

Permutation. In fully connected networks, rearranging the neurons within a hidden layer while adjusting the incoming and outgoing weights accordingly leaves the output unchanged, but results in a different parameterization. This also holds in other architectures such as convolutional neural networks, where the feature maps at a hidden layer can be treated as units that can be permuted in the channel dimension, but not along their height or width. Denoting the permutation of each hidden layer as $P_i \in \mathcal{P}_{d_i}$, a output preserving transform $\pi : \Theta \to \Theta$ for fully connected networks is given by,

$$\pi(\theta) = \{P_i W_i P_{i-1}^T, P_i b_i \mid P_0 = \mathbb{I}_{d_0}, P_k = \mathbb{I}_{d_k}, 1 \leq i \leq k\}$$

Scaling. For some activation functions, the network may be invariant to continuous transformations. Consider a layer l_i in a fully connected network with ReLU activation. The incoming and outgoing weights can be positively scaled without changing the output. Suppose $D_i \in \mathbb{R}^{d_i \times d_i}$ is a diagonal matrix with only positive elements. The scaling transform $\rho_\lambda : \Theta \to \Theta$ that preserves the output is given by,

$$\rho_\lambda(\theta) = \{D_i W_i D_{i-1}^{-1}, D_i b_i \mid D_0 = \mathbb{I}_{d_0}, D_k = \mathbb{I}_{d_k}, 1 \leq i \leq k\}$$

A detailed analysis on the symmetries of 2-layer ReLU networks is given by Petzka, Trimmel, and Sminchisescu [PTS20] and the invariance induced by other activation functions are discussed by Godfrey et al. [God+22].

These are not the only output preserving transformations. In some weight

configurations, such as when all incoming weights of a hidden neuron are zero, the outgoing weights can be arbitrarily modified without changing the output. Similarly, if all outgoing weights of a neuron are zero, the inverse holds true. If two neurons in a hidden layer perform the same computation, their outgoing weights can be adjusted while maintaining their total contributions. For ReLU activation, this can be further relaxed as shown by Petzka, Trimmel, and Sminchisescu [PTS20]. Consider two neurons in a ReLU layer with incoming weights $\mathbf{w}_i, \mathbf{w}_j$, outgoing weights $\mathbf{v}_i, \mathbf{v}_j$, such that $\lambda(\mathbf{w}_i \cdot x + b_i) = \mathbf{w}_j \cdot x + b_j$ for some $\lambda > 0$, then changing the outgoing weights as shown below does not affect the network output for all c.

$$\mathbf{v}_i \leftarrow \mathbf{v}_i - c$$
$$\mathbf{v}_j \leftarrow \mathbf{v}_j + \frac{c}{\lambda}$$

2.1.2 Linear mode connectivity

Linear interpolation between the parameters of two trained networks often results in a network with low test accuracy. For simplicity, consider two 2-layer *linear* networks, denoted as θ_A and θ_B, with no bias terms or activation functions. Define a new network $\theta_\alpha := \alpha\theta_A + (1-\alpha)\theta_B$, where α is between 0 and 1. In the functional space, we have

$$f_{\theta_A} = W_2^A W_1^A$$
$$f_{\theta_B} = W_2^B W_1^B$$

and therefore, f_{θ_α} can be expressed as:

$$\alpha^2 W_2^A W_1^A + \alpha(1-\alpha)(W_2^A W_1^B + W_2^B W_1^A) + (1-\alpha)^2(W_2^B W_1^B)$$

While the first and last terms resemble the trained networks, the others are combinations of the original weights. Consequently, we can generally expect such restitched networks to perform poorly.

Definition 3 (Linear mode connectivity). Two solutions θ_A and θ_B are said to be linearly ϵ-mode connected if for some $\epsilon \geq 0$, along the line $\omega : [0,1] \to \Theta$ such that,

$$\omega(\alpha) := \alpha\theta_A + (1-\alpha)\theta_B$$

the following inequality holds for all $\alpha \in [0, 1]$,

$$\mathcal{L}_S(\omega(\alpha)) \leq \max(\mathcal{L}_S(\theta_A), \mathcal{L}_S(\theta_B)) + \epsilon$$

Linear mode connectivity has been demonstrated between networks in the fine-tuning regime [Fra+20], *i.e.,* solutions that are fine-tuned from a common pretrained model. More recently, it has also been observed between networks with completely independent training trajectories when the permutation invariance of their hidden layers is taken into account [AHS23] [Pit+22].

Definition 4 (Linear mode connectivity up to permutation). Two solutions θ_A and θ_B are said to be linearly mode connected up to permutation if there exists some $\theta_A^* \in [\theta_A]$ and $\theta_B^* \in [\theta_B]$ such that θ_A^* and θ_B^* are linearly mode connected. Here, $[\theta]$ represents an equivalence class with the relation $\theta_i \sim \theta_j$ if and only if there exists a permutation transform π such that $\theta_j = \pi(\theta_i)$, as described in subsection 2.1.1.

Definition 5 (Loss barrier). The loss barrier $B : \Theta \times \Theta \to \mathbb{R}$ between two networks θ_A and θ_B for $\alpha \in [0, 1]$ is defined as:

$$B(\theta_A, \theta_B) = \max_{\alpha} \left[\mathcal{L}_S(\alpha\theta_A + (1-\alpha)\theta_B) - (\alpha\mathcal{L}_S(\theta_A) + (1-\alpha)\mathcal{L}_S(\theta_B))\right]$$

Instead of denoting the degree of linear mode connectedness by ϵ, it is also common to use the loss barrier. Observe that zero loss barrier is a stricter condition than linear 0-mode connectivity.

2.2 Related work

This section discusses the current literature on mode connectivity with a focus on the specific case of linear mode connectivity.

2.2.1 Mode connectivity

In 2-layer ReLU networks, before any empirical evidence, Freeman and Bruna [FB22] predicted that the parameters will be asymptotically mode connected as the layer width increases. Later, Garipov et al. [Gar+18] demonstrated this by minimizing the expected loss along a parametric curve that connects two independent network weights. Concurrently, similar results were shown by Draxler et al. [Dra+18], where the connecting curves were found using

the Nudged Elastic Band method [JMJ98]. To explain mode connectivity in these realistic deep networks, Kuditipudi et al. [Kud+19] used dropout stability, which requires that predictions do not change by dropping half the neurons at every hidden layer. This needs to hold for just a fixed dropout pattern, not any arbitrary one. They proved that all dropout stable solutions are connected, with the connecting curve being piece-wise linear and its number of segments growing linearly with network depth. Shevchenko and Mondelli [SM20] also showed that SGD solutions can be connected by a piece-wise linear path, and they bounded the increase in loss along the path, which vanishes asymptotically with the total number of neurons. Going beyond simple mode connecting curves, Benton et al. [Ben+21] showed that independent solutions can be connected via volumes of low-loss simplicial complexes. They did this by sequentially adding a new connecting vertex that minimizes expected loss within the volume, while a regularizer term in the objective kept the simplicial complex volume from collapsing to zero. Lastly, instead of connecting independent solutions, Simsek et al. [Sim+21] proved that adding just one extra neuron than required at every hidden layer is sufficient to connect networks that vary just by permutation of their hidden neurons.

2.2.2 Linear mode connectivity

While previous research had explored piece-wise linear paths connecting independent solutions, Frankle et al. [Fra+20] discovered a linear, low-loss path between solutions that share part of their training trajectory. From the convex hull formed by these connected solutions, Yunis et al. [Yun+23] contend that it is high dimensional, and uniformly drawn samples from it have lower loss than the nearest vertex with high probability. They also conclude that the hull comprises of functionally diverse solutions, as the endpoints do not agree on their output predictions. Extending beyond solutions which share their training trajectory, Entezari et al. [Ent+22] hypothesized that independent gradient descent based solutions are linearly mode connected up to permutation. Various works that followed, including [AHS23], [Pit+22], and [Peñ+22], have since demonstrated this using a permutation transform to be applied on one of two given networks, thereby empirically observing linear mode connectivity. The permutation estimation methods vary from matching hidden neuron activation, or the weights themselves as a linear assignment problem, to implicit Sinkhorn differentiation [Eis+22], where the permutation matrix is relaxed to a doubly stochastic matrix so that gradient-

based optimization of any chosen loss function can be used. In addition, Benzing et al. [Ben+22] demonstrated that fully connected networks can be linearly mode connected up to permutation even upon initialization, and that the average network performed better than to random. These findings suggest that there exists one near-convex basin in the loss surface that gradient descent based solutions converge to.

But to the contrary, other recent works indicate the existence of multiple basins in the loss surface, even after accounting for permutation invariance. Ilharco et al. [Ilh+22] finetuned a large model on various tasks and consider the difference between finetuned and pretrained weights as the task vector. They demonstrated that these task vectors obey simple arithmetic operations, where adding two task vectors to a pretrained model produces a multitask model. Moreover, negating a task vector reduces the pretrained model's performance on that specific task while preserving its other capabilities. Similarly, Juneja et al. [Jun+22] found that independent runs of finetuning a large language model result in solutions that are linearly mode connected within those that generalize and those that do not, but not across these two groups. It is likely that in the fine-tuning regime, these models can be considered permutationally aligned as they share much of their early layer weights. Finally, Lubana et al. [Lub+22] provided empirical support for the relationship between linear mode connectivity up to permutation and the mechanistic similarity of networks in arriving at a prediction.

2.2.3 Model similarity

With the general theme of comparing different trained networks and their neighbourhood in the loss surface, this subsection briefly discusses some literature on quantifying the similarity in neural networks. Li et al. [Li+15] investigated *convergent learning* in networks, which refers to the extent to which different neural networks converge to the same representation. By analyzing the correlation of activation between networks, they showed that some features are consistently learned across them. Following works have built upon this to quantify representational similarity using measures such as SVCCA [Rag+17], projection weighted CCA [MRB18], CKA [Kor+19], and deconfounded CKA [Cui+22]. More recently, Moschella et al. [Mos+22] argued that representations learned by different architectures are a quasi-isometric transformation of one another. While Somepalli et al. [Som+22] used decision boundary visualizations to argue that different model architectures exhibit visually distinguishable boundaries.

2.3 Summary

While many works have reasoned about the prevalence of mode connectivity both empirically and theoretically, the assumptions made still leave room for a better understanding of why mode connectivity occurs in practical networks. With the notations and definitions established and the research question contextualized within the current state of research, the following section delves into the methodology.

Chapter 3

Methods

This chapter provides details of the data, models, and training procedures, along with the rationale for these selections. It is followed by the procedure used to account for permutation invariance. Lastly, it concludes with a summary of the model zoo on which the experiments are preformed in the subsequent chapters.

3.1 Data

Most of the experiments are performed with networks trained on one of two labeled datasets. To visualize the learned features and decision boundaries of the network, a 2-dimensional toy dataset referred to as Moons is used. It comprises of two interleaving half circles, each belonging to a different class. Additionally, to assess the results on less trivial settings, we utilize the MNIST dataset [Den12] consisting of handwritten digits. Figure 3.1 provides visualizations of the data.

(a) Sample images from MNIST

(b) Data distribution of Moons

Figure 3.1: Visualizations of the labelled training data

3.2 Model zoo

On the Moons data, 2-layer fully connected networks with varying numbers of hidden neurons are trained on the same data. The hidden layer uses ReLU activation, and the output consists of a single node with sigmoid activation. Keeping the overall complexity low, then 4-layer fully connected networks with the shape $[784, \mathrm{w}, \mathrm{w}, \mathrm{w}, 10]$ are trained on the MNIST dataset, where w is the width of the hidden layer. Additionally, Layer Normalization [BKH16] is applied to each hidden layer. The hidden units use ReLU activation, and the final output layer uses softmax activation.

On the Moons dataset, 50 models for each width are trained using 512 training samples over 100 epochs with binary cross-entropy loss. On the MNIST dataset, 50 models for each width are trained over 60 epochs using cross-entropy loss. In both cases, the AdamW optimizer [LH17] is used to perform mini-batch gradient descent with a batch size of 256. Lastly, cosine annealing of the learning rate is used on Moons, and the 1cycle learning rate policy is used on MNIST. All of the above are implemented with the standard PyTorch library. The code can be accessed at https://github.com/Adhithyan8/LMC.

