Explaining Landscape Connectivity of Low-cost Solutions for Multilayer Nets

Rohith Kuditipudi
Duke University
rohith.kuditipudi@duke.edu

Xiang Wang
Duke University
xwang@cs.duke.edu

Holden Lee
Princeton University
holdenl@princeton.edu

Yi Zhang
Princeton University
y.zhang@cs.princeton.edu

Zhiyuan Li
Princeton University
zhiyuanli@cs.princeton.edu

Wei Hu
Princeton University
huwei@cs.princeton.edu

Sanjeev Arora
Princeton University and Institute for Advanced Study
arora@cs.princeton.edu

Rong Ge
Duke University
rongge@cs.duke.edu

Abstract

Mode connectivity (Garipov et al., 2018; Draxler et al., 2018) is a surprising phenomenon in the loss landscape of deep nets. Optima—at least those discovered by gradient-based optimization—turn out to be connected by simple paths on which the loss function is almost constant. Often, these paths can be chosen to be piece-wise linear, with as few as two segments.

We give mathematical explanations for this phenomenon, assuming generic properties (such as dropout stability and noise stability) of well-trained deep nets, which have previously been identified as part of understanding the generalization properties of deep nets. Our explanation holds for realistic multilayer nets, and experiments are presented to verify the theory.

1 Introduction

Efforts to understand how and why deep learning works have led to a focus on the *optimization landscape* of training loss. Since optimization to near-zero training loss occurs for many choices of random initialization, it is clear that the landscape contains many global optima (or near-optima). However, the loss can become quite high when interpolating between found optima, suggesting that these optima occur at the bottom of "valleys" surrounded on all sides by high walls. Therefore the phenomenon of *mode connectivity* (Garipov et al., 2018; Draxler et al., 2018) came as a surprise: optima (at least the ones discovered by gradient-based optimization) are connected by simple paths in the parameter space, on which the loss function is almost constant. In other words, the optima are not walled off in separate valleys as hitherto believed. More surprisingly, the paths connecting discovered optima can be piece-wise linear with as few as two segments.

Mode connectivity begs for theoretical explanation. One paper (Freeman and Bruna, 2016) attempted such an explanation for 2-layer nets, even before the discovery of the phenomenon in multilayer nets. However, they require the width of the net to be exponential in some relevant parameters. Others (Venturi et al., 2018; Liang et al., 2018; Nguyen et al., 2018; Nguyen, 2019) require special structure in neural network where the number of neurons need to be more than the number of training data points. Thus it remains an open

1

problem to explain even the 2-layer case with realistic parameter settings, let alone standard multilayer architectures.

At first sight, finding a mathematical explanation of mode connectivity phenomenon for multilayer nets—e.g., for a 50-layer ResNet on ImageNet—appears very challenging. However, the glimmer of hope is that since the phenomenon exists for a variety of architectures and datasets, it must arise from some generic property of trained nets. The fact that the connecting paths between optima can have as few as two linear segments further bolsters this hope.

Strictly speaking, empirical findings such as in (Garipov et al., 2018; Draxler et al., 2018) do not show connectivity between *all* optima, but only for *typical* optima discovered by gradient-based optimization. It seems an open question whether connectivity holds for all optima in overparametrized nets. Section 5 answers this question, via a simple example of an overparametrized two-layer net, not all of whose optima are connected via low-cost paths.

Thus to explain mode connectivity one must seek generic properties that hold for optima obtained via gradient-based optimization on realistic data. A body of work that could be a potential source of such generic properties is the ongoing effort to understand the generalization puzzle of over-parametrized nets—specifically, to understand the "true model capacity". For example, Morcos et al. (2018) note that networks that generalize are insensitive to linear restrictions in the parameter space. Arora et al. (2018) define a *noise stability* property of deep nets, whereby adding Gaussian noise to the output of a layer is found to have minimal effect on the vector computed at subsequent layers. Such properties seem to arise in a variety of architectures purely from gradient-based optimization, without any explicit noise-injection during training—though of course using small-batch gradient estimates is an implicit source of noise-injection. (Sometimes training also explicitly injects noise, e.g. dropout or batch-normalization, but that is not needed for the noise stability to emerge.)

Since resilience to perturbations arises in a variety of architectures, such resilience counts as a "generic" property, and it is natural to prove mode-connectivity as a consequence of this property. We carry this out in the current paper. Note that our goal here is not to explain every known detail of mode connectivity, but rather to give a plausible first-cut explanation.

First, in Section 3 we explain mode connectivity by assuming the network is trained via dropout. In fact, the desired property is weaker: so long as there *exists* even a *single* dropout pattern that keeps the training loss close to optimal on the two solutions, our proof constructs a piece-wise linear path between them. The number of linear segments grows linearly with the depth of the net.

Then, in Section 4 we make a stronger assumption of noise stability along the lines of Arora et al. (2018) and show that it implies mode-connectivity using paths with 10 linear segments. While this assumption is strong, it appears to be close to what is satisfied in practice. (Of course, one could explicitly train deep nets to satisfy the needed noise stability assumption, and the theory applies directly to them.)

1.1   Related work

The landscape of the loss function for training neural networks has received a lot of attention. Dauphin et al. (2014); Choromanska et al. (2015) conjectured that local minima of multi-layer neural networks have similar loss function values, and proved the result in idealized settings. For linear networks, it is known (Kawaguchi, 2016) that all local minima are also globally optimal.

Several theoretical works explored whether a neural network has spurious valleys (non-global minima that are surrounded by other points with higher loss). Freeman and Bruna (2016) showed that for a two-layer net, if it is sufficiently overparametrized then all the local minimizers are (approximately) connected. However, in order to guarantee a small loss along the path they need the number of neurons to be exponential in the number of input dimensions. Venturi et al. (2018) proved that if the number of neurons is larger than the number of training samples or the intrinsic dimension (infinite for standard architectures), then the neural network cannot have spurious valleys. Liang et al. (2018) proved similar results for the binary classification setting. Nguyen et al. (2018); Nguyen (2019) relaxed the requirement on overparametrization, but still requires the output layer to have more direct connections than the number of training samples.

Some other papers study the existence of spurious local minima. Yun et al. (2018) showed that in most cases neural networks have spurious local minima. Note that a local minimum only need to have loss no

larger than the points in its neighborhood, so a local minimum is not necessarily a spurious valley. Safran and Shamir (2018) found spurious local minima for simple two-layer neural networks under a Gaussian input distribution. These spurious local minima are indeed spurious valleys as they have positive definite Hessian.

2 Preliminaries

Notations For a vector v, we use $\|v\|$ to denote its ℓ_2 norm. For a matrix A, we use $\|A\|$ to denote its operator norm, and $\|A\|_F$ to denote its Frobenius norm. We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We use I_n to denote the identity matrix in $\mathbb{R}^{n \times n}$. We use $O(\cdot), \Omega(\cdot)$ to hide constants and use $\widetilde{O}(\cdot), \widetilde{\Omega}(\cdot)$ to hide poly-logarithmic factors.

Neural network In most of the paper, we consider fully connected neural networks with ReLU activations. Note however that our results can also be extended to convolutional neural networks (in particular, see Remark 1 and the experiments in Section 6).

Suppose the network has d layers. Let the vector before activation at layer i be x^i, $i \in [d]$, where x^d is just the output. For convenience, we also denote the input x as x^0. Let A_i be the weight matrix at i-th layer, so that we have $x^i = A_i \phi(x^{i-1})$ for $2 \le i \le d$ and $x^1 = A_1 x^0$. For any layer i, $1 \le i \le d$, let the width of the layer be h_i. We use $[A_i]_j$ to denote the j-th column of A_i. Let the maximum width of the hidden layers be $h_{\max} := \max\{h_1, h_2, \ldots, h_{d-1}\}$ and the minimum width of the hidden layers be $h_{\min} := \min\{h_1, h_2, \ldots, h_{d-1}\}$.

We use Θ to denote the set of parameters of neural network, and in our specific model, $\Theta = \mathbb{R}^{h_1 \times h_0} \times \mathbb{R}^{h_2 \times h_1} \times \cdots \times \mathbb{R}^{h_d \times h_{d-1}}$ which consists of all the weight matrices $\{A_i\}$'s.

Throughout the paper, we use f_θ, $\theta \in \Theta$ to denote the function that is computed by the neural network. For a data set $(x, y) \sim \mathcal{D}$, the loss is defined as $L_\mathcal{D}(f_\theta) := \mathbb{E}_{(x,y) \sim \mathcal{D}}[l(y, f_\theta(x))]$ where l is a loss function. The loss function $l(y, \hat{y})$ is convex in the second parameter. We omit the distribution $\mathcal{D}$ when it is clear from the context.