Width	**Loss**		**Accuracy**	
	train	**test**	**train**	**test**
2	0.264 ± 0.088	0.284 ± 0.084	0.879 ± 0.074	0.859 ± 0.077
4	0.186 ± 0.087	0.206 ± 0.092	0.920 ± 0.041	0.903 ± 0.049
8	0.105 ± 0.095	0.124 ± 0.105	0.957 ± 0.045	0.941 ± 0.057
16	0.019 ± 0.038	0.028 ± 0.043	0.996 ± 0.017	0.991 ± 0.024
32	0.005 ± 0.001	0.012 ± 0.002	1.0 ± 0.0	0.996 ± 0.001
64	0.003 ± 0.001	0.009 ± 0.002	1.0 ± 0.0	0.997 ± 0.001
128	0.002 ± 0.0	0.006 ± 0.001	1.0 ± 0.0	0.998 ± 0.001
256	0.002 ± 0.0	0.005 ± 0.001	1.0 ± 0.0	0.999 ± 0.001
512	0.001 ± 0.0	0.004 ± 0.0	1.0 ± 0.0	0.998 ± 0.001

Table 3.1: Mean and standard deviation of losses and accuracy computed over 50 networks trained and evaluated on Moons

Width	Loss		Accuracy	
	train	**test**	**train**	**test**
2	2.301 ± 0.0	2.301 ± 0.0	0.112 ± 0.0	0.113 ± 0.002
4	0.660 ± 0.047	0.690 ± 0.056	0.803 ± 0.017	0.798 ± 0.018
8	0.172 ± 0.007	0.225 ± 0.010	0.950 ± 0.002	0.938 ± 0.003
16	0.040 ± 0.002	0.134 ± 0.008	0.989 ± 0.001	0.966 ± 0.002
32	0.003 ± 0.0	0.120 ± 0.010	0.999 ± 0.0	0.978 ± 0.002
64	0.001 ± 0.0	0.108 ± 0.008	1.0 ± 0.0	0.983 ± 0.001
128	0.0 ± 0.0	0.102 ± 0.009	1.0 ± 0.0	0.985 ± 0.001
256	0.0 ± 0.0	0.102 ± 0.009	1.0 ± 0.0	0.985 ± 0.001
512	0.0 ± 0.0	0.111 ± 0.009	1.0 ± 0.0	0.985 ± 0.001

Table 3.2: Mean and standard deviation of losses and accuracy computed over 50 networks trained and evaluated on MNIST. Networks of width 2 are underpowered and perform close to random

3.3 Permutation alignment

Generally two independently trained networks are not linearly mode connected. To evaluate if they are linearly mode connected up to permutation, all permutations of their hidden neurons and loss along their pairwise connecting line should be considered, in principle. However, this becomes computationally infeasible when the hidden layer has a large number of neurons. For instance, a hidden layer with m units has $m!$ permutations, and if the network has k hidden layers, that is $k \cdot m!$ total configurations to consider. So instead, a greedy approach of selecting one model as the reference and permuting the hidden neurons of the others is chosen. The goal is to align the neurons such that the new weights are in the same basin, and thus are linear mode connected. Estimating this target permutation is a challenging task for which no method currently available guarantees an optimal solution. One approach involves implicit Sinkhorn differentiation [Peñ+22], to estimate a permutation that minimizes a chosen cost function through gradient descent, such as the expected loss along the line connecting two network weights. However, this requires extensive hyper-parameter tuning, and the results obtained were sub-optimal. In this thesis, a much simpler heuristic method called weight matching [AHS23] is used. It aims to find the optimal permutation by minimizing the Frobenius norm between the weights of two networks. For example, considering two 2-layer networks

with no bias terms, the permutation P sought is as follows:

$$
\begin{aligned}
&\arg\min_P \left[\|W_1^A - PW_1^B\|_F^2 + \|W_2^A - W_2^B P^T\|_F^2\right] \\
&= \arg\min_P \begin{bmatrix} \text{tr}\left((W_1^A - PW_1^B)^T(W_1^A - PW_1^B)\right) \\ + \text{tr}\left((W_2^A - W_2^B P^T)^T(W_2^A - W_2^B P^T)\right) \end{bmatrix} \\
&= \arg\min_P \begin{bmatrix} \text{tr}\begin{pmatrix} (W_1^A)^T W_1^A - (W_1^A)^T P W_1^B \\ - (W_1^B)^T P^T W_1^A + (W_1^B)^T P^T P W_1^B \end{pmatrix} \\ + \text{tr}\begin{pmatrix} (W_2^A)^T W_2^A - (W_2^A)^T W_2^B P^T \\ - P(W_2^B)^T W_2^A + P(W_2^B) T W_2^B P^T \end{pmatrix} \end{bmatrix} \\
&= \arg\min_P \begin{bmatrix} - \text{tr}\begin{pmatrix} (W_1^A)^T P W_1^B + (W_1^B)^T P^T W_1^A \\ + (W_2^A)^T W_2^B P^T + P(W_2^B)^T W_2^A \end{pmatrix} \\ + \text{tr}\left(P(W_2^B) T W_2^B P^T\right) \end{bmatrix}, \text{as } P^T P = I \\
&= \arg\max_P \left[\text{tr}\begin{pmatrix} P W_1^B (W_1^A)^T + P^T W_1^A (W_1^B)^T \\ + P^T (W_2^A)^T W_2^B + P(W_2^B)^T W_2^A \end{pmatrix}\right] \\
&= \arg\max_P \; 2\left(\text{tr}\left(P W_1^B (W_1^A)^T + P(W_2^B)^T W_2^A\right)\right), \text{as tr}(A^T) = \text{tr}(A) \\
&= \arg\max_P \; \text{tr}\left(P^T\left(W_1^A (W_1^B)^T + (W_2^A)^T W_2^B\right)\right) \\
&= \arg\max_P \langle P, W_1^A (W_1^B)^T + (W_2^A)^T W_2^B \rangle_F, \text{ as } \langle A, B\rangle_F = \text{tr}(A^T B)
\end{aligned}
$$

For the 2-layer neural networks, optimal permutation can be solved as a linear assignment problem (LAP). However, for deeper networks where multiple hidden layers may need to be permuted, it is no longer a linear assignment problem and estimating the permutations are shown to be NP-hard [AHS23]. To address this, the layers are iteratively aligned using their incoming and outgoing weights, as well as biases, with an LAP solver until convergence is reached. In the context of this thesis, a fixed number of iterations per layer, specifically 50 iterations, is used as the stopping criterion for weight matching.

To align 50 networks, the first network is chosen as the reference without loss of generality. Then, the weights of the remaining networks are permuted using weight matching. This new set of reparameterized networks are considered to be permutation aligned.

3.4 Stochastic Weight Averaging

To understand how weight averaging improves performance and its relation to permutation alignment, Stochastic Weight Averaging (SWA) [Izm+18] is chosen as the method to be studied. The idea is to average the weight iterates near the end of the gradient descent trajectory while maintaining a constant or cyclic learning rate. The rationale is that these weights correspond to various well-performing networks, and their mean is likely to reside in a wide minima of the loss surface. This method has been empirically shown to improve generalization in various network architectures and data modalities. Figure 3.2 provides an illustration of this idea.

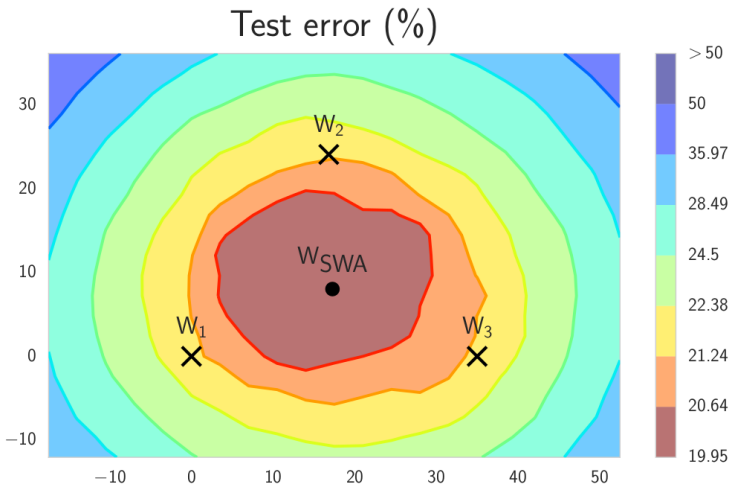


Figure 3.2: Illustration of Stochastic Weight Averaging depicting the samples and the averaged network on the loss surface. Figure is adapted from Izmailov et al. [Izm+18]

In this thesis, a trained network is randomly chosen from each hidden layer width and further trained for 20 epochs using a constant learning rate. At the end of each epoch, the network weights are saved as the SWA samples. Subsequently, linear mode connectivity among these samples and the averaged model is studied.

3.5 Summary

Having described the network architectures, the training data, and the method for permutation alignment, the next chapter begins with examining the loss

along naive interpolations of network weights before any weight permutations. This is compared with loss along permutation aligned networks to investigate if they are linearly mode connected up to permutation. Finally, similar experiments are done on SWA samples to understand weight averaging.

Chapter 4

Results

This chapter presents the results from measuring the loss along interpolations of trained network weights before and after reparameterization for permutation alignment. To quantify linear mode connectivity, we utilize the ϵ measure, as introduced in Section 2.1.1. Based on these observations, further analysis to explain interesting trends is given in the subsequent chapter.

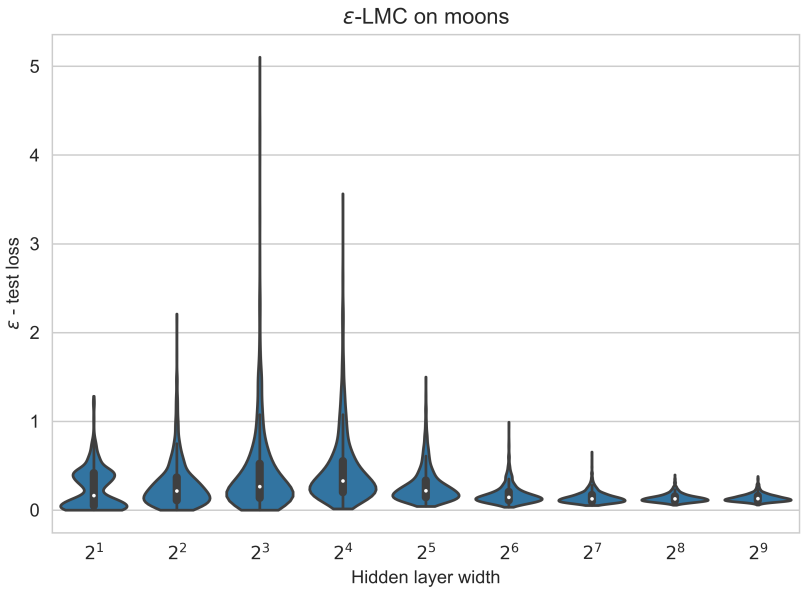
4.1 Naive weight interpolation

To assess linear mode connectivity between independently trained networks, the loss along the connecting line for all pairs of network weights is calculated. The test loss along is considered as there are no significant differences if the train loss is used instead. The loss is computed at 11 equally spaced weights $\theta_\alpha = \alpha\theta_A + (1-\alpha)\theta_B$ by varying α from zero to one. The ϵ loss is computed as,

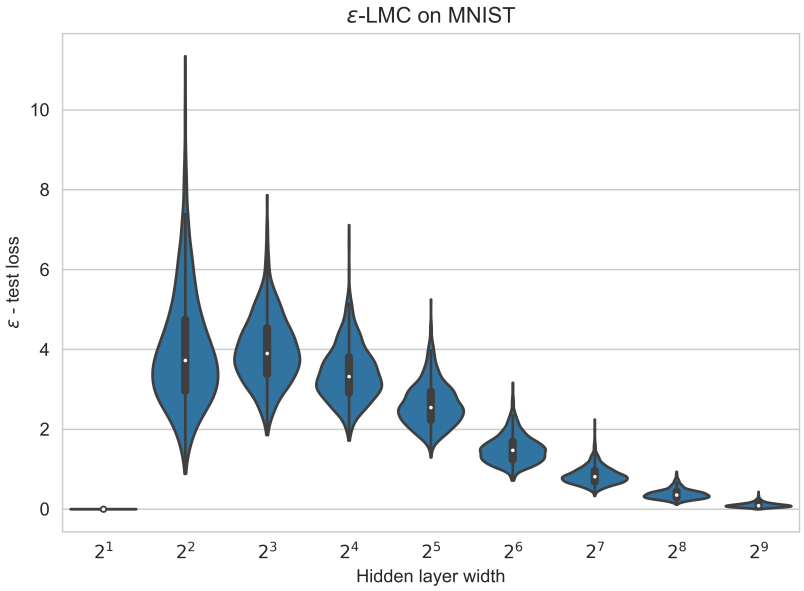
$$\epsilon = \max_{\alpha} \mathcal{L}_S(\theta_\alpha) - \max(\mathcal{L}_S(\theta_A), \mathcal{L}_S(\theta_B))$$

The distribution of ϵ against the network widths for both datasets is depicted in Figure 4.1. The x-axis denoting the hidden layer width is on the logarithmic scale.

In Figure 4.1a, networks with a hidden layer width of 2 appear to be linearly mode connected. However, this is misleading because these networks perform no better than random, as shown in Table 3.2. Both the trained networks and their linear interpolations yield high losses, resulting in an ϵ value that is close to zero. Interestingly, ϵ decreases with increasing hidden layer width for networks trained on both datasets. This is counterintuitive, considering the theoretical insights shared in Section 2.1.2. To understand this, the loss along



(a) ϵ-loss between networks trained on Moons. The networks are more linearly mode connected with increasing hidden layer width



(b) ϵ-loss between networks trained on MNIST. Again linear mode connectedness increases with hidden layer width

Figure 4.1: Distribution of ϵ-LMC under naive interpolations of network weights for different hidden layer widths

the linear interpolations is visualized between all pairs of trained networks for both Moons and MNIST. The y-axis is scaled differently in each subplot.