Mode connectivity and spurious valleys Fixing a neural network architecture, a data set $\mathcal{D}$ and a loss function, we say two sets of parameters/solutions θ^A and θ^B are ϵ-**connected** if there is a path $\pi(t) : \mathbb{R} \to \Theta$ that is continuous with respect to t and satisfies: 1. $\pi(0) = \theta^A$; 2. $\pi(1) = \theta^B$ and 3. for any $t \in [0, 1]$, $L(f_{\pi(t)}) \le \max\{L(f_{\theta^A}), L(f_{\theta^B})\} + \epsilon$. If $\epsilon = 0$, we omit ϵ and just say they are connected.

If all the local minimizers are connected, then we say that the loss function has the **mode connectivity property**. However, as we later show in Section 5, this property is very strong and is not true even for overparametrized two-layer nets. Therefore we restrict our attention to classes of low-cost solutions that can be found by the gradient-based algorithms (in particular in Section 3 we focus on solutions that are dropout stable, and in Section 4 we focus on solutions that are noise stable). We say the loss function has ϵ-**mode connectivity property** with respect to a class of low-cost solutions $\mathcal{C}$, if any two minimizers in $\mathcal{C}$ are ϵ-connected.

Mode connectivity is closely related to the notion of spurious valleys and connected sublevel sets (Venturi et al., 2018). If a loss function has all its sublevel sets ($\{\theta : L(f_\theta) \le \lambda\}$) connected, then it has the mode connectivity property. When the network only has the mode connectivity property with respect to a class of solutions $\mathcal{C}$, as long as the class $\mathcal{C}$ contains a global minimizer, we know there are no spurious valleys in $\mathcal{C}$.

However, we emphasize that neither mode connectivity or lack of spurious valleys implies any local search algorithm can efficiently find the global minimizer. These notions only suggest that it is unlikely for local search algorithms to get completely stuck.

3 Connectivity of dropout-stable optima

In this section we show that *dropout-stable* solutions are connected. More concretely, we define a solution θ to be ϵ-dropout stable if we can remove a subset of half its neurons in each layer such that the loss remains

steady.

Definition 1. *(Dropout Stability) A solution θ is ϵ-**dropout stable** if for all i such that $1 \leq i < d$, there exists a subset of at most $\lfloor h_j/2 \rfloor$ hidden units in each of the layers j from i through $d-1$ such that after rescaling the outputs of these hidden units (or equivalently, the corresponding rows and/or columns of the relevant weight matrices) by some factor r[1] and setting the outputs of the remaining units to zero, we obtain a parameter θ_i such that $L(f_{\theta_i}) \leq L(f_\theta) + \epsilon$.*

Intuitively, if a solution is ϵ-dropout stable then it is essentially only using half of the network's capacity. We show that such solutions are connected:

Theorem 1. *Let θ^A and θ^B be two ϵ-dropout stable solutions. Then there exists a path in parameter space $\pi : [0,1] \to \Theta$ between θ^A and θ^B such that $L(f_{\pi(t)}) \leq \max\{L(f_{\theta^A}), L(f_{\theta^B})\} + \epsilon$ for $0 \leq t \leq 1$. In other words, letting $\mathcal{C}$ be the set of solutions that are ϵ-dropout stable, a ReLU network has the ϵ-mode connectivity property with respect to $\mathcal{C}$.*

Our path construction in Theorem 1 consists of two key steps. First we show that we can rescale at least half the hidden units in both θ^A and θ^B to zero via continuous paths of low loss, thus obtaining two parameters θ_1^A and θ_1^B satisfying the criteria in Definition 1.

Lemma 1. *Let θ be an ϵ-dropout stable solution and let θ_i be specified as in Definition 1 for $1 \leq i < d$. Then there exists a path in parameter space $\pi : [0,1] \to \Theta$ between θ and θ_1 such that $L(f_{\pi(t)}) \leq L(f_\theta) + \epsilon$ for $0 \leq t \leq 1$.*

Though naïvely one might expect to be able to directly connect the weights of θ and θ_1 via interpolation, such a path may incur high loss as the loss function is not convex over Θ. In our proof of Lemma 1, we rely on a much more careful construction. The construction uses two types of steps: (a) interpolate between two weights in the top layer (the loss is convex in the top layer weights); (b) if a set of neurons already have 0 output weights, then we can change their input weights arbitrarily. See Figure 1 for a path for a 3-layer network. Here we have separated the weight matrices into blocks: $A_3 = \begin{bmatrix} L_3 & | & R_3 \end{bmatrix}$, $A_2 = \begin{bmatrix} L_2 & C_2 \\ \hline D_2 & R_2 \end{bmatrix}$ and $A_1 = \begin{bmatrix} L_1 \\ \hline B_1 \end{bmatrix}$. The path consist of 6 steps alternating between type (a) and type (b). Note that for all the type (a) steps, we only update the top layer weights; for all the type (b) steps, we only change rows of a weight matrix (inputs to neurons) if the corresponding columns in the previous matrix (outputs of neurons) are already 0. In Section A we show how such a path can be generalized to any number of layers.

We then show that we can permute the hidden units of θ_1^A such that its non-zero units do not intersect with those of θ_1^B, thus allowing us two interpolate between these two parameters. This is formalized in the following lemma and the proof is deferred to supplementary material.

Lemma 2. *Let θ and and θ' be two solutions such that at least $\lceil h_i/2 \rceil$ of the units in each hidden layer have been set to zero in both. Then there exists a path in parameter space $\pi : [0,1] \to \Theta$ between θ and θ' with 8 line segments such that $L(f_{\pi(t)}) \leq \max\{L(f_\theta), L(f_{\theta'})\}$.*

Theorem 1 follows immediately from Lemma 1 and Lemma 2, as one can first connect θ^A to its dropout version θ_1^A using Lemma 1, then connect θ_1^A to dropout version θ_1^B of θ^B using Lemma 2, and finally connect θ_1^B to θ^B using Lemma 1 again.

Finally, our results can be generalized to convolutional networks if we do *channel-wise* dropout (Tompson et al., 2015; Keshari et al., 2018).

Remark 1. *For convolutional networks, a* channel-wise *dropout will randomly set entire channels to 0 and rescale the remaining channels using appropriate factor. Theorem 1 can be extended to work with channel-wise dropout on convolutional networks.*

---

[1]Note our results will also work if r is allowed to vary for each layer i.

$$
\begin{array}{clccc}
 & A_3 & A_2 & A_1 & \\
(1) & \left[\begin{array}{c|c} L_3 & R_3 \end{array}\right] & \left[\begin{array}{c|c} L_2 & C_2 \\ \hline D_2 & R_2 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array}\right] & \\
(2) & \left[\begin{array}{c|c} rL_3 & 0 \end{array}\right] & \left[\begin{array}{c|c} L_2 & C_2 \\ \hline D_2 & R_2 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array}\right] & (a) \\
(3) & \left[\begin{array}{c|c} rL_3 & 0 \end{array}\right] & \left[\begin{array}{c|c} L_2 & C_2 \\ \hline rL_2 & 0 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array}\right] & (b) \\
(4) & \left[\begin{array}{c|c} 0 & rL_3 \end{array}\right] & \left[\begin{array}{c|c} L_2 & C_2 \\ \hline rL_2 & 0 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array}\right] & (a) \\
(5) & \left[\begin{array}{c|c} 0 & rL_3 \end{array}\right] & \left[\begin{array}{c|c} rL_2 & 0 \\ \hline rL_2 & 0 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array}\right] & (b) \\
(6) & \left[\begin{array}{c|c} rL_3 & 0 \end{array}\right] & \left[\begin{array}{c|c} rL_2 & 0 \\ \hline rL_2 & 0 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array}\right] & (a) \\
(7) & \left[\begin{array}{c|c} rL_3 & 0 \end{array}\right] & \left[\begin{array}{c|c} rL_2 & 0 \\ \hline 0 & 0 \end{array}\right] & \left[\begin{array}{c} L_1 \\ \hline 0 \end{array}\right] & (b)
\end{array}
$$

Figure 1: Example path, 6 line segments from a 3-layer network to its dropout version.

4 Connectivity via noise stability

In this section, we relate mode connectivity to another notion of robustness for neural networks—noise stability. It has been observed (Morcos et al., 2018) that neural networks often perform as well even if a small amount of noise is injected into the hidden layers. This was formalized in (Arora et al., 2018), where the authors showed that noise-stable networks tend to generalize well. In this section we use a very similar notion of noise stability, and show that all noise-stable solutions can be connected as long as the network is sufficiently overparametrized.

We begin in Section 4.1 by restating the definitions of noise stability in (Arora et al., 2018) and also highlighting the key differences. In Section 6 we verify these assumptions in practice. In Section 4.2, we first prove that noise stability implies dropout stability (so Theorem 1 applies) and then show that it is in fact possible to connect noise stable neural networks via even simpler paths.

4.1 Noise stability

First we will introduce some additional notations and assumptions. In this section, we consider a finite and fixed training set S. For a network parameter θ, the empirical loss function is $L(\theta) = \frac{1}{|S|} \sum_{(x,y) \in S} l(y, f(x))$. Here the loss function $l(y, \hat{y})$ is assumed to be β-Lipschitz in $\hat{y}$: for any $\hat{y}, \hat{y}' \in \mathbb{R}^{h_d}$ and any $y \in \mathbb{R}^{h_d}$, we have $|l(y, \hat{y}) - l(y, \hat{y}')| \leq \beta \|\hat{y} - \hat{y}'\|$. The standard cross entropy loss over the softmax function is $\sqrt{2}$-Lipschitz.

For any two layers $i \leq j$, let $M^{i,j}$ be the operator for composition of these layers, such that $x^j = M^{i,j}(x^i)$. Let $J_{x^i}^{i,j}$ be the Jacobian of operator $M^{i,j}$ at input x^i. Since the activation functions are ReLU's, we know $M^{i,j}(x^i) = J_{x^i}^{i,j} x^i$.

Arora et al. (2018) used several quantities to define the noise stability. We list the definitions of these quantities below.

Definition 2 (Noise Stability Quantities). *Given sample set S, the **layer cushion** of layer i is defined as $\mu_i := \min_{x \in S} \frac{\|A_i \phi(x^{i-1})\|}{\|A_i\|_F \|\phi(x^{i-1})\|}$.*

*For any two layers $i \leq j$, the **interlayer cushion** $\mu_{i,j}$ is defined as $\mu_{i,j} = \min_{x \in S} \frac{\|J_{x^i}^{i,j} x^i\|}{\|J_{x^i}^{i,j}\| \|x^i\|}$.*

*Furthermore, for any layer i the **minimal interlayer cushion** is defined as[2] $\mu_{i \to} = \min_{i \leq j \leq d} \mu_{i,j}$.*

---

[2] Note that $J_{x^i}^{i,i} = I_{h_i}$ and $\mu_{i,i} = 1$.

The **activation contraction** c is defined as $c = \max_{x \in S, \ 1 \leq i \leq d-1} \frac{\|x^i\|}{\|\phi(x^i)\|}$.

Intuitively, these quantities measures the stability of the network for both a single layer and multiple layers. Note that this definition of the *interlayer cushion* is slightly different from the original definition in (Arora et al., 2018). Specifically, in the denominator of the definition, we replace the Frobenius norm of $J_{x^i}^{i,j}$ by its spectral norm. In the original definition, the interlayer cushion is at most $1/\sqrt{h_i}$, simply because $J_{x^i}^{i,i} = I_{h_i}$ and $\mu_{i,i} = 1/\sqrt{h_i}$. With this new definition, the interlayer cushion need not depend on the layer width h_i.

The final quantity of interest is interlayer smoothness, which measures how close the network is to its linear approximation for specific type of noise. In this work we focus on the noise generated by the dropout procedure (Algorithm 1). Let $\theta = \{A_1, A_2, ..., A_d\}$ be weights of the original network, and let $\theta^i = \{A_1, \hat{A}_2, \ldots, \hat{A}_i, A_{i+1}, \ldots, A_d\}$ be the result of applying Algorithm 1 to weight matrices from layer 2 to layer i.[3] For any input x, let $\hat{x}_i^i(t)$ and $\hat{x}_{i-1}^i(t)$ be the vector before activation at layer i using parameters $\theta t + \theta^i(1-t)$ and $\theta t + \theta^{i-1}(1-t)$ respectively.

Definition 3 (Interlayer Smoothness). *Given the scenario above, define **interlayer smoothness** ρ to be the smallest number such that with probability at least $1/2$ over the randomness in Algorithm 1 for any two layers i, j satisfying for every $2 \leq i \leq j \leq d$, $x \in S$, and $0 \leq t \leq 1$*

$$\|M^{i,j}(\hat{x}_i^i(t)) - J_{x^i}^{i,j}(\hat{x}_i^i(t))\| \leq \frac{\|\hat{x}_i^i(t) - x^i\| \|x^j\|}{\rho \|x^i\|},$$

$$\|M^{i,j}(\hat{x}_{i-1}^i(t)) - J_{x^i}^{i,j}(\hat{x}_{i-1}^i(t))\| \leq \frac{\|\hat{x}_{i-1}^i(t) - x^i\| \|x^j\|}{\rho \|x^i\|}.$$

If the network is smooth (has Lipschitz gradient), then interlayer smoothness holds as long as $\|\hat{x}_i^i(t) - x^i\|, \|\hat{x}_{i-1}^i(t) - x^i\|$ is small. The assumption here asks the network to behave smoothly in the random directions generated by randomly dropping out columns of the matrices.

Similar to (Arora et al., 2018), we have defined multiple quantities measuring the noise stability of a network, these quantities are indeed small constants as we verify experimentally in Section 6. Finally we combine all these quantities to define a single number that measures the noise stability of a network.

Definition 4 (Noise Stability). *For a network θ with layer cushion μ_i, minimal interlayer cushion $\mu_{i \to}$, activation contraction c and interlayer smoothness ρ, if the minimum width layer h_{min} is at least $\widetilde{\Omega}(1)$ wide, $\rho \geq 3d$ and $\|\phi(\hat{x}_i^i(t))\|_\infty = O(1/\sqrt{h_i}) \|\phi(\hat{x}_i^i(t))\|$ for $1 \leq i \leq d-1, 0 \leq t \leq 1$, we say the network θ is ϵ-noise stable for*

$$\epsilon = \frac{\beta c d^{3/2} \max_{x \in S}(\|f_\theta(x)\|)}{h_{\min}^{1/2} \min_{2 \leq i \leq d}(\mu_i \mu_{i \to})}.$$

The network is more robust if ϵ is small. Note that the quantity ϵ is small as long as the hidden layer width $h_{\min}$ is large compared to the noise stable parameters. Intuitively, we can think of this ϵ value as a single parameter that measures the noise stability of the network.

4.2 Noise stability implies dropout stability

We now show that noise stable local minimizers must also be dropout stable, from which it follows that noise stable local minimizers are connected. We first define the dropout procedure we will be using in Algorithm 1.

The main theorem that we prove in this section is:

Theorem 2. *Let θ^A and θ^B be two fully connected networks that are both ϵ-noise stable, there exists a path with 10 line segments in parameter space $\pi : [0,1] \to \Theta$ between θ^A and θ^B such that[4] $L(f_{\pi(t)}) \leq \max\{L(f_{\theta^A}), L(f_{\theta^B})\} + \widetilde{O}(\epsilon)$ for $0 \leq t \leq 1$.*

---

[3]Note that A_1 is excluded because dropping out columns in $\hat{A}_2$ already drops out the neurons in layer 1; dropping out columns in A_1 would drop out input coordinates, which is not necessary.

[4]Here $\widetilde{O}(\cdot)$ hides log factors on relevant factors including $|S|, d, \|x\|, 1/\epsilon$ and $h_i \|A_i\|$ for layers $i \in [d]$.

Algorithm 1 Dropout (A_i, p)

Input: Layer matrix $A_i \in \mathbb{R}^{h_i \times h_{i-1}}$, dropout probability $0 < p < 1$.
Output: Returns $\hat{A}_i \in \mathbb{R}^{h_i \times h_{i-1}}$.
1: For each $j \in [h_{i-1}]$, let δ_j be an i.i.d. Bernoulli random variable which takes the value 0 with probability p and takes the value $\frac{1}{1-p}$ with probability $(1-p)$.
2: For each $j \in [h_{i-1}]$, let $[\hat{A}_i]_j$ be $\delta_j [A_i]_j$, where $[\hat{A}_i]_j$ and $[A_i]_j$ are the j-th column of $\hat{A}_i$ and A_i respectively.

To prove the theorem, we will first show that the networks θ^A and θ^B are $\widetilde{O}(\epsilon)$-dropout stable. This is captured in the following main lemma:

Lemma 3. *Let θ be an ϵ-noise stable network, and let θ_1 be the network with weight matrices from layer 2 to layer d dropped out by Algorithm 1 with dropout probability $\widetilde{\Omega}(1/h_{min}) < p \leq \frac{3}{4}$. For any $2 \leq i \leq d$, assume $\|[A_i]_j\| = O(\sqrt{p})\|A_i\|_F$ for $1 \leq j \leq h_{i-1}$. For any $0 \leq t \leq 1$, define the network on the segment from θ to θ_1 as $\theta_t := \theta + t(\theta_1 - \theta)$. Then, with probability at least $1/4$ over the weights generated by Algorithm 1, $L(f_{\theta_t}) \leq L(f_\theta) + \widetilde{O}(\sqrt{p}\epsilon)$, for any $0 \leq t \leq 1$.*

The main difference between Lemma 3 and Lemma 1 is that we can now directly interpolate between the original network and its dropout version, which reduces the number of segments required. This is mainly because in the noise stable setting, we can prove that after dropping out the neurons, not only does the output remains stable, *every* intermediate layer also remains stable.