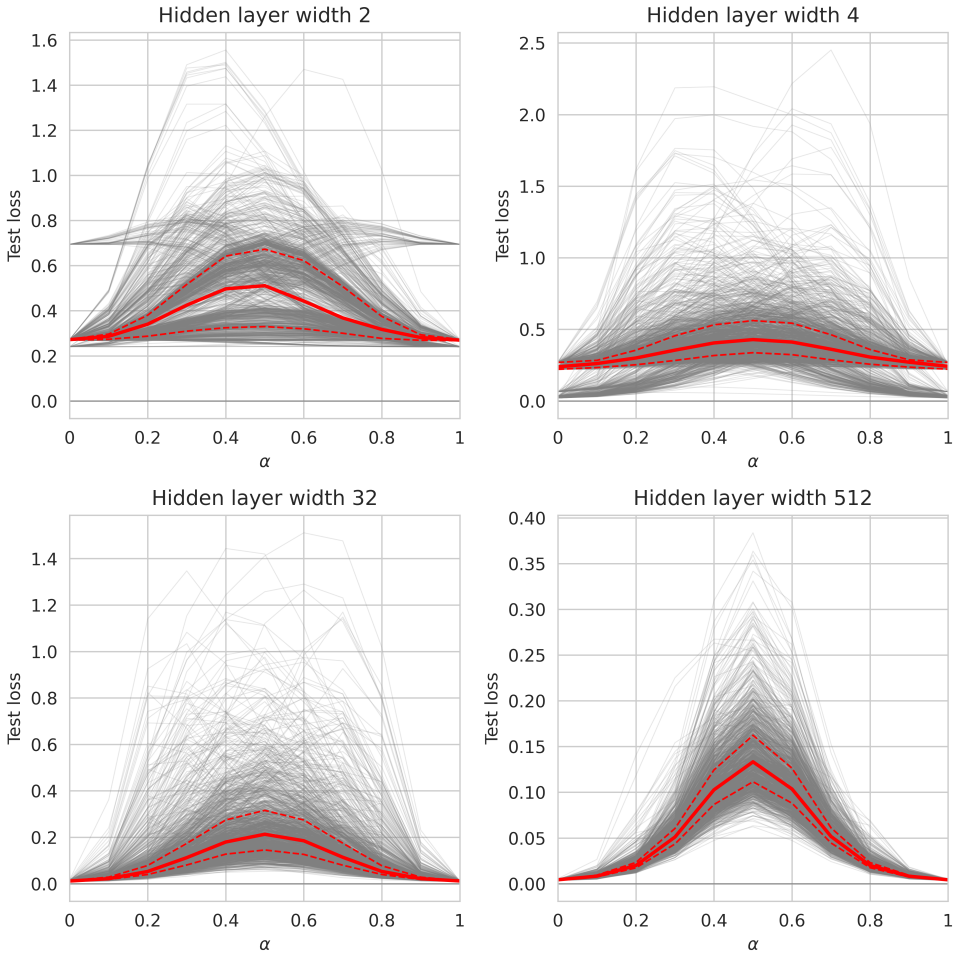


Figure 4.2: Loss along the lines connecting network pairs trained on Moons. Point-wise median and the Q1, Q3 quartiles are highlighted in red, and the subplots are zoomed in for large widths. The average network is slightly worse for large hidden layer widths.

While the distributions in Figure 4.1 do not immediately reveal this, upon closer examination, it is apparent that the average of two trained networks does perform worse than either of the individual networks. However, what is surprising is that the performance is still remarkably good. Additionally, for wide networks, some values of α result in interpolated networks that achieve even lower loss than the trained networks, as shown in Figure 4.3. To explain the low loss of naive averages, decision boundaries and features learned by the

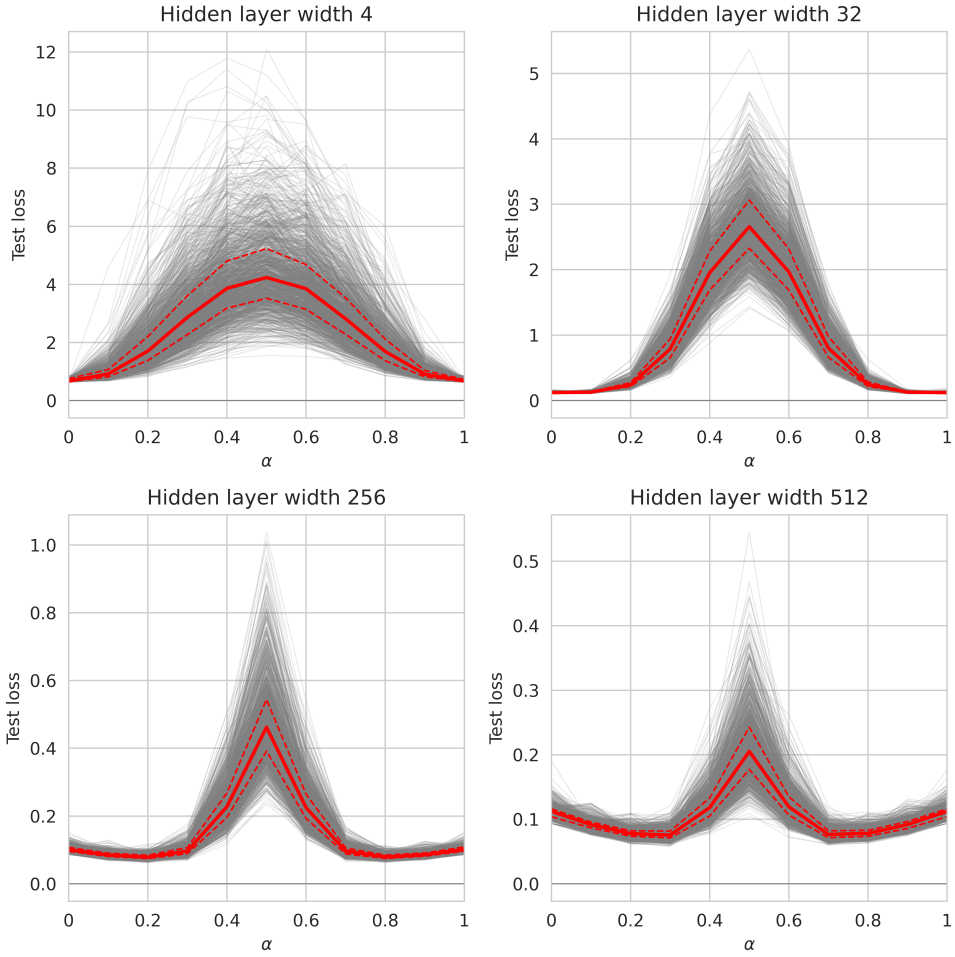


Figure 4.3: Loss along the lines connecting network pairs trained on MNIST. Point-wise median and the Q1, Q3 quartiles are highlighted in red, and the subplots are zoomed in for large widths. The average network is slightly worse for large hidden layer widths.

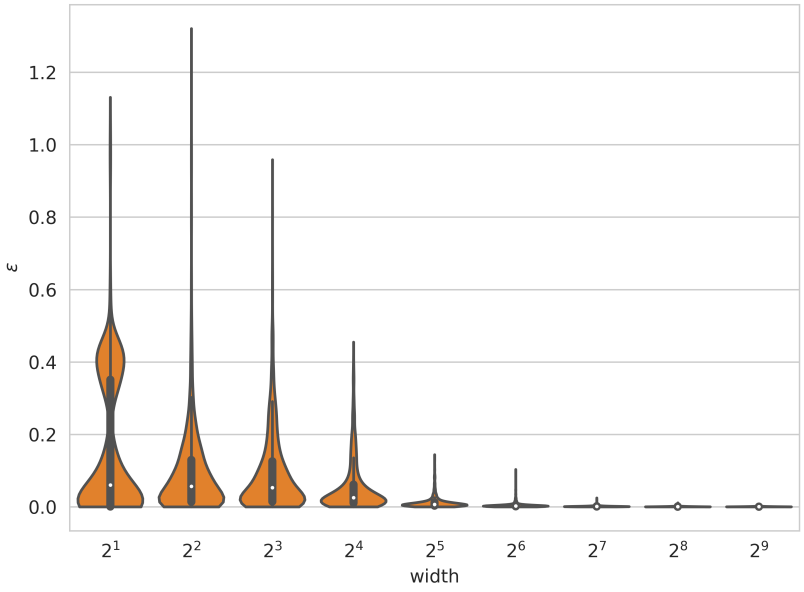
networks are analysed in the subsequent chapter. The key finding is that the low loss value of average networks is due to the redundancies present in the features represented at the hidden layer. So the averages are likely to maintain their performance even without explicit permutation alignment. However, this effect diminishes when the networks are pruned to reduce such redundancies.

4.2 Permutation aligned networks

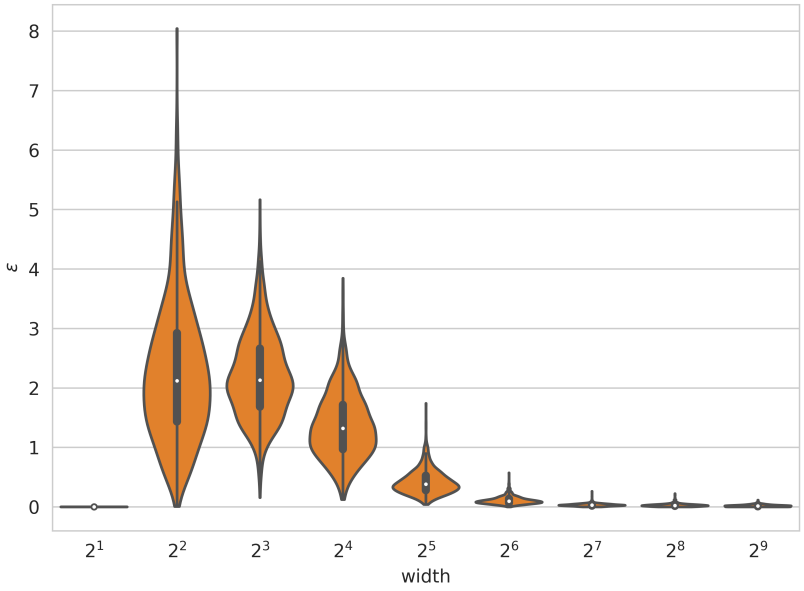
Now, to explore if the trained networks are linearly mode connected up to permutation of hidden neurons, assume that the optimal permutation with respect to a reference can be found through weight matching. As described in Section 3.3, using a random network as the reference, the other network weights are permuted. The ϵ loss from pairwise linear interpolation of aligned networks is measured on both datasets. The results are depicted in Figure 4.4, where the x-axis is on a logarithmic scale.

The median ϵ loss among reparameterized networks is consistently lower than that among the originals, regardless of the hidden layer widths. Note that the 4-layer networks with a width of 2 trained on MNIST have ϵ values close to zero, which is due to them performing only slightly better than random, as explained earlier. These reductions in ϵ are further emphasized in Figure 4.5.

On the moons data, narrow networks have the mode of the ϵ distribution close to zero. However, there is a long tail, indicating that some networks are not linearly mode connected even up to permutation of hidden neurons. This suggests there may be clusters of networks that are linearly mode connected up to permutation among themselves but not between them. Conversely, for narrow networks trained on MNIST, the median ϵ is much larger than zero, and such clusters with linear mode connectivity up to permutation appear to be non-existent. To explicitly bring out these clusters, the ϵ loss is treated as a measure of similarity between networks. When ϵ is close to zero, it implies the network pairs learn similar features up to permutation of hidden units. Conversely, larger ϵ indicate that the features cannot be matched, and the networks are diverse. They may produce similar outputs but do this by evaluating dissimilar features. But ϵ is not a true metric as it does not obey the triangle inequality. To compute the clusters, hierarchical clustering with the single linkage method is done on the pairwise ϵ matrix. The results are shown in Figure 4.6. In both, the network with index 0 is the reference to which the others are aligned, and is marked on the axis. The colors are scaled such that zero ϵ is the minimum and the mean test loss of the networks is the maximum.

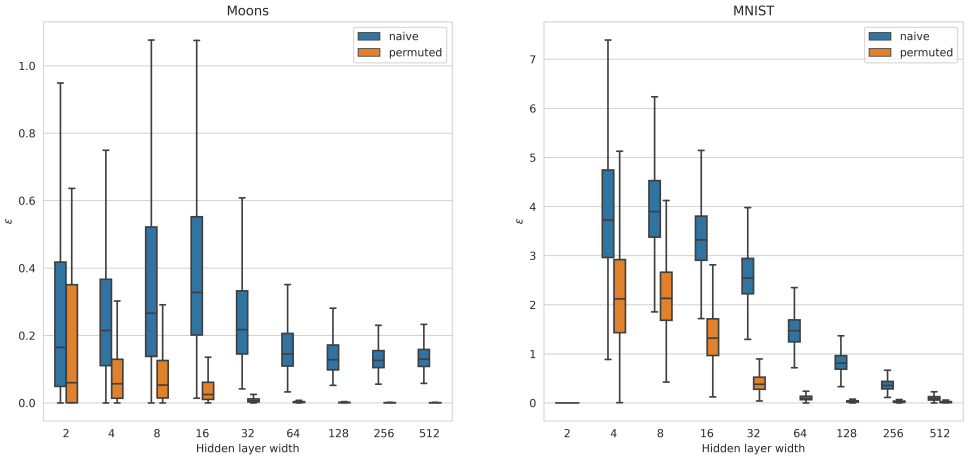


(a) ϵ-loss between networks trained on Moons after reparameterization



(b) ϵ-loss between networks trained on MNIST after reparameterization

Figure 4.4: Distribution of ϵ-LMC up to permutation by interpolating reparameterized network weights for different hidden layer widths. Linear mode connectivity improves upon reparameterization for permutation alignment