From Lemma 3, the proof of Theorem 2 is very similar to the proof of Theorem 1. The detailed proof is given in Section B.

The additional power of Lemma 3 also allows us to consider a smaller dropout probability. The theorem below allows us to trade dropout fraction with the energy barrier ϵ that we can prove—if the network is highly overparametrized, one can choose a small dropout probability p which allow the energy barrier ϵ to be smaller.

Theorem 3. *Suppose there exists a network θ^* with layer width h_i^* for each layer i that achieves loss $L(f_{\theta^*})$, and minimum hidden layer width $h_{min}^* = \widetilde{\Omega}(1)$. Let θ^A and θ^B be two ϵ-noise stable networks. For any dropout probability $1.5 \max_{1 \leq i \leq d-1}(h_i^*/h_i) \leq p \leq 3/4$, if for any $2 \leq i \leq d$, $1 \leq j \leq h_{i-1}$, $\|[A_i]_j\| = O(\sqrt{p})\|A_i\|_F$ then there exists a path with 13 line segments in parameter space $\pi : [0,1] \to \Theta$ between θ^A and θ^B such that $L(f_{\pi(t)}) \leq \max\{L(f_{\theta^A}) + \widetilde{O}(\sqrt{p}\epsilon), L(f_{\theta^B}) + \widetilde{O}(\sqrt{p}\epsilon), L(f_{\theta^*})\}$ for $0 \leq t \leq 1$.*

Intuitively, we prove this theorem by connecting θ^A and θ^B via the neural network θ^* with narrow hidden layers. The detailed proof is given in Section B.

5   Disconnected modes in two-layer nets

The mode connectivity property is not true for every neural network. Freeman and Bruna (2016) gave a counter-example showing that if the network is not overparametrized, then there can be different global minima of the neural network that are not connected. Venturi et al. (2018) showed that spurious valleys can exist for 2-layer ReLU nets with an arbitrary number of hidden units, but again they do not extend their result to the overparametrized setting. In this section, we show that even if a neural network is overparametrized—in the sense that there exists a network of smaller width that can achieve optimal loss—there can still be two global minimizers that are not connected.

In particular, suppose we are training a two-layer ReLU student network with h hidden units to fit a dataset generated by a ground truth two-layer ReLU teacher network with h_t hidden units such that the samples in the dataset are drawn from some input distribution and the labels computed via forward passes through the teacher network. The following theorem demonstrates that regardless of the degree to which the student network is overparametrized, we can always construct such a dataset for which global minima are not connected.

Theorem 4. *For any width h and and convex loss function $l : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ such that $l(y, \hat{y})$ is minimized when $y = \hat{y}$, there exists a dataset generated by ground-truth teacher network with two hidden units (i.e. $h_t = 2$) and one output unit such that global minimizers are not connected for a student network with h hidden units.*

Our proof is based on an explicit construction. The detailed construction is given in Section C.

6 Experiments

We now demonstrate that our assumptions and theoretical findings accurately characterize mode connectivity in practical settings. In particular, we empirically validate our claims using standard convolutional architectures—for which we treat individual filters as the hidden units and apply channel-wise dropout (see Remark 1)—trained on datasets such as CIFAR-10 and MNIST.

Training with dropout is not necessary for a network to be either dropout-stable or noise-stable. Recall that our definition of dropout-stability merely requires the existence of a particular sub-network with half the width of the original that achieves low loss. Moreover, as Theorem 3 suggests, if there exists a narrow network that achieves low loss, then we need only be able to drop out a number of filters equal to the width of the narrow network to connect local minima.
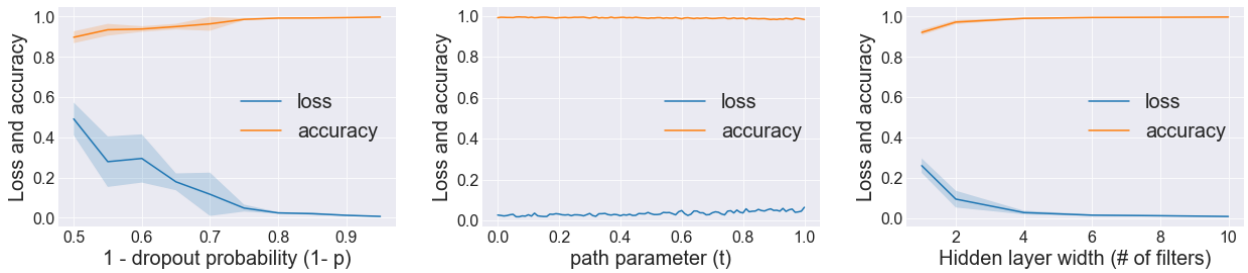

Figure 2: Results for convolutional networks trained on MNIST.

First, we demonstrate in the left plot in Figure 2, on MNIST that 3-layer convolutional nets (not counting the output layer) with 32 3×3 filters in each layer tend to be fairly dropout stable—both in the original sense of Definition 1 and especially if we relax the definition to allow for wider subnetworks—despite the fact that no dropout was applied in training. For each trial, we randomly sampled 20 dropout networks with *exactly* $\lfloor 32(1 - p) \rfloor$ non-zero filters in each layer and report the performance of the best one. In the center plot, we verify for $p = 0.2$ we can construct a linear path $\pi(t) : \mathbb{R} \to \Theta$ from our convolutional net to a dropout version of itself. Similar results were observed when varying p. Finally, in the right plot we demonstrate the existence of 3-layer convolutional nets just a few filters wide that are able to achieve low loss on MNIST. Taken together, these results indicate that our path construction in Theorem 3 performs well in practical settings.

We also demonstrate that the VGG-11 (Simonyan and Zisserman, 2014) architecture trained with channel-wise dropout (Tompson et al., 2015; Keshari et al., 2018) with $p = 0.25$ at the first three layers[5] and $p = 0.5$ at the others on CIFAR-10 converges to a noise stable minima—as measured by layer cushion, interlayer cushion, activation contraction and interlayer smoothness. The networks under investigation achieves 95% training and 91% test accuracy with channel-wise dropout *activated*, in comparison to 99% training and 92% test accuracy with dropout turned off. Figure 3 plots the distribution of the noise stability parameters over different data points in the training set. These plots suggest that they behave nicely. Interestingly, we also discovered that networks trained without channel-wise dropout exhibit similarly nice behavior on all but the first few layers. Finally, we demonstrate that the training loss and accuracy along the path described by Theorem 3 between two noise stable VGG-11 networks θ_A and θ_B remain fairly low and high respectively.

Further details on all experiments are provided in Section D.1.

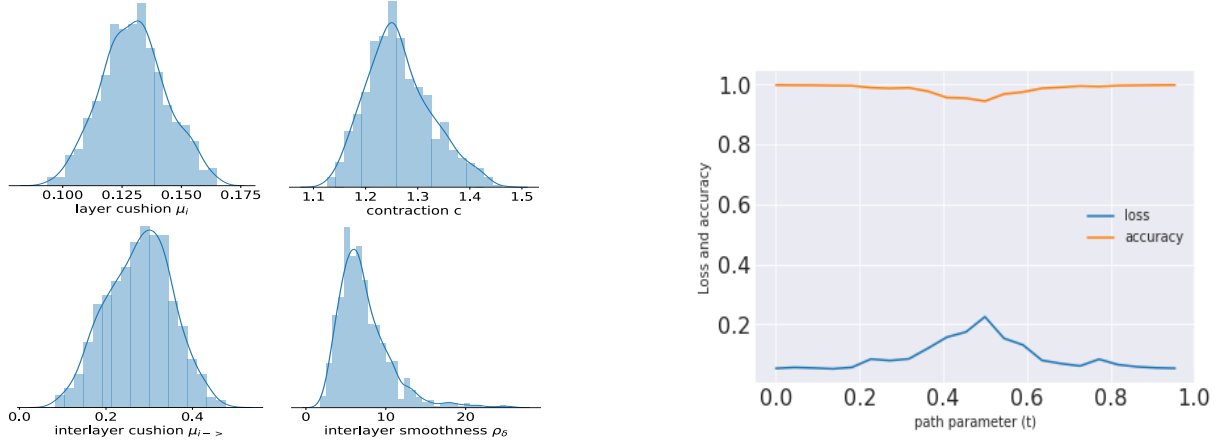[5] we find the first three layers are less resistant to channel-wise dropout.

Figure 3: Left) Distribution of layer cushion, activation contraction, interlayer cushion and interlayer smoothness of the 6-th layer of a VGG-11 network on the training set. The other layers' parameters are exhibited in Section D.2. Right) The loss and training accuracy along the interpolation path between two noise stable VGG-11 networks, described by Theorem 3.

References

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. (2018). Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2015). The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941.

Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. A. (2018). Essentially no barriers in neural network energy landscape. *arXiv preprint arXiv:1803.00885*.

Freeman, C. D. and Bruna, J. (2016). Topology and geometry of half-rectified network optimization. *arXiv preprint arXiv:1611.01540*.

Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. (2018). Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, pages 8789–8798.

Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances in neural information processing systems*, pages 586–594.

Keshari, R., Singh, R., and Vatsa, M. (2018). Guided dropout. *arXiv preprint arXiv:1812.03965*.

Liang, S., Sun, R., Li, Y., and Srikant, R. (2018). Understanding the loss surface of neural networks for binary classification. In *International Conference on Machine Learning*, pages 2840–2849.

Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., and Botvinick, M. (2018). On the importance of single directions for generalization. *arXiv preprint arXiv:1803.06959*.

Nguyen, Q. (2019). On connected sublevel sets in deep learning. *arXiv preprint arXiv:1901.07417*.

Nguyen, Q., Mukkamala, M. C., and Hein, M. (2018). On the loss landscape of a class of deep neural networks with no bad local valleys. *arXiv preprint arXiv:1809.10749.*

Safran, I. and Shamir, O. (2018). Spurious local minima are common in two-layer relu neural networks. In *International Conference on Machine Learning*, pages 4430–4438.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556.*

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., and Bregler, C. (2015). Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656.

Tropp, J. A. (2012). User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434.

Venturi, L., Bandeira, A. S., and Bruna, J. (2018). Spurious valleys in two-layer neural network optimization landscapes. *arXiv preprint arXiv:1802.06384.*

Yun, C., Sra, S., and Jadbabaie, A. (2018). A critical view of global optimality in deep learning. *arXiv preprint arXiv:1802.03487.*

A   Proofs for connectivity of dropout-stable optima

Proof of Lemma 1. Without loss of generality, suppose for each θ_i that the subset of $\lfloor h_i/2 \rfloor$ non-zero hidden units in each layer are all indexed between 1 and $\lfloor h_i/2 \rfloor$. For $1 < i < d$, we can partition A_i into quadrants such that $A_i = \left[\begin{array}{c|c} L_i & C_i \\ \hline D_i & R_i \end{array} \right]$. (Here, $L_i \in \mathbb{R}^{\lfloor h_i/2 \rfloor \times \lfloor h_i/2 \rfloor}$. If h_i is odd, when we write L_i in the other quadrants we implicitly pad it with zeros in a consistent manner.) Similarly, we can partition A_1 such that $A_1 = \left[\begin{array}{c} L_1 \\ \hline B_1 \end{array} \right]$ and A_d such that $A_d = \left[\begin{array}{c|c} L_d & R_d \end{array} \right]$. We will sometimes use the notation A_i to refer to the value of A_i at a given point on our path, while A_i^θ will always refer to the value of A_i at θ. We now proceed to prove via induction the existence of a path from θ to θ_i for all i whose loss is bounded by $L(f_\theta) + \epsilon$, from which the main result immediately follows.

Base case: from θ to θ_{d-1} As a base case of the induction, we need to construct a path from θ to θ_{d-1}, such that the loss is bounded by $L(f_\theta) + \epsilon$. First, note that setting a particular subset of columns (e.g. the right half of columns) in A_i to zero is equivalent to setting the corresponding rows (e.g. the bottom half of rows) of A_{i-1} to zero. So from the fact that $L(f_{\theta_{d-1}}) \leq L(f_\theta) + \epsilon$ it follows that we can equivalently replace A_d^θ with $\left[\begin{array}{c|c} rL_d^\theta & 0 \end{array} \right]$ without increasing our loss by more than ϵ.

In fact, because our loss function is convex over A_d we can actually interpolate A_d between A_d^θ and $\left[\begin{array}{c|c} rL_d^\theta & 0 \end{array} \right]$ while keeping our loss below $L(f_\theta) + \epsilon$ at every point along this subpath.

Then, because $R_d = 0$ we can modify both D_{d-1} and R_{d-1} any way we'd like without affecting the output of our network. In particular, we can interpolate A_{d-1} between A_{d-1}^θ and $\left[\begin{array}{c|c} L_{d-1}^\theta & C_{d-1}^\theta \\ \hline 0 & 0 \end{array} \right]$ while keeping our loss constant long this subpath, thus arriving at θ_{d-1}.

From θ_k to θ_{k-1}

Suppose we have found a path from θ to θ_k such that (1) $A_d^{\theta_k} = \left[\begin{array}{c|c} rL_d^\theta & 0 \end{array} \right]$, (2) $A_i^{\theta_k} = \left[\begin{array}{c|c} rL_i^\theta & 0 \\ \hline 0 & 0 \end{array} \right]$ for $k < i < d$, (3) $A_k^{\theta_k} = \left[\begin{array}{c|c} L_k^\theta & C_k^\theta \\ \hline 0 & 0 \end{array} \right]$, and (4) $A_i^{\theta_k} = A_i^\theta$ for $i < k$, such that the loss along the path is at most $L(f_\theta) + \epsilon$. Note that θ_{d-1} satisfies all these assumptions, including in particular (2) as there are of course no A_i between A_{d-1} and A_d. Now let us extend this path to θ_{k-1}.

First, because the rightmost columns of A_i are zero for $k < i \leq d$, we can modify the bottom rows of A_i for $k \leq i < d$ without affecting the output of our network. In particular, we can set A_k to $\left[\begin{array}{c|c} L_k^\theta & C_k^\theta \\ \hline rL_k^\theta & 0 \end{array} \right]$, as well as A_i to $\left[\begin{array}{c|c} rL_i^\theta & 0 \\ \hline 0 & rL_i^\theta \end{array} \right]$ for $k < i < d$. From the fact that the loss is convex over A_d and that $L(f_{\theta_{k-1}}) < L(f_\theta) + \epsilon$, it then follows that we can set A_d to $\left[\begin{array}{c|c} 0 & rL_d^\theta \end{array} \right]$ via interpolation while keeping our loss below $L(f_\theta) + \epsilon$. In particular, note that because the off-diagonal blocks of A_i are zero for $k < i < d$, interpolating between the leftmost columns of A_d being non-zero and the rightmost columns of A_d being non-zero simply amounts to interpolating between the outputs of the two subnetworks comprised respectively of the first $\lfloor h_i/2 \rfloor$ and last $\lfloor h_i/2 \rfloor$ rows of A_i for $k \leq i < d$.

Once we have the leftmost columns of A_d set to zero and A_i in block-diagonal form for $k < i < d$, we can proceed to modify the top rows of A_k however we'd like without affecting the output of our network. Specifically, let us set A_k to $\left[\begin{array}{c|c} rL_k^\theta & 0 \\ \hline rL_k^\theta & 0 \end{array} \right]$. We can then reset A_d to $\left[\begin{array}{c|c} rL_d^\theta & 0 \end{array} \right]$ via interpolation—this time without affecting our loss since the weights of our two subnetworks are equivalent—and afterwards set D_k to zero and R_i to zero for $k \leq i < d$—again without affecting our loss since the rightmost columns of A_d are now zero, meaning that the bottom rows of A_i have no affect on our network's output.

Following these steps, we will have $A_i = \left[\begin{array}{c|c} rL_i^\theta & 0 \\ \hline 0 & 0 \end{array} \right]$ for $k \leq i < d$ and $A_d = \left[\begin{array}{c|c} rL_d^\theta & 0 \end{array} \right]$. And so we are now free to set the bottom rows of A_{k-1} to zero without affecting our loss, thus arriving at θ_{k-1}. $\qquad\square$

Lemma 4. *Let θ be a parameter such that at least $\lceil h_i/2 \rceil$ of the units in each hidden layer have been set to zero. Then we can achieve an arbitrary permutation of the non-zero hidden units of θ via a path consisting of just 5 line segments such that our loss is constant along this path.*

Proof. Let $\pi : [h_i] \mapsto [h_i]$ be some permutation over the units in layer i. Without loss of generality, suppose all non-zero units in layer i are indexed between 0 and $\lfloor h_i/2 \rfloor$, and define $\pi' : [\lfloor h_i/2 \rfloor] \mapsto [h_i] \setminus [\lfloor h_i/2 \rfloor]$ as any one-to-one mapping such that $\pi'(i) = \pi(i)$ if $\pi(i) \in [h_i] \setminus [\lfloor h_i/2 \rfloor]$. Note that when we refer to a unit j as "set to zero", we mean that both row j of A_i and column j of A_{i+1} have been set to zero.