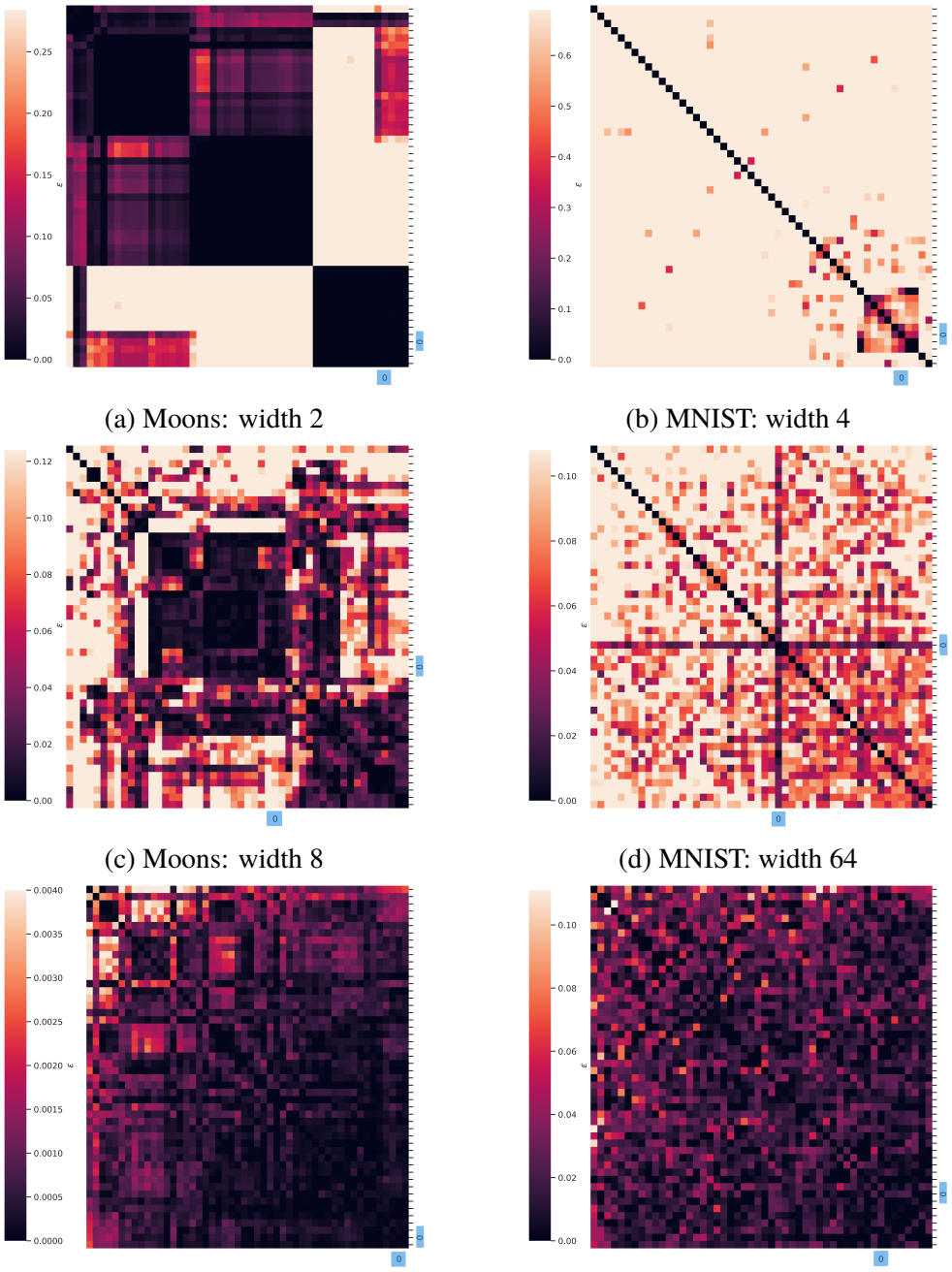
(a) ϵ-LMC on Moons

(b) ϵ-LMC on MNIST

Figure 4.5: Box plot of ϵ-loss from linear interpolations of trained networks compared to networks reparameterized for permutation alignment. Outliers are omitted from the plots. Reparameterization reduces the loss barrier

As expected, in Figure 4.6a narrow networks trained on moons form clear clusters, corresponding to the different features or half-spaces learned for the classification task. However, for narrow networks trained on MNIST, no such prominent clusters are seen. Most networks are not linearly mode connected up to permutation, indicated by low ϵ being mostly along the diagonal. As the hidden layer width is increased, we do observe multiple minima up to permutation that the gradient descent solutions converge to, as seen in Figure 4.6c. Interestingly, in Figure 4.6d the networks appear to be linearly mode connected up to permutation with the reference network, but not among themselves, from by the clear horizontal and vertical line. This seems to be an artifact of how the permutation is found. The weight matching algorithm is based on a heuristic that closeness of reparameterized network weights in an Euclidean sense would give the optimal permutation, but this is an arbitrary choice. And for deeper networks, the greedy approach to finding the permutation of each hidden layer may also result in a sub-optimal outcome.

In wide networks, as shown in Figure 4.6f for MNIST and Figure 4.6e for moons, nearly all trained networks appear to be linearly mode connected up to permutation. Why do all these networks appear to prefer just one minimum (up to permutation) among the many that could yield similar performance? To further explore this, we focus on the training dynamics, specifically the influence of weight initialization and choice of optimizer in the next chapter.



(a) Moons: width 2

(b) MNIST: width 4

(c) Moons: width 8

(d) MNIST: width 64

(e) Moons: width 512

(f) MNIST: width 512

Figure 4.6: Pairwise ϵ loss between reparameterized networks trained on Moons (left) and MNIST (right) after reordering the indices with hierarchical clustering. Colors are scaled such that the mean test loss is the maximum. The reference network to which others are aligned is highlighted on the axis

4.3 Understanding SWA

Instead of averaging independently trained networks, Stochastic Weight Averaging (SWA) uses weights along the end of the training trajectory, and averages them to produce a single network with improved test loss compared to any individual weight. Does this result in samples that are implicitly permutation aligned? Or does the average network only appear to be good due to feature redundancies like in the case of naive averaging? The loss neighborhood of these SWA weight samples are probed to check for linear mode connectivity by considering pairwise linear interpolations.

One network each from hidden layer widths of 8, 16, 64, and 512, trained on MNIST is randomly chosen. They were originally trained for 60 epochs. Starting from this weight, training is continued for 10 more epochs at a high learning rate of 0.05. After each epoch, the network weights are saved and will be used as SWA samples to be averaged. In Figure 4.7, the test losses of both the individual weight samples and the running average model is presented.

Except in narrow networks, Stochastic Weight Averaging (SWA) obtains a final network with lower test loss. However, the individual weight samples perform as good as the original trained network or at times worse, as seen in Figure 4.7 for hidden layer width 16. Without any reparameterization for permutation alignment, the test loss along pairwise linear interpolations of the samples is measured. If a large ϵ value is observed, it may suggest that these samples are from diverse minima, and the low loss of the average network is due to feature redundancies.

But to the contrary, Figure 4.8 shows that even the pairwise averages often incur a lower test loss than the individual pairs. So the samples are often from the same loss basin, as ϵ values are close to zero, with few outliers. This suggests that the weight samples obtained through Stochastic Weight Averaging (SWA) are from a near convex basin in the loss landscape and are implicitly permutation aligned from sharing most of their initial training trajectory.

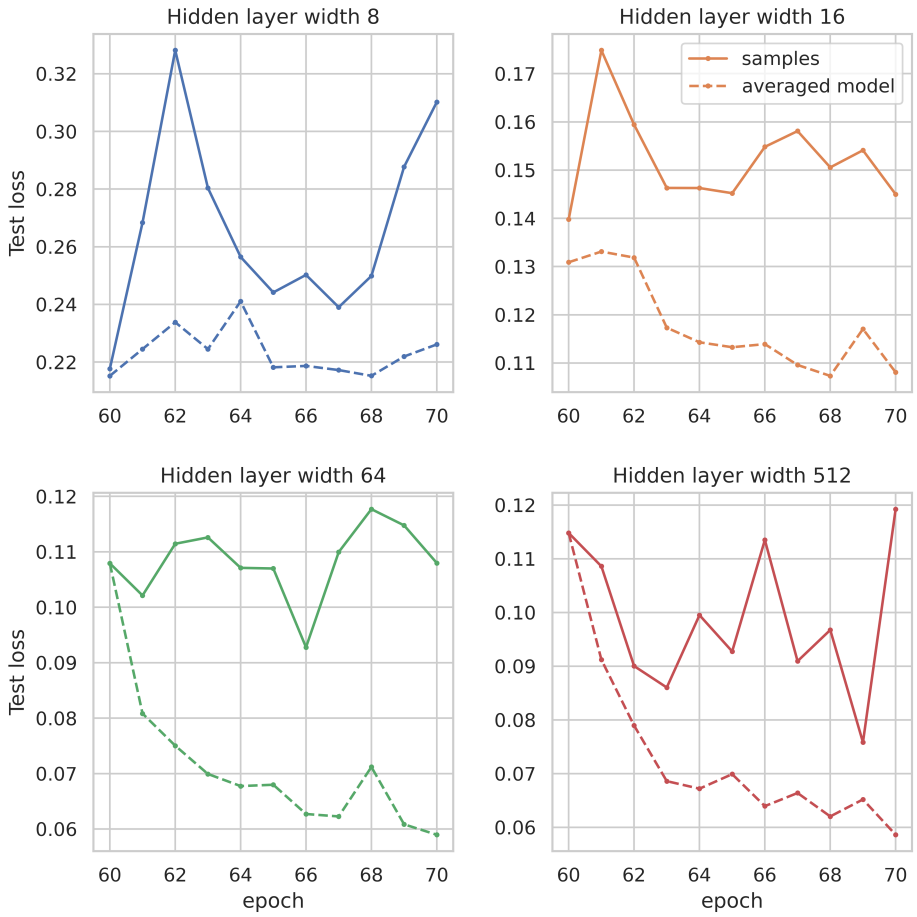


Figure 4.7: Test loss of SWA weight samples and the running average network evaluated on MNIST for different hidden layer widths. The individual samples are always worse than the their average

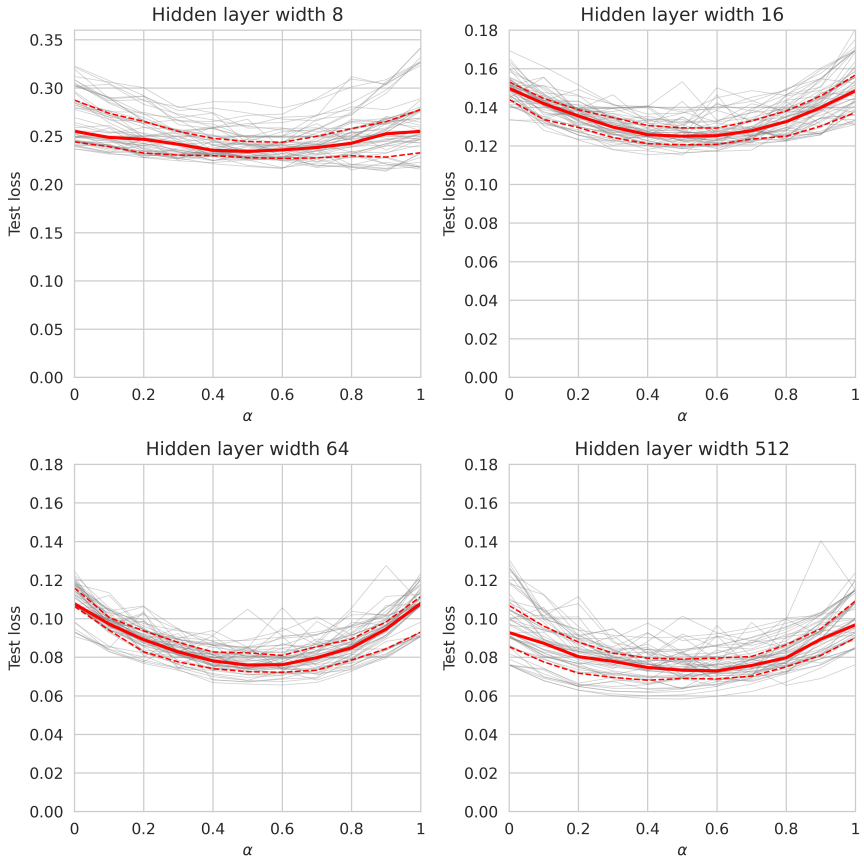


Figure 4.8: Loss along pairwise linear interpolations of SWA weight samples evaluated on MNIST. The point-wise median loss and the Q1, Q3 quartiles are highlighted in red. Even pairwise averages incur lower loss than the original networks