To permute the units of layer i, we can first simultaneously copy the non-zero rows of A_i into a subset of the rows that have been set to zero. Specifically, for $j \in [\lfloor h_i/2 \rfloor]$ we can copy row j of A_i into row $\pi'(j)$ via interpolation and without affecting our loss, due to the fact that column $\pi'(j)$ in A_{i+1} is set to zero. We can then set column j of A_{i+1} to zero while copying its value to column $\pi'(j)$, again via interpolation and without affecting our loss since rows j and $\pi'(j)$ of A_i are now equivalent.

Following these first two steps, the first $\lfloor h_i/2 \rfloor$ columns of A_{i+1} will have been set to zero. Thus, for all $j \in [\lfloor h_i/2 \rfloor]$ such that $\pi(j) \in [h_i/2]$ we can copy row $\pi'(j)$ of A_i into row $\pi(j)$ without affecting our loss. We can then set column $\pi'(j)$ of A_{i+1} to zero while copying its value into column $\pi(j)$ via interpolation and without affecting our loss since rows $\pi'(j)$ and $\pi(j)$ of A_i are now equivalent. Setting row $\pi'(j)$ to zero—again for all $j \in [\lfloor h_i/2 \rfloor]$ such that $\pi(j) \in [h_i/2]$—completes the permutation for layer i.

Note that because we leave the output of layer i unchanged throughout the course of permuting the units of layer i, it follows that we can perform all swaps across all layers simultaneously. And so from the fact that permuting each layer can be done in 5 steps—each of which consists of a single line segment in parameter space—the main result immediately follows. $\qquad\square$

Proof of Lemma 2. Without loss of generality, suppose for θ that the subset of $\lfloor h_i/2 \rfloor$ non-zero hidden units in each layer i are all indexed between 0 and $\lfloor h_i/2 \rfloor$. Note that when we refer to a unit as "set to zero", we mean that both the corresponding row of A_i and column of A_{i+1} have been set to zero. Adopting our notation in Lemma 1, we can construct a path from θ to θ' as follows.

First, from the fact that the second half of units in each hidden layer i have been set to zero in θ we have that $A_1^\theta = \left[\dfrac{L_1^\theta}{0} \right]$, $A_i^\theta = \left[\begin{array}{c|c} L_i^\theta & 0 \\ \hline 0 & 0 \end{array} \right]$ for $1 < i < d$, and $A_d^\theta = \left[\begin{array}{c|c} L_d^\theta & 0 \end{array} \right]$. Similarly, half the rows of $A_1^{\theta'}$ are zero, half the rows and columns of $A_i^{\theta'}$ are zero for $1 < i < d$, and half the columns of $A_d^{\theta'}$ are zero. Note that the indices of the non-zero units in θ' may intersect with those of the non-zero units in θ. For $1 \le i \le d$, let B_i denote the submatrix of A_i corresponding to the non-zero rows and columns of $A_i^{\theta'}$.

Because A_i^θ are block-diagonal for $1 < i < d$ and the rightmost columns of A_d^θ are zero, starting from θ we can modify the bottom rows of A_i for $1 \le i < d$ any way we'd like without affecting our loss—as done in our path construction for Lemma 1. In particular, let us set A_i to $\left[\begin{array}{c|c} L_i^\theta & 0 \\ \hline 0 & B_i^{\theta'} \end{array} \right]$ for $1 < i < d$ and A_1 to $\left[\dfrac{L_1^\theta}{B_1^{\theta'}} \right]$. Then, from the fact that our loss function is convex over A_d it follows that we can set A_d to $\left[\begin{array}{c|c} 0 & B_d^{\theta'} \end{array} \right]$ via interpolation while keeping our loss below $\max\{L(f_\theta), L(f_{\theta'})\}$. Finally, from the fact that the leftmost columns of A_d are now zero and A_i are still block-diagonal for $1 < i < d$, it follows that we can set L_i to zero for $1 \le i < d$ without affecting our loss—thus making A_i equal to $\left[\begin{array}{c|c} 0 & 0 \\ \hline 0 & B_i^{\theta'} \end{array} \right]$ for $1 < i < d$ and A_1 equal to $\left[\dfrac{0}{B_1^{\theta'}} \right]$.

To complete our path from θ to θ' we now simply need to permute the units of each hidden layer so as to return the elements of $B_i^{\theta'}$ to their original positions in A_i for each i. From Lemma 4 it follows that we can accomplish this permutation via 5 line segments in parameter space without affecting our loss. Combined with the previous steps above, we have constructed path from θ to θ' consisting of a total of 8 line segments whose loss is bounded by $\max\{L(f_\theta), L(f_{\theta'})\}$. $\qquad\square$

Proof of Theorem 1. First, from Lemma 1 we know we can construct paths from both θ^A to θ_1^A and θ^B to θ_1^B while keeping our loss below $L(f_{\theta^A}) + \epsilon$ and $L(f_{\theta^B}) + \epsilon$ respectively. From Lemma 2 we know that we

can construct a path from θ_1^A to θ_1^B such that the loss along the path is bounded by $\max\{L(f_{\theta_1^A}), L(f_{\theta_1^B})\}$. The main result then follows from the fact that $L(f_{\theta_1^A}) \leq L(f_{\theta^A}) + \epsilon$ and $L(f_{\theta_1^B}) \leq L(f_{\theta^B}) + \epsilon$ due to θ^A and θ^B both being ϵ-dropout stable. $\qquad\square$

B Proofs for connectivity via noise stability

In this section, we give detailed proofs showing that noise stability implies connectivity. In the following lemma, we first show that the network output is stable if we randomly dropout columns in a single layer using Algorithm 1.

Lemma 5. *For any layer $2 \leq i \leq d$, let $G = \{(U^{(l)}, x^{(l)})\}_{l=1}^m$ be a set of matrix/vector pairs of size m where $U \in \mathbb{R}^{h_d \times h_i}$ and $x \in \mathbb{R}^{h_{i-1}}$ satisfying $\|x\|_\infty = O\left(\frac{\|x\|}{\sqrt{h_{i-1}}}\right)$. Given A_i, let $\hat{A}_i \in \mathbb{R}^{h_i \times h_{i-1}}$ be the output of Algorithm 1 with dropout probability $0 < p \leq \frac{3}{4}$. Assume $\|[A_i]_j\| = O(\sqrt{p})\|A_i\|_F$ for $1 \leq j \leq h_{i-1}$. Given any $0 < \delta < 1$, let $\epsilon' = O\left(\sqrt{\frac{\log(mh_d/\delta)p}{h_{\min}}}\right)$, with probability at least $1 - \delta$, we have for any $(U, x) \in G$ that $\|U(\hat{A}_i - A_i)x\| \leq \epsilon'\|A_i\|_F\|U\|\|x\|$. Further assuming $h_{\min} = \Omega\left(\frac{\log(1/\delta)}{p}\right)$, we know with probability at least $1 - \delta$, no less than $\frac{2}{3}p$ fraction of columns in $\hat{A}_i$ are zero vectors.*

Intuitively, this lemma upper-bounds the change in the network output after dropping out a single layer. In the lemma, we should think of x as the input to the current layer, A_i as the layer matrix and U as the Jacobian of the network output with respect to the layer output. If the activation pattern does not change after the dropping out, $U\hat{A}_i x$ is exactly the output of the dropped out network and $\|U(\hat{A}_i - A_i)x\|$ is the change in the network output.

Proof of Lemma 5. Fixing $2 \leq i \leq d$ and one pair $(U, x) \in G$, we show with probability at least $1 - \frac{\delta}{m}$, $\|U(\hat{A}_i - A_i)x\| \leq \epsilon'\|A_i\|_F\|U\|\|x\|$. Let U_k be the k-th column of U. Then by definition of $\hat{A}_i$ in the algorithm, we know

$$U(\hat{A}_i - A_i)x = \sum_{k,j} U_k [A_i]_{kj} x_j (\delta_j - 1)$$
$$= \sum_j \left(\sum_k U_k [A_i]_{kj}\right) x_j (\delta_j - 1),$$

where δ_j is an i.i.d. Bernoulli random variable which takes the value 0 with probability p and takes the value $\frac{1}{1-p}$ with probability $(1-p)$.

Let $[A_i]_j$ be the j-th column of A_i. Because $p \leq \frac{3}{4}$, $\frac{1}{1-p} = O(1)$ (any p bounded away from 1 will work). Hence the norm for each individual term can be bounded as follows.

$$\left\|\left(\sum_k U_k [A_i]_{kj}\right) x_j (\delta_j - 1)\right\| \overset{(*)}{\leq} O\left(\frac{\|x\|}{\sqrt{h_{i-1}}}\right)\|U[A_i]_j\|$$
$$\leq O\left(\frac{\|x\|}{\sqrt{h_{\min}}}\right)\|U\|\|[A_i]_j\|$$
$$\overset{(\dagger)}{\leq} O\left(\frac{\sqrt{p}\|U\|\|A_i\|_F\|x\|}{\sqrt{h_{\min}}}\right),$$

where (*) uses the assumption that $\|x\|_\infty = O\left(\frac{\|x\|}{\sqrt{h_{i-1}}}\right)$ and (†) holds because we assume $\|[A_i]_j\| = O(\sqrt{p})\|A_i\|_F$ for $1 \leq j \leq h_{i-1}$.

For the total variance, we have

$$
\begin{aligned}
\sigma^2 :&= \sum_j \mathbb{E}\left[\left\|\left(\sum_k U_k[A_i]_{kj}\right)x_j(\delta_j-1)\right\|^2\right] \\
&\leq \sum_j \|U[A_i]_j\|^2|x_j|^2\left((0-1)^2\times p + \left(\frac{1}{1-p}-1\right)^2\times(1-p)\right) \\
&\overset{(*)}{=} \sum_j \|U[A_i]_j\|^2\cdot O\left(\frac{\|x\|^2}{h_{i-1}}\right)\cdot p\left(1+\frac{p}{1-p}\right) \\
&\leq \|UA_i\|_F^2\cdot O\left(\frac{\|x\|^2}{h_{\min}}\right)\cdot p \\
&\leq O\left(\frac{p\|U\|^2\|A_i\|_F^2\|x\|^2}{h_{\min}}\right),
\end{aligned}
$$

where inequality $(*)$ uses the assumption that $\|x\|_\infty = O\left(\frac{\|x\|}{\sqrt{h_{i-1}}}\right)$. Then, by the vector Bernstein inequality (Lemma 8), we know given $0 < \delta < 1$, there exists $\epsilon' = O\left(\sqrt{\frac{p\log(mh_d/\delta)}{h_{\min}}}\right)$, with probability at least $1-\frac{\delta}{m}$, we have

$$\|U(\hat{A}_i-A_i)x\| \leq \epsilon'\|A_i\|_F\|U\|\|x\|.$$

Taking the union bound over all (U,x) pairs in G, we know that with probability at least $1-\delta$, for any $(U,x)\in G$, $\|U(\hat{A}_i-A_i)x\|\leq \epsilon'\|A_i\|_F\|U\|\|x\|$.

Suppose $h_{\min} = \Omega\left(\frac{\log(1/\delta)}{p}\right)$; then by the Chernoff bound, we know with probability at least $1-\delta$, the dropped out fraction is at least $\frac{2}{3}p$. Taking another union bound concludes our proof. $\qquad\square$

Now we are ready to prove Lemma 3. The idea is similar to (Arora et al., 2018), but we give the proof here for completeness.

Lemma 3. *Let θ be an ϵ-noise stable network, and let θ_1 be the network with weight matrices from layer 2 to layer d dropped out by Algorithm 1 with dropout probability $\widetilde{\Omega}(1/h_{min}) < p \leq \frac{3}{4}$. For any $2\leq i\leq d$, assume $\|[A_i]_j\| = O(\sqrt{p})\|A_i\|_F$ for $1\leq j\leq h_{i-1}$. For any $0\leq t\leq 1$, define the network on the segment from θ to θ_1 as $\theta_t := \theta + t(\theta_1-\theta)$. Then, with probability at least $1/4$ over the weights generated by Algorithm 1, $L(f_{\theta_t}) \leq L(f_\theta) + \widetilde{O}(\sqrt{p}\epsilon)$, for any $0\leq t\leq 1$.*

Proof of Lemma 3. We first bound the difference between the dropped out network θ_1 and the original network θ.

Bounding $\|f_\theta(x)-f_{\theta_1}(x)\|$: We first show that with probability at least $1/2-\delta$, $\|f_\theta(x)-f_{\theta_1}(x)\| = \|x^d-\hat{x}_d^d\|\leq \epsilon'\|f_\theta(x)\|$, where ϵ' will be specified later. For any layer $i\geq 1$ and letting $\hat{x}_i^j$ be the vector before activation at layer j if the weights $A_2,\ldots,A_i$ are replaced by $\hat{A}_2,\ldots,\hat{A}_i$.

According to Lemma 5, for any layer $2\leq i\leq d$, given $0 < \delta < 1$, let $\epsilon' = O\left(\sqrt{\frac{pc^2d^2\log(mdh_d/\delta)}{h_{\min}\min\limits_{2\leq i\leq d}(\mu_i^2\mu_{i\to}^2)}}\right)$, with probability at least $1-\delta/d$ over $\hat{A}_i$, we have

$$\|U(\hat{A}_i-A_i)x\| \leq \frac{\epsilon'\mu_i\mu_{i\to}}{6cd}\|A\|_F\|U\|\|x\| \tag{1}$$

for any $(U,x)\in\{(J_{x^i}^{i,j},\phi(\hat{x}_{i-1}^{i-1}))|x\in S, i\leq j\leq d\}$. By taking a union bound over i, we know inequality (1) holds with probability at least $1-\delta$ for every i. Recall that the interlayer smoothness holds with probability

14

at least $1/2$. Taking another union, we know with probability at least $1/2 - \delta$, interlayer smoothness holds and inequality (1) holds for every $2 \le i \le d$. Next, conditioning on the success of these two events, we will inductively prove for any $1 \le i \le d$, for any $i \le j \le d$,

$$\|\hat{x}_i^j - x^j\| \le (i/d)\epsilon'\|x^j\|.$$

For the base case $i = 1$, since we are not dropping out any weight matrix, the inequality is trivial. For any $1 \le i - 1 \le d - 1$, suppose $\|\hat{x}_{i-1}^j - x^j\| \le \frac{i-1}{d}\epsilon'\|x^j\|$ for any $i - 1 \le j \le d$; we prove the induction hypothesis holds for layer i.

For any $i \le j \le d$ we have

$$\|\hat{x}_i^j - x^j\| = \|(\hat{x}_i^j - \hat{x}_{i-1}^j) + (\hat{x}_{i-1}^j - x^j)\| \le \|\hat{x}_i^j - \hat{x}_{i-1}^j\| + \|\hat{x}_{i-1}^j - x^j\|.$$

By the induction hypothesis, we know the second term can be bounded by $(i-1)\epsilon'\|x^j\|/d$. Therefore, in order to complete the induction step, it suffices to show that the first term is bounded by $\epsilon'\|x^j\|/d$. For simplicity, we also denote $\hat{x}_{i-1}^{i-1}$ as $\hat{x}^{i-1}$. Let $\Delta_i = \hat{A}_i - A_i$. We can decompose the error into two terms:

$$
\begin{aligned}
\|\hat{x}_i^j - \hat{x}_{i-1}^j\| &= \|M^{i,j}(\hat{A}_i\phi(\hat{x}^{i-1})) - M^{i,j}(A_i\phi(\hat{x}^{i-1}))\| \\
&= \|M^{i,j}(\hat{A}_i\phi(\hat{x}^{i-1})) - M^{i,j}(A_i\phi(\hat{x}^{i-1})) + J_{x^i}^{i,j}(\Delta_i\phi(\hat{x}^{i-1})) - J_{x^i}^{i,j}(\Delta_i\phi(\hat{x}^{i-1}))\| \\
&\le \|J_{x^i}^{i,j}(\Delta_i\phi(\hat{x}^{i-1}))\| + \|M^{i,j}(\hat{A}_i\phi(\hat{x}^{i-1})) - M^{i,j}(A_i\phi(\hat{x}^{i-1})) - J_{x^i}^{i,j}(\Delta_i\phi(\hat{x}^{i-1}))\| \quad (2)
\end{aligned}
$$

The first term of (2) can be bounded as follows:

$$
\begin{aligned}
&\|J_{x^i}^{i,j}\Delta_i\phi(\hat{x}^{i-1})\| \\
&\le (\epsilon'\mu_i\mu_{i\to}/6cd)\|J_{x^i}^{i,j}\|\|A_i\|_F\|\phi(\hat{x}^{i-1})\| && \text{Lemma 5} \\
&\le (\epsilon'\mu_i\mu_{i\to}/6cd)\|J_{x^i}^{i,j}\|\|A_i\|_F\|\hat{x}^{i-1}\| && \phi \text{ (ReLU) is 1-Lipschitz} \\
&\le (\epsilon'\mu_i\mu_{i\to}/3cd)\|J_{x^i}^{i,j}\|\|A_i\|_F\|x^{i-1}\| && \text{Induction hypothesis,} \\
& && \left\|\hat{x}^{i-1} - x^{i-1}\right\| \le \frac{(i-1)\epsilon'\left\|x^{i-1}\right\|}{d} < \left\|x^{i-1}\right\| \\
&\le (\epsilon'\mu_i\mu_{i\to}/3d)\|J_{x^i}^{i,j}\|\|A_i\|_F\|\phi(x^{i-1})\| && \text{Activation Contraction} \\
&\le (\epsilon'\mu_{i\to}/3d)\|J_{x^i}^{i,j}\|\|A_i\phi(x^{i-1})\| && \text{Layer Cushion} \\
&= (\epsilon'\mu_{i\to}/3d)\|J_{x^i}^{i,j}\|\|x^i\| && x^i = A_i\phi(x^{i-1}) \\
&\le (\epsilon'/3d)\|x^j\| && \text{Interlayer Cushion} \quad (3)
\end{aligned}
$$