Chapter 5

Analysis and Discussion

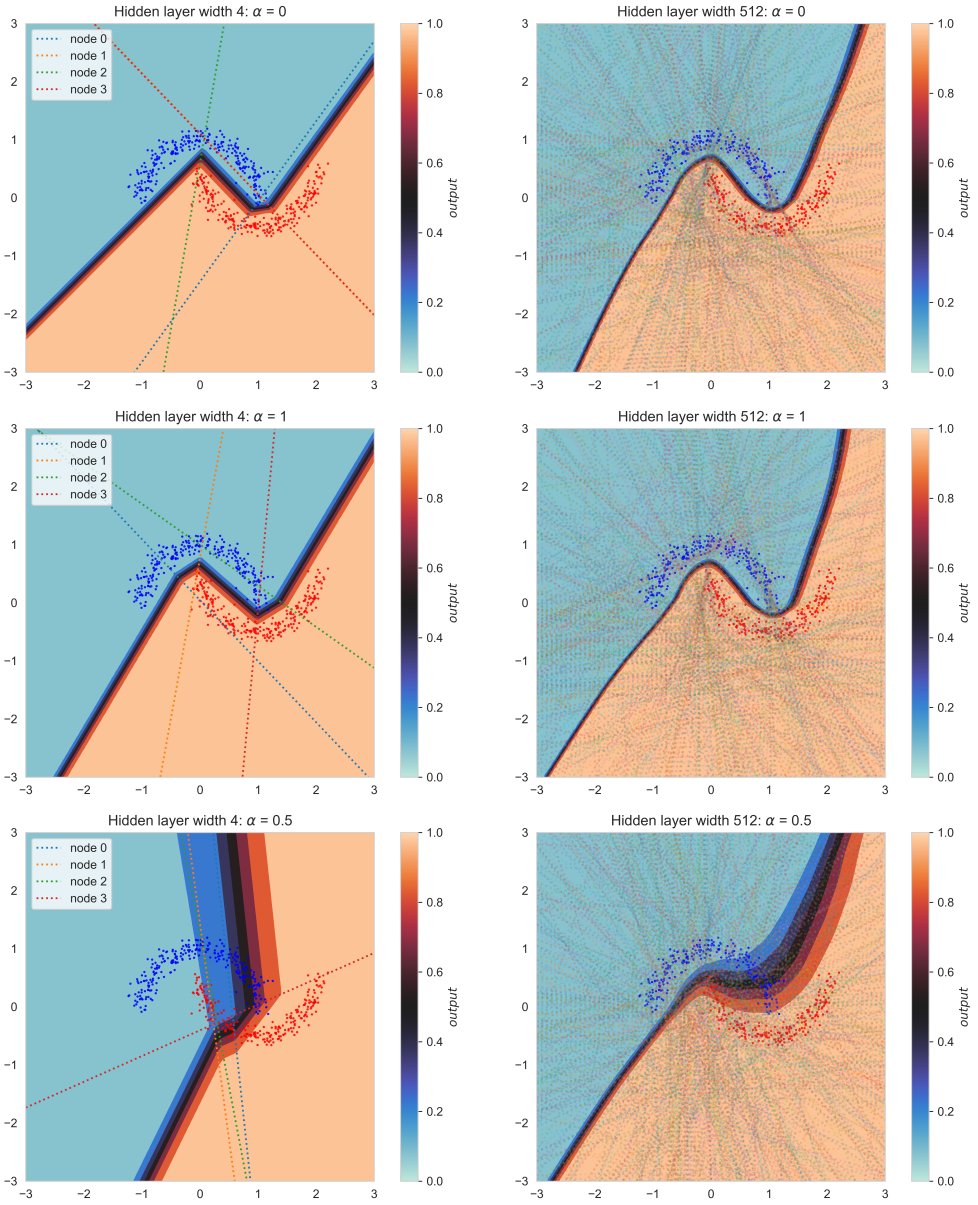
From the last chapter, the results that warranted further investigation, are explored in the sections that follow. Specifically, understanding the loss low obtained by naive averages and experiments to verify if wide networks always converge to the same minima up to permutation are discussed.

5.1 Understanding naive weight averaging

The naive average of two trained networks, without any reparameterization, was shown to produce an average network with relatively low loss, particularly when the hidden layer width is large in Section 4.1. The decision boundary of the networks and the hyperplanes computed at each hidden neuron are visualized to understand how they partition the input space. As a reminder, the Moons dataset consists of two-dimensional inputs, and the networks trained on this dataset have one hidden layer. The visualizations for a random pair of trained networks are shown in Figures 5.1 for narrow and wide networks respectively.

As seen in Figure 5.1a, moving from one network weight to another along the connecting line in the parameter space is equivalent to rotating and translating the hyperplane corresponding to each hidden neuron. Without explicitly matching these neurons, the average network performs poorly, as expected.

For the wide networks shown in Figure 5.1b, many neurons appear to learn the same hyperplane. And the average network only shows a small degradation in test accuracy. This is potentially due to the redundancy in features computed by hidden neurons. When a large proportion of neurons in both networks compute similar features, they are likely to have the same positional index



(a) Two trained narrow networks (top) and their weight-wise average (bottom)

(b) Two trained wide networks (top) and their weight-wise average (bottom)

Figure 5.1: Decision boundary and the hyperplanes corresponding to hidden units for a pair of narrow networks of width 4 (left), and wide networks of width 512 (right) along their connecting line. Many neurons appear to compute the same features in wide networks.

even without explicit permutation alignment. So, upon averaging, the learned features may be preserved, enabling the average network to retain some classification power. To test this hypothesis, a network of width 512 is randomly chosen and the pairwise cosine similarity of its hidden neurons are examined. Specifically, the incoming weights and biases of each neurons are treated as its vector representation. If many neurons show high similarity among themselves, it would lend support to the hypothesis. Figure 5.2 presents the pairwise cosine similarity after hierarchical clustering using the single linkage criteria.

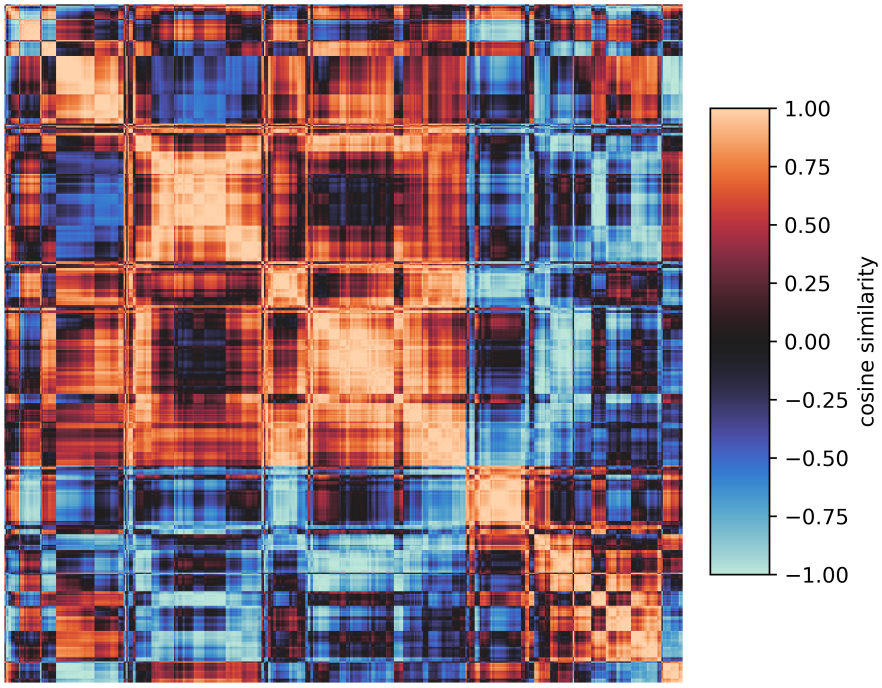


Figure 5.2: Pairwise cosine similarity between the features computed by the hidden neurons of a network with width 512 trained on Moons. x and y-axis correspond to the positional indices of the neurons after reordering based on hierarchical clustering. Large clusters of neurons compute similar features

Figure 5.2 qualitatively shows that the neurons indeed compute similar features, as seen from the clusters based on pairwise cosine similarity. As an additional test, the feature redundancy is reduced by combining neurons that learn similar features, and the loss of the average network is calculated. An arbitrary cosine similarity threshold of 0.99 is used for this purpose. If the similarity between two neurons, n_1 and n_2, exceeds this threshold, then they are combined following the guidelines from Petzka, Trimmel,

and Sminchisescu [PTS20]. If the incoming weights and biases of n_1 are proportional to those of n_2 by a factor of λ, then n_2 can be removed from the network by updating the outgoing weights of n_1 as $\mathbf{v}_1 + \lambda\mathbf{v}_2$, where $\mathbf{v}_1$ and $\mathbf{v}_2$ represent the original outgoing weights of n_1 and n_2 respectively. After such pruning, the reduced model sizes and their corresponding test losses are summarized in Table 5.1.

Configuration	**Width**	**Test Loss**	**Test Accuracy**
Full	2	0.285 ± 0.084	0.859 ± 0.077
Reduced	2 ± 0.0	0.285 ± 0.084	0.859 ± 0.077
Full	4	0.206 ± 0.092	0.903 ± 0.049
Reduced	3.720 ± 0.492	0.220 ± 0.111	0.897 ± 0.056
Full	8	0.124 ± 0.105	0.941 ± 0.057
Reduced	7.180 ± 0.865	0.148 ± 0.126	0.934 ± 0.062
Full	16	0.028 ± 0.043	0.991 ± 0.024
Reduced	13.040 ± 1.216	0.057 ± 0.086	0.979 ± 0.037
Full	32	0.012 ± 0.002	0.996 ± 0.001
Reduced	23.280 ± 1.887	0.045 ± 0.088	0.986 ± 0.026
Full	64	0.009 ± 0.002	0.997 ± 0.001
Reduced	39.260 ± 1.598	0.050 ± 0.111	0.985 ± 0.028
Full	128	0.006 ± 0.001	0.998 ± 0.001
Reduced	73.620 ± 1.340	0.047 ± 0.122	0.987 ± 0.023
Full	256	0.005 ± 0.001	0.999 ± 0.001
Reduced	141.020 ± 2.005	0.048 ± 0.148	0.987 ± 0.027
Full	512	0.004 ± 0.0	0.998 ± 0.001
Reduced	271.940 ± 2.176	0.027 ± 0.052	0.990 ± 0.019

Table 5.1: Mean and standard deviation of network widths, test losses and accuracies on Moons dataset after pruning of redundant hidden neurons. Pruning only results in minor degradation during inference

The reduced networks perform close to that of the originals, with a small degradation attributed to the arbitrary cosine similarity threshold of 0.99. To evaluate if these reduced networks are linearly mode connected, the loss along the pairwise interpolations should be measured. But some pairs have different hidden layer widths, so in such cases, the narrower network is padded with neurons that have both incoming weights, biases, and outgoing weights set to zero. The resulting distribution of ϵ loss between the reduced networks is shown in Figure 5.3.

The ϵ-loss is now comparable across different network widths after combining neurons that represent similar features. And the naive average

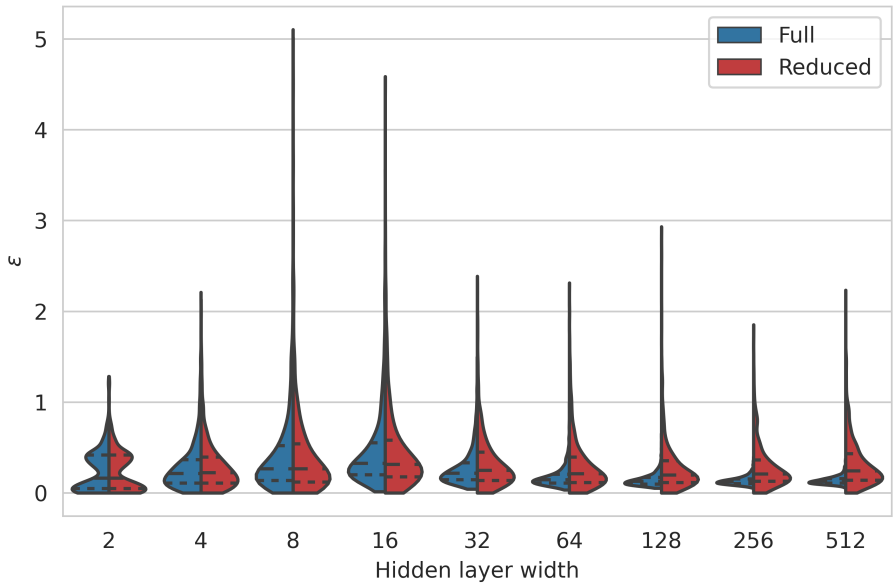


Figure 5.3: Distribution of ϵ-LMC under naive interpolations of networks compared to the networks after reduction in feature redundancy. The loss low of naive averages is absent once feature redundancies are lowered.

network incurs much higher loss than the original trained networks. This confirms that naive averaging of independently trained networks appear to work due to the feature redundancies in the hidden layer.

5.2 LMC in reparameterized networks

In Section 4.2, wide networks trained on both the Moons and MNIST datasets appeared to be linearly mode connected up to permutation. To verify if this claim is robust to training dynamics, the weight initialization strategy and choice of optimizer are varied. If LMC up to permutation is no longer observed, it may narrow down the factors responsible for this.

Two different weight initialization schemes: Kaiming normal and Kaiming uniform [He+15], and two different optimizers: AdamW and RMSprop, are considered. Under each of the four configurations resulting from their combinations, ten 2-layer networks with hidden layer width of 512 are trained on Moons. The training procedure remains the same as described in Section 3.2, and the hyperparameters of each configuration are tuned such that they consistently produce networks with low test low. To ensure comparability, the

training and test performances under all these configurations are verified to be high. The train and test metrics are shared in Table 5.2.

Configuration	**Train loss**	**Test loss**	**Train acc.**	**Test acc.**
Norm., AdamW	0.0 ± 0.0	0.003 ± 0.0	1.0 ± 0.0	0.998 ± 0.0
Norm., RMSp.	0.0 ± 0.0	0.003 ± 0.001	1.0 ± 0.0	0.999 ± 0.001
Unif., AdamW	0.001 ± 0.0	0.003 ± 0.0	1.0 ± 0.0	0.998 ± 0.001
Unif., RMSp.	0.0 ± 0.0	0.003 ± 0.0	1.0 ± 0.0	0.999 ± 0.001

Table 5.2: Norm.: Kaiming normal, Unif.: Kaiming uniform, RMSp.: RMSprop. Mean and standard deviation of losses and accuracies of networks with width 512 trained on Moons under different weight initialization and optimizer configurations.

We use the network with the least test loss, which is trained using Kaiming normal initialization and RMSprop, as the reference. All the other trained networks are reparameterized to be permutation aligned to it using the weight matching method. The test loss along pairwise linear interpolations of aligned networks is presented in Figure 5.4. If all wide networks with low test loss are linearly mode connected up to permutation, then we would observe low values of ϵ loss.

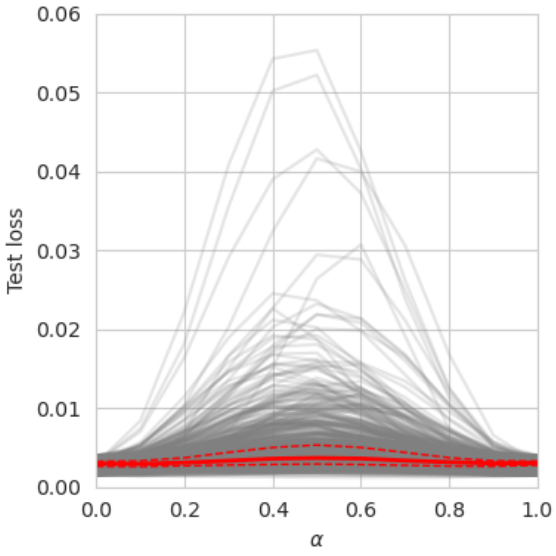


Figure 5.4: Loss along linear interpolations among reparameterized networks of width 512 trained with different initialization and optimizers on Moons. The point-wise median, and Q1, Q3 quartiles are highlighted in red

From Figure 5.4, while the median ϵ is close to zero, some network pairs do not appear to be linearly mode connected up to permutation. Considering ϵ as a measure of similarity as before, the pairwise ϵ matrix between permutation aligned networks is shown in 5.5. But unlike previous analyses, no hierarchical clustering is done to reorder the indices.

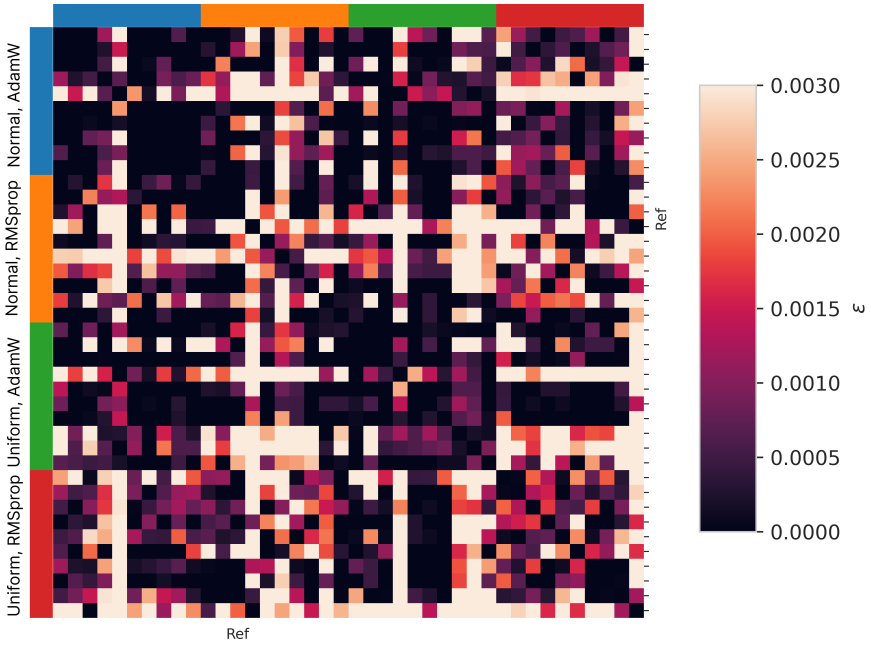
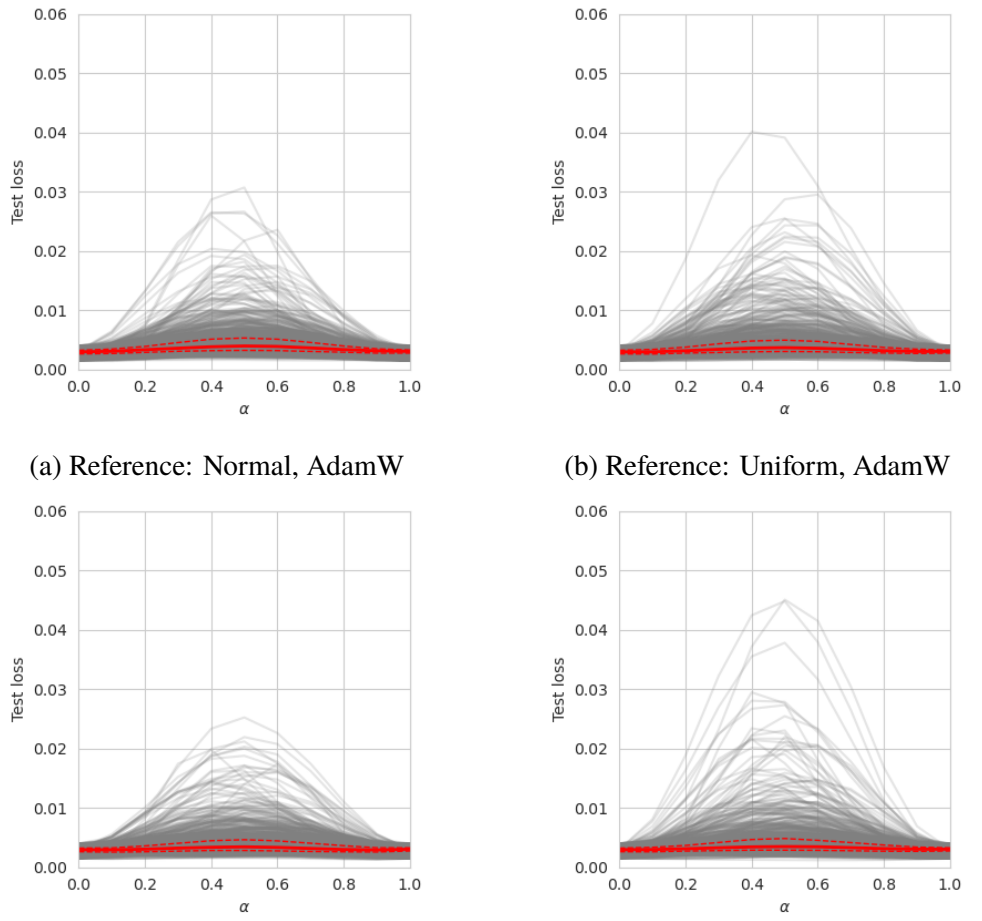


Figure 5.5: Pairwise ϵ-loss between reparameterized networks of width 512 trained on Moons with different initialization and optimizers. The colors are scaled so that the maximum value corresponds to the mean test loss. The reference network to which others are aligned is highlighted on the axis

On average, the networks trained using AdamW optimizer are more linearly mode connected up to permutation among themselves than those trained using RMSprop, irrespective of the initialization scheme. Recall that all these networks have a low and comparable test loss. This challenges the claim that all well-trained, wide networks are linearly mode connected up to permutation, at least under the assumption that the permutation alignment method works. To explain linear mode connectivity up to permutation, it may be necessary to explore the weight space dynamics induced by the different optimizers. To strengthen this result, sensitivity analysis with respect to the choice of reference network, the hidden unit matching algorithm, data complexity, and the depth of trained networks are presented.

5.2.1 Sensitivity to the reference network

The networks were reparameterized to be permutation aligned with the reference network, which was arbitrarily chosen to be the one with the least test loss. Given that this was trained under the Kaiming normal and RMSprop configuration, the experiments are repeated by choosing different references from each of the three other configurations instead . The test loss along the linear interpolations of the reparameterized networks are presented in Figure 5.6, and the pairwise ϵ matrix between the aligned networks are in Figure 5.7.



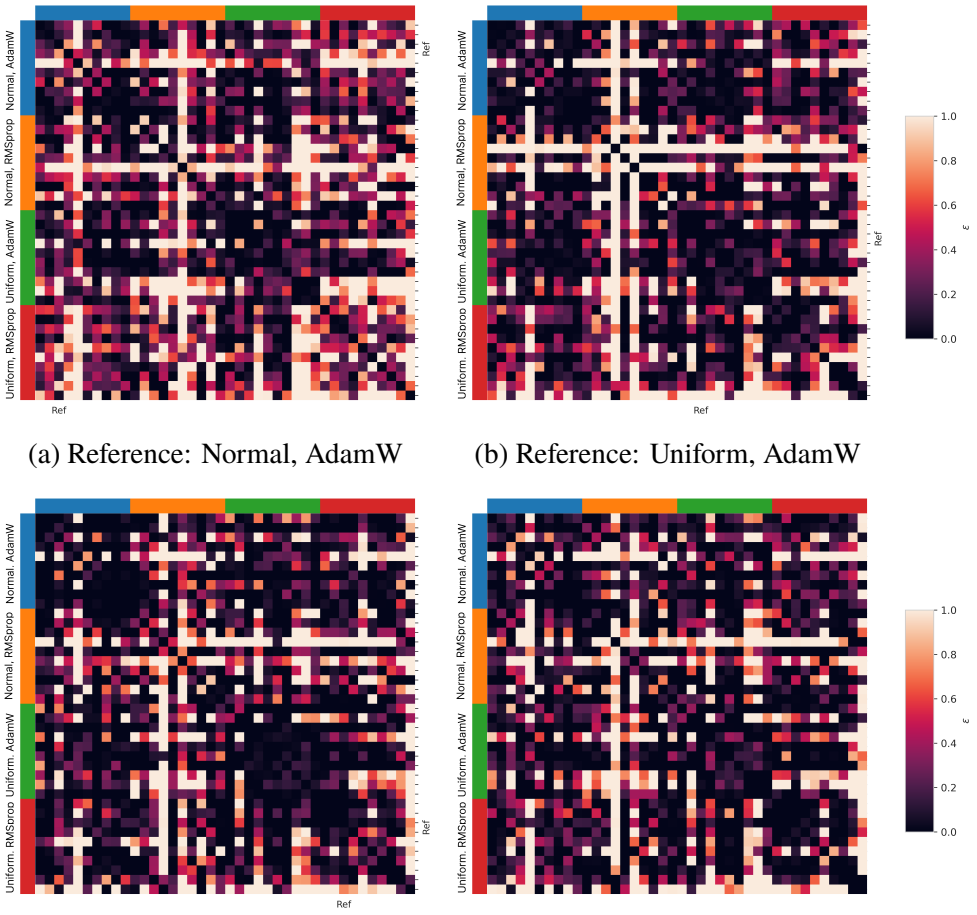
(a) Reference: Normal, AdamW

(b) Reference: Uniform, AdamW

(c) Reference: Uniform, RMSprop

(d) Each pair is aligned to one another

Figure 5.6: Test loss along linear interpolations of aligned networks with width 512 using different reference networks. The point-wise median, and Q1, Q3 quartiles of the loss are highlighted in red



(a) Reference: Normal, AdamW

(b) Reference: Uniform, AdamW

(c) Reference: Uniform, RMSprop

(d) Each pair is aligned to one another

Figure 5.7: Pairwise ϵ-loss between reparameterized networks trained on Moons with different choice of reference networks. The colors are scaled such that the maximum value corresponds to the mean test loss. The reference network is highlighted on the axis in each subplot, except 5.7d, where each network pair is aligned to one another

From the plots, it is evident that the choice of reference network changes the ϵ connectivity, but qualitatively its effects are not significant. Instead of choosing a network as the reference and aligning others to it, we also present an experiment where each pair of network is permutation aligned before measuring the test loss along the linear interpolation. These results are depicted in Figures 5.6d and 5.7d. Interestingly, explicitly aligning each pair of network increases ϵ loss at times. This once again reflects on the heuristic nature of the weight matching methods, which does not guarantee an optimal permutation for LMC.

5.2.2 Sensitivity to data and network depth

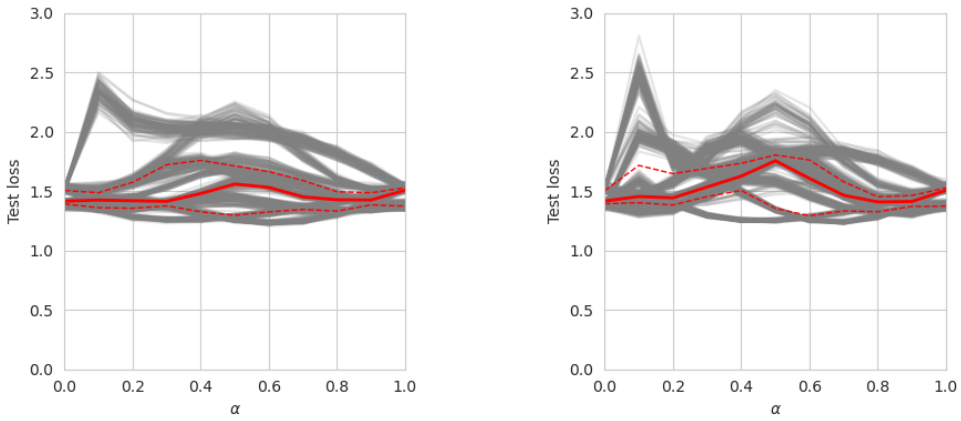
The previous experiments were performed on simple 2-layer ReLU networks trained on the Moons dataset. To evaluate if the conclusions extend to more non-trivial settings, 4-layer ReLU networks with 512 neurons in each hidden layer were trained on the CIFAR-10 [KH+09] classification task. The input space is 3072-dimensional, and the output layer has 10 units. The optimizers considered include AdamW, SGD with momentum, RMSprop, and Adagrad. Hyperparameters for each of the optimizers are individually tuned, and the 1cycle learning rate policy is used to train the networks for 60 epochs with cross-entropy loss. For each optimizer, ten networks are trained, and their average training and testing metrics are presented in Table 5.3.

Optimizer	**Train loss**	**Test loss**	**Train acc.**	**Test acc.**
AdamW	0.597 ± 0.005	1.533 ± 0.015	0.810 ± 0.002	0.577 ± 0.005
SGD	0.691 ± 0.004	1.362 ± 0.015	0.781 ± 0.001	0.579 ± 0.004
RMSprop	0.560 ± 0.004	1.509 ± 0.016	0.819 ± 0.001	0.547 ± 0.004
Adagrad	0.572 ± 0.005	1.402 ± 0.008	0.827 ± 0.001	0.566 ± 0.001

Table 5.3: Mean and standard deviation of losses and accuracies of 4-layer networks trained on CIFAR-10 with different optimizers.

For permutation alignment, two reference networks are used: one trained using SGD, which achieves the highest test accuracy, and another trained using RMSprop. This is to ensure that the results are not sensitive to the choice of the reference network. Using these references, the other networks are reparameterized using the weight matching method. The test loss is evaluated along linear interpolations of the aligned networks under both reference networks, as shown in Figure 5.8.

Figure 5.8 indicates that not all reparameterized networks are linearly mode connected to one another. While the choice of reference affects



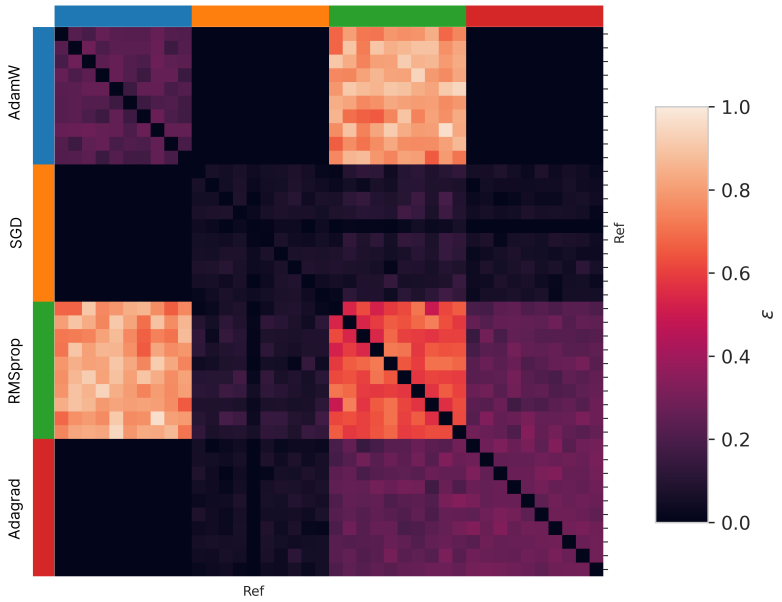
(a) Reference: SGD

(b) Reference: RMSprop

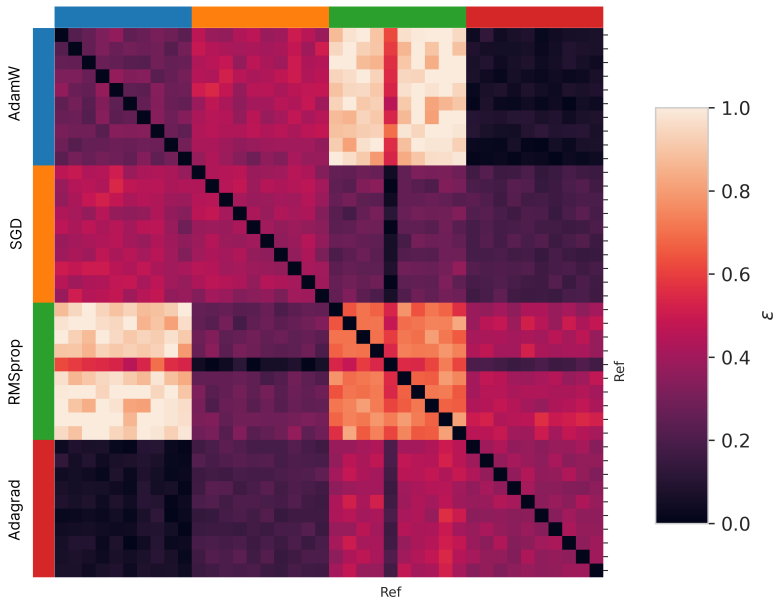
Figure 5.8: Test loss along linear interpolations of the reparameterized networks trained with different optimizers on CIFAR-10. The point-wise median value, and Q1, Q3 quartiles are highlighted in red

ϵ, the results remain qualitatively consistent, except in the case of SGD trained networks which curiously show a strong dependency. Using the pairwise epsilon matrix shown in Figure 5.9, the networks trained using RMSprop still remain the most diverse up to permutation, with low linear mode connectivity. On the other hand, the AdamW trained networks show stronger linear mode connectivity among themselves. Similar to the case of MNIST-trained networks, the reparameterized networks are more connected to the reference network than among themselves, as evident from the clear horizontal and vertical bands at the reference index.

This result further strengthens the claim that linear mode connectivity up to permutation is highly influenced by the training dynamics, and may not be as ubiquitous as thought to be. Understanding this weight space dynamics and the implicit bias of different optimizers towards the kind of minima they converge to, remains an open question for future work.



(a) Reference: SGD



(b) Reference: RMSprop

Figure 5.9: Pairwise ϵ-loss between reparameterized networks trained on CIFAR-10 with different optimizers. The colors are scaled such that the maximum value corresponds to the maximum ϵ. The reference network is highlighted on the axis

Chapter 6

Conclusions and Future work

This final chapter presents a brief summary of the report, and relates the experimental findings to the current research on linear mode connectivity and model averaging techniques. Some insights on possible future research directions are also shared, along with a discussion on the ethical implications of the work.

6.1 Summary

We introduce the intriguing phenomenon of mode connectivity in neural networks, where the loss surface around the minima is shown to not be isolated but connected to other minima through simple, low-loss curves. Subsequently, we define linear mode connectivity up to permutation when accounting for the symmetries of the fully connected network. To achieve this, we use a heuristically motivated reparameterization method recently introduced in the literature. With this weight matching method [AHS23] we align a set of networks trained on datasets such as Moons, MNIST, and CIFAR-10 to a reference, and examine the loss in the neighborhood of these aligned networks, specifically along the line connecting pairs of networks. Our work provides a crucial insight that while permutation alignment may appear unnecessary for the generalization of wide network averages, this is due to feature redundancies in the hidden layers. Therefore, weight averaging methods aiming to improve generalization [Ram+22], or to be used in federated learning settings [McM+17] can benefit significantly from aligning the networks before averaging them. Current methods that do not explicitly perform alignment [Izm+18] [Ilh+22], are shown to benefit from implicit alignment from the samples sharing their training trajectories.

We demonstrate that permutation alignment, particularly using the described weight matching method, may not be sufficient to ensure linear mode connectivity. This could be attributed to methodological issues, such as the heuristic argument that the proximity of weights in the Euclidean sense may be enough for optimal permutation alignment. Or it is possible that different optimizers may prefer converging to different functions, preventing permutation alignment alone to result in linear mode connectivity. This aligns with the current understanding in the research community that sub-optimal minima exist in the loss surface, and it is the implicit regularization of optimization that leads to well-generalizing models. The type of regularization induced by different methods remains an intriguing open question. It is also necessary to continue researching more efficient, general, and theoretically motivated reparameterization methods [Peñ+22] that can account for network symmetries. Regarding recent studies that connect linear mode connectivity to mechanistic interpretations of networks [Lub+22] [Jun+22], we argue that varying the optimization methods could serve as a valuable approach to validate these findings.

6.2 Social, ethical, sustainable aspects

This work focuses on exploring linear mode connectivity in neural networks, where multiple independently trained models can be averaged to combine their capabilities. This approach holds significant promise, particularly in the context of federated learning, where small models can be trained on edge devices without the need to transmit or store private data on a central system. These models can then be combined to harness the benefits of a larger dataset. However, it is crucial to investigate whether the data can be guaranteed to remain private or if the weights themselves could potentially lead to data leakage. Furthermore, when models are combined, it is important to ensure that they do not become increasingly biased. An in-depth understanding of the loss surface of neural networks and the training dynamics of different optimizers could lead to the development of more effective optimization methods. This, in turn, is a key step towards reducing the energy and fiscal costs associated with training large models, ultimately increasing the accessibility of research communities to study the impact of scale in networks. Such efforts contribute to the creation of more sustainable AI systems.

6.3 Future work

Among many potential directions for further research, we highlight two particular extensions of the presented work. The first is an extensive validation of the results presented in this report. This study is limited to fully connected ReLU networks trained for classification on vision datasets. However, as with most empirically backed claims, there are various other factors that are not explicitly controlled for, such as different network architectures, activation functions, and data modalities. While the choice of optimizer is shown to impact linear mode connectivity, this is based on optimal tuning of the training parameters. Exploring a wider range of hyperparameters is omitted due to limited computational constraints. Another promising research direction would be to understand how the training dynamics induced by optimization methods affect the distribution of network weights and, in turn, linear mode connectivity up to permutation. While it is possible that different optimization methods converge to different types of minima on the loss surface, it may also be possible that symmetries other than permutation dominate under some methods.

References

[AHS23] Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. *Git Re-Basin: Merging Models modulo Permutation Symmetries*. Mar. 1, 2023. DOI: 10.48550/arXiv.2209.04836. arXiv: 2209.04836[cs]. URL: http://arxiv.org/abs/2209.04836 (visited on 03/06/2023).

[Ben+21] Gregory Benton, Wesley Maddox, Sanae Lotfi, and Andrew Gordon Gordon Wilson. "Loss Surface Simplexes for Mode Connecting Volumes and Fast Ensembling." In: *Proceedings of the 38th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, July 1, 2021, pp. 769–779. URL: https://proceedings.mlr.press/v139/benton21a.html (visited on 03/06/2023).

[Ben+22] 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." In: OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop). Nov. 23, 2022. URL: https://openreview.net/forum?id=HS5zuN_qFI (visited on 03/06/2023).

[BKH16] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. "Layer normalization." In: *arXiv preprint arXiv:1607.06450* (2016).

[Cui+22] Tianyu Cui, Yogesh Kumar, Pekka Marttinen, and Samuel Kaski. "Deconfounded Representation Similarity for Comparison of Neural Networks." In: Advances in Neural Information Processing Systems. Oct. 31, 2022. URL: https://openreview.net/forum?id=mMdRZipvld2 (visited on 03/06/2023).

[Den12] Li Deng. “The mnist database of handwritten digit images for machine learning research.” In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[Dra+18] Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht. “Essentially No Barriers in Neural Network Energy Landscape.” In: *Proceedings of the 35th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, July 3, 2018, pp. 1309–1318. URL: https://proceedings.mlr.press/v80/draxler18a.html (visited on 03/06/2023).

[Eis+22] Marvin Eisenberger, Aysim Toker, Laura Leal-Taixé, Florian Bernard, and Daniel Cremers. “A Unified Framework for Implicit Sinkhorn Differentiation.” In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, pp. 509–518. URL: https://openaccess.thecvf.com/content/CVPR2022/html/Eisenberger_A_Unified_Framework_for_Implicit_Sinkhorn_Differentiation_CVPR_2022_paper.html (visited on 03/06/2023).

[Ent+22] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. “The Role of Permutation Invariance in Linear Mode Connectivity of Neural Networks.” In: International Conference on Learning Representations. Jan. 28, 2022. URL: https://openreview.net/forum?id=dNigytemkL (visited on 03/06/2023).

[FB22] C. Daniel Freeman and Joan Bruna. “Topology and Geometry of Half-Rectified Network Optimization.” In: International Conference on Learning Representations. July 21, 2022. URL: https://openreview.net/forum?id=Bk0FWVcgx (visited on 03/06/2023).

[Fra+20] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. “Linear Mode Connectivity and the Lottery Ticket Hypothesis.” In: *Proceedings of the 37th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, Nov. 21, 2020, pp. 3259–3269. URL: https://proceedings.mlr.press/v119/frankle20a.html (visited on 03/06/2023).

[Gar+18] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. "Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs." In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/be3087e74e9100d4bc4c6268cdbe8456-Abstract.html (visited on 03/06/2023).

[God+22] Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. "On the Symmetries of Deep Learning Models and their Internal Representations." In: Advances in Neural Information Processing Systems. Oct. 31, 2022. URL: https://openreview.net/forum?id=8qugS9JqAxD (visited on 03/06/2023).

[He+15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[Ilh+22] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. *Editing Models with Task Arithmetic*. Dec. 8, 2022. DOI: 10.48550/arXiv.2212.04089. arXiv: 2212.04089[cs]. URL: http://arxiv.org/abs/2212.04089 (visited on 03/06/2023).

[Izm+18] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. "Averaging weights leads to wider optima and better generalization." In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Association For Uncertainty in Artificial Intelligence (AUAI). 2018, pp. 876–885.

[JMJ98] Hannes J?nsson, Greg Mills, and Karsten W. Jacobsen. "Nudged elastic band method for finding minimum energy paths of transitions." In: *Classical and Quantum Dynamics in Condensed Phase Simulations*. WORLD SCIENTIFIC, June 1998, pp. 385–404. ISBN: 978-981-02-3498-0. DOI: 10.1142/9789812839664_0016. URL: https://www.worldscientific.co

m/doi/abs/10.1142/9789812839664_0016 (visited on 03/06/2023).

[Jun+22] Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. "Linear Connectivity Reveals Generalization Strategies." In: ICML 2022: Workshop on Spurious Correlations, Invariance and Stability. July 21, 2022. URL: https://openreview.net/forum?id=_B9Ql_poTnE (visited on 03/06/2023).

[KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images." In: (2009).

[Kor+19] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. "Similarity of Neural Network Representations Revisited." In: *Proceedings of the 36th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, May 24, 2019, pp. 3519–3529. URL: https://proceedings.mlr.press/v97/kornblith19a.html (visited on 03/06/2023).

[Kud+19] 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: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/46a4378f835dc8040c8057beb6a2da52-Abstract.html (visited on 03/06/2023).

[LH17] Ilya Loshchilov and Frank Hutter. "Decoupled weight decay regularization." In: *arXiv preprint arXiv:1711.05101* (2017).

[Li+15] Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. "Convergent Learning: Do different neural networks learn the same representations?" In: *Proceedings of the 1st International Workshop on Feature Extraction: Modern Questions and Challenges at NIPS 2015*. Feature Extraction: Modern Questions and Challenges. ISSN: 1938-7228. PMLR, Dec. 8, 2015, pp. 196–212. URL: https://proceedings.mlr.press/v44/li15convergent.html (visited on 03/06/2023).

[Lub+22] Ekdeep Singh Lubana, Eric J. Bigelow, Robert P. Dick, David Krueger, and Hidenori Tanaka. *Mechanistic Mode Connectivity*. Nov. 15, 2022. DOI: 10.48550/arXiv.2211.08422. arXiv: 2211.08422[cs]. URL: http://arxiv.org/abs/2211.08422 (visited on 03/06/2023).

[McM+17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. "Communication-efficient learning of deep networks from decentralized data." In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[Mos+22] Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. *Relative representations enable zero-shot latent space communication*. Sept. 30, 2022. DOI: 10.48550/arXiv.2209.15430. arXiv: 2209.15430[cs]. URL: http://arxiv.org/abs/2209.15430 (visited on 03/06/2023).

[MRB18] Ari Morcos, Maithra Raghu, and Samy Bengio. "Insights on representational similarity in neural networks with canonical correlation." In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/a7a3d70c6d17a73140918996d03c014f-Abstract.html (visited on 03/06/2023).

[Peñ+22] Fidel A. Guerrero Peña, Heitor Rapela Medeiros, Thomas Dubail, Masih Aminbeidokhti, Eric Granger, and Marco Pedersoli. *Re-basin via implicit Sinkhorn differentiation*. Dec. 22, 2022. DOI: 10.48550/arXiv.2212.12042. arXiv: 2212.12042[cs]. URL: http://arxiv.org/abs/2212.12042 (visited on 03/06/2023).

[Pit+22] 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: *Proceedings of the 39th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, June 28, 2022, pp. 17759–17781. URL: https://proceedings.mlr.press/v162/pittorino22a.html (visited on 03/06/2023).

[PTS20] Henning Petzka, Martin Trimmel, and Cristian Sminchisescu. "Notes on the Symmetries of 2-Layer ReLU-Networks." en. In: *Proceedings of the Northern Lights Deep Learning Workshop* 1 (Feb. 2020), pp. 6–6. ISSN: 2703-6928. DOI: 10.7557/18.5150. URL: https://septentrio.uit.no/index.php/nldl/article/view/5150 (visited on 05/21/2023).

[Rag+17] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. "SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and Interpretability." In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html (visited on 03/06/2023).

[Ram+22] Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. "Diverse weight averaging for out-of-distribution generalization." In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 10821–10836.

[Sim+21] 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: *International Conference on Machine Learning*. PMLR. 2021, pp. 9722–9732.

[SM20] Alexander Shevchenko and Marco Mondelli. "Landscape Connectivity and Dropout Stability of SGD Solutions for Overparameterized Neural Networks." In: *Proceedings of the 37th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, Nov. 21, 2020, pp. 8773–8784. URL: https://proceedings.mlr.press/v119/shevchenko20a.html (visited on 03/06/2023).

[Som+22] Gowthami Somepalli, Liam Fowl, Arpit Bansal, Ping Yeh-Chiang, Yehuda Dar, Richard Baraniuk, Micah Goldblum, and Tom Goldstein. "Can Neural Nets Learn the Same Model Twice? Investigating Reproducibility and Double Descent From the Decision Boundary Perspective." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern

Recognition. 2022, pp. 13699–13708. URL: https://openaccess.thecvf.com/content/CVPR2022/html/Somepalli_Can_Neural_Nets_Learn_the_Same_Model_Twice_Investigating_Reproducibility_CVPR_2022_paper.html (visited on 03/06/2023).

[Yun+23] David Yunis, Kumar Kshitij Patel, Pedro Henrique Pamplona Savarese, Gal Vardi, Jonathan Frankle, Matthew Walter, Karen Livescu, and Michael Maire. "On Convexity and Linear Mode Connectivity in Neural Networks." In: OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop). Jan. 10, 2023. URL: https://openreview.net/forum?id=TZQ3PKL3fPr (visited on 03/06/2023).

TRITA-EECS-EX- 2023:0000

€€€€ For DIVA €€€€

{
”Author1”: { ”Last name”: ”Kalaivanan”,
”First name”: ”Adhithyan”,
”Local User Id”: ”u1oayjiw”,
”E-mail”: ”adhkal@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
}
},
”Cycle”: ”2”,
”Course code”: ”DA233X”,
”Credits”: ”30.0”,
”Degree1”: {”Educational program”: ”Master’s Programme, Machine Learning, 120 credits”
,”programcode”: ”TMAIM”
,”Degree”: ”Master’s Programme”
,”subjectArea”: ”Machine Learning”
},
”Title”: {
”Main title”: ”On Linear Mode Connectivity up to Permutation of Hidden Neurons in Neural Networks”,
”Subtitle”: ”When does Weight Averaging work?”,
”Language”: ”eng” },
”Alternative title”: {
”Main title”: ”Anslutning i linjärt läge upp till permutation av dolda neuroner i neurala nätverk”,
”Subtitle”: ”När fungerar Viktmedelvärde?”,
”Language”: ”swe”
},
”Supervisor1”: { ”Last name”: ”Sullivan”,
”First name”: ”Josephine”,
”Local User Id”: ”u18xk10j”,
”E-mail”: ”sullivan@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Computer Science” }
},
”Examiner1”: { ”Last name”: ”Azizpour”,
”First name”: ”Hossein”,
”Local User Id”: ”u1j08ow8”,
”E-mail”: ”azizpour@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Computer Science” }
},
”National Subject Categories”: ”10201, 10207”,
”Other information”: {”Year”: ”2023”, ”Number of pages”: ”xv,55”},
”Copyrightleft”: ”copyright”,
”Series”: { ”Title of series”: ”TRITA-EECS-EX” , ”No. in series”: ”2023:0000” },
”Opponents”: { ”Name”: ”A. B. Normal & A. X. E. Normalè”},
”Presentation”: { ”Date”: ”2022-03-15 13:00”
,”Language”:”eng”
,”Room”: ”via Zoom https://kth-se.zoom.us/j/ddddddddddd”
,”Address”: ”Isafjordsgatan 22 (Kistagången 16)”
,”City”: ”Stockholm” },
”Number of lang instances”: ”2”,
”Abstract[eng]”: €€€€


```
% Write an abstract that is about 250 and 350 words (1/2 A4-page)  with the following components:
% % key parts of the abstract
% \begin{itemize}
%   \item What is the topic area? (optional) Introduces the subject area for the project.
%   \item Short problem statement
%   \item Why was this problem worth a Bachelor's/'Masters thesis project? (\ie, why is the problem
both significant and of a suitable degree of difficulty for a Bachelor's/'Masters thesis project? Why
has no one else solved it yet?)
%   \item How did you solve the problem? What was your method/insight?
%   \item Results/Conclusions/Consequences/Impact: What are your key
results/\linebreak[4]conclusions? What will others do based on your results? What can be done now
that you have finished - that could not be done before your thesis project was completed?
% \end{itemize}
Neural networks trained using gradient-based optimization methods exhibit a surprising phenomenon
known as mode connectivity, where two independently trained network weights are not isolated low loss
minima in the parameter space.
Instead, they can be connected by simple curves along which the loss remains low.
In case of linear mode connectivity up to permutation, even linear interpolations of the trained
weights incur low loss when networks that differ by permutation of their hidden neurons are
considered equivalent.
While some recent research suggest that this implies existence of a single near-convex loss basin to
which the parameters converge, others have empirically shown distinct basins corresponding to
different strategies to solve the task.
In some settings, averaging multiple network weights naively, without explicitly accounting for
permutation invariance still results in a network with improved generalization.
```

In this thesis, linear mode connectivity among a set of neural networks independently trained on labelled datasets, both naively and upon reparameterization to account for permutation invariance is studied.
Specifically, the effect of hidden layer width on the connectivity is empirically evaluated.
The experiments are conducted on a two dimensional toy classification problem, and the insights are extended to deeper networks trained on handwritten digits and images.
It is argued that accounting for permutation of hidden neurons either explicitly or implicitly is necessary for weight averaging to improve test performance.
Furthermore, the results indicate that the training dynamics induced by the optimization plays a significant role, and large model width alone may not be a sufficient condition for linear model connectivity.

€€€€,
"Keywords[eng]": €€€€
Mode Connectivity, Representation Learning, Loss Landscape, Network Symmetry €€€€,
"Abstract[swe]": €€€€

Neurala nätverk som tränats med gradientbaserade optimeringsmetoder uppvisar ett överraskande fenomen som kallas modeconnectivity, där två oberoende tränade nätverksvikter inte är isolerade lågförlustminima i parameterutrymmet.
Istället kan de kopplas samman med enkla kurvor längs vilka förlusten förblir låg.
I händelse av linjär mode-anslutning upp till permutation medför även linjära interpolationer av de tränade vikterna låga förluster när nätverk som skiljer sig åt genom permutation av deras dolda neuroner anses vara likvärdiga.
Medan en del nyare undersökningar tyder på att detta innebär att det finns en enda nära-konvex förlustbassäng till vilken parametrarna konvergerar, har andra empiriskt visat distinkta bassänger som motsvarar olika strategier för att lösa uppgiften.
I vissa inställningar resulterar ett naivt medelvärde av flera nätverksvikter, utan att uttryckligen ta hänsyn till permutationsinvarians, fortfarande i ett nätverk med förbättrad generalisering.
I den här avhandlingen studeras linjärmodsanslutningar mellan en uppsättning neurala nätverk som är oberoende tränade på märkta datamängder, både naivt och vid omparameterisering för att ta hänsyn till permutationsinvarians.
Specifikt utvärderas effekten av dold lagerbredd på anslutningen empiriskt.
Experimenten utförs på ett tvådimensionellt leksaksklassificeringsproblem, och insikterna utökas till djupare nätverk som tränas på handskrivna siffror och bilder.
Det hävdas att redogörelse för permutation av dolda neuroner antingen explicit eller implicit är nödvändigt för viktgenomsnitt för att förbättra testprestanda.
Dessutom indikerar resultaten att träningsdynamiken som induceras av optimeringen spelar en betydande roll, och enbart stor modellbredd kanske inte är ett tillräckligt villkor för linjär modellanslutning.

€€€€,
"Keywords[swe]": €€€€
Lägesanslutning, representationsinlärning, förlustlandskap, nätverkssymmetri €€€€,
}

acronyms.tex

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: t
%%% End:
% The following command is used with glossaries-extra
\setabbreviationstyle[acronym]{long-short}
% The form of the entries in this file is \newacronym{label}{acronym}{phrase}
%                                      or \newacronym[options]{label}{acronym}{phrase}
% see "User Manual for glossaries.sty" for the  details about the options, one example is shown below
% note the specification of the long form plural in the line below
% \newacronym[longplural={Debugging Information Entities}]{DIE}{DIE}{Debugging Information Entity}
% % The following example also uses options
% \newacronym[shortplural={OSes}, firstplural={operating systems (OSes)}]{OS}{OS}{operating system}
% % note the use of a non-breaking dash in long text for the following acronym
% \newacronym{IQL}{IQL}{Independent -QLearning}
% \newacronym{KTH}{KTH}{KTH Royal Institute of Technology}
% \newacronym{LAN}{LAN}{Local Area Network}
% \newacronym{VM}{VM}{virtual machine}
% % note the use of a non-breaking dash in the following acronym
% \newacronym{WiFi}{-WiFi}{Wireless Fidelity}
% \newacronym{WLAN}{WLAN}{Wireless Local Area Network}
% \newacronym{UN}{UN}{United Nations}
% \newacronym{SDG}{SDG}{Sustainable Development Goal}

\newacronym{IPBNS}{IPBNS}{I'm just a poor boy, I need no sympathy}
```