The second term of (2) can be bounded as:

$$
\begin{aligned}
&\|M^{i,j}(\hat{A}_i\phi(\hat{x}^{i-1})) - M^{i,j}(A_i\phi(\hat{x}^{i-1})) - J_{x^i}^{i,j}(\Delta_i\phi(\hat{x}^{i-1}))\| \\
&= \|(M^{i,j} - J_{x^i}^{i,j})(\hat{A}_i\phi(\hat{x}^{i-1})) - (M^{i,j} - J_{x^i}^{i,j})(A_i\phi(\hat{x}^{i-1}))\| \\
&\le \|(M^{i,j} - J_{x^i}^{i,j})(\hat{A}_i\phi(\hat{x}^{i-1}))\| + \|(M^{i,j} - J_{x^i}^{i,j})(A_i\phi(\hat{x}^{i-1}))\|. \quad (4)
\end{aligned}
$$

Both terms of (4) can be bounded using the interlayer smoothness condition. For the second term of (4), notice that $A_i\phi(\hat{x}^{i-1}) = \hat{x}_{i-1}^i$. Thus by the induction hypothesis, we know

$$\|A_i\phi(\hat{x}^{i-1}) - x^i\| = \|\hat{x}_{i-1}^i - x^i\| \le (i-1)\epsilon'\|x^i\|/d \le \epsilon'\|x^i\|. \quad (5)$$

Now, by interlayer smoothness,

$$
\begin{aligned}
\|(M^{i,j} - J_{x^i}^{i,j})(A_i\phi(\hat{x}^{i-1}))\| &= \|(M^{i,j} - J_{x^i}^{i,j})(x^i + (A_i\phi(\hat{x}^{i-1}) - x^i))\| \\
&\le \frac{\|A_i\phi(\hat{x}^{i-1}) - x^i\|\|x^j\|}{\rho\|x^i\|} \\
&\overset{(*)}{\le} \frac{\epsilon'\|x^i\|\|x^j\|}{3d\|x^i\|} = \frac{\epsilon'\|x^j\|}{3d} \quad (6)
\end{aligned}
$$

where in (*) we use (5) and the assumption $\rho \geq 3d$. For the first term of (4), we know $\hat{A}_i\phi(\hat{x}^{i-1}) = \hat{x}^i_{i-1} + \Delta_i\phi(\hat{x}^{i-1})$. Therefore by the induction hypothesis and (3) for $i = j$,

$$\|\hat{A}_i\phi(\hat{x}^{i-1}) - x^i\| \leq \|\hat{x}^i_{i-1} - x^i\| + \|\Delta_i\phi(\hat{x}^{i-1})\| \leq (i-1)\epsilon'\|x^i\|/d + \epsilon'\|x^i\|/3d \leq \epsilon'\|x^i\|,$$

so again we have

$$\|(M^{i,j} - J^{i,j}_{x^i})(\hat{A}_i\phi(\hat{x}^{i-1}))\| \leq (\epsilon'/3d)\|x^j\|. \tag{7}$$

Together, (7) and (6) show that (4) is $\leq \frac{2\epsilon'}{3d}\|x^j\|$. Together with (3) we obtain from 2 that $\|\hat{x}^j_i - \hat{x}^j_{i-1}\| \leq \frac{\epsilon'}{d}\|x^j\|$, and hence that $\|\hat{x}^j_i - x^j\| \leq \frac{i\epsilon'\|x^j\|}{d}$, completing the induction step.

Conditioning on the success of interlayer smoothness and inequality (1), we've shown,

$$\|\hat{x}^j_i - x^j\| \leq (i/d)\epsilon'\|x^j\|,$$

for any $i \leq j \leq d$. Recall that with probability at least $1/2 - \delta$, interlayer smoothness holds and inequality (1) holds for every $2 \leq i \leq d$. Thus, let $\epsilon' = O\left(\sqrt{\frac{pc^2d^2\log(mdh_d/\delta)}{h_{\min}\min_{2\leq i\leq d}\mu_i^2\mu_{i\to}^2}}\right)$, we know with probability at least $1/2 - \delta$,

$$\|f_\theta(x) - f_{\theta_1}(x)\| = \|x^d - \hat{x}^d_d\| \leq \epsilon'\|f_\theta(x)\|.$$

Bounding $\|f_\theta(x) - f_{\theta_t}(x)\|$ for any fixed t: The proof for a fixed network on the path is almost the same as the proof for the end point. Instead of considering $\hat{x}^j_i$, now we consider $\hat{x}^j_i(t)$, which is the vector before activation at layer j if the weights $A_2, \ldots, A_i$ are replaced by $A_2 + t(\hat{A}_2 - A_2), \ldots, A_i + t(\hat{A}_i - A_i)$. We can still use Lemma 5 to bound the noise produced by replacing the weight matrix at a single layer because

$$\|U(A_i + t(\hat{A}_i - A_i) - A_i)x\| = t\|U(\hat{A}_i - A_i)x\| \leq \|U(\hat{A}_i - A_i)x\|.$$

Thus, we can still use the above induction proof to show that for any fixed $0 \leq t \leq 1$, let $\epsilon' = O\left(\sqrt{\frac{pc^2d^2\log(mdh_d/\delta)}{h_{\min}\min_{2\leq i\leq d}\mu_i^2\mu_{i\to}^2}}\right)$, with probability at least $1/2 - \delta$,

$$\|f_\theta(x) - f_{\theta_t}(x)\| \leq \epsilon'\|f_\theta(x)\|.$$

Bounding $\|f_\theta(x) - f_{\theta_t}(x)\|$ for every t: Finally, we show that $\|f_\theta(x) - f_{\theta_t}(x)\|$ is bounded for every point on the path via an ϵ'-net argument. Similar to previous steps, letting $\epsilon' = O\left(\sqrt{\frac{pc^2d^2\max_{x\in S}(\|f_\theta(x)\|^2)\log(mdh_d/\delta)}{h_{\min}\min_{2\leq i\leq d}(\mu_i^2\mu_{i\to}^2)}}\right)$, we know that with probability at least $1/2 - \delta$,

$$\|f_\theta(x) - f_{\theta_1}(x)\| \leq \epsilon'/2.$$

Next, we show that on the path, the network output is smooth in terms of the parameters. According to Algorithm 1, we know for any $2 \leq i \leq d$, we have $\|\hat{A}_i\| \leq 4\|A_i\|$, so $\|\hat{A}_i - A_i\| \leq 5\|A_i\|$. For any $2 \leq i \leq d$, let $A_{i,t} = A_i + t(\hat{A}_i - A_i)$. Note $\|A_{i,t}\| \leq (1-t)\|\hat{A}_i\| + t\|A_i\| \leq 4\|A_i\|$. For any t, t' and any $2 \leq i \leq d$, let $\theta^i_{t,t'}$ be θ_t with the weight matrix at every layer $2 \leq j \leq i$ replaced by $(A^j + t'(\hat{A}^j - A^j))$. For convenience, we also denote θ_t as $\theta^1_{t,t'}$. Given $\tau < 1/2$, for any $\tau \leq t \leq 1 - \tau$ and for any $-\tau \leq \kappa \leq \tau$, we can bound $\|f_{\theta_{t+\kappa}}(x) - f_{\theta_t}(x)\|$ as follows:

$$\|f_{\theta_{t+\kappa}}(x) - f_{\theta_t}(x)\| \leq \sum_{2\leq i\leq d} \|f_{\theta^i_{t,t+\kappa}}(x) - f_{\theta^{i-1}_{t,t+\kappa}}(x)\|$$

The output of layer $i - 1$ is the same for the two networks, of norm $\leq \|x\| \prod_{j=1}^{i-1} \|A_{j,t+\kappa}\|$. Hence the output of layer i differs by at most $\kappa \|x\| \|\hat{A}_i - A_i\| \prod_{j=1}^{i-1} \|A_{j,t+\kappa}\|$ and the output differs by $\kappa \|x\| \|\hat{A}_i - A_i\| \prod_{j=1}^{i-1} \|A_{j,t+\kappa}\| \prod_{j=i+1}^{d} \|A_{j,t}\| \leq 5^d \kappa \|x\| \prod_{j=1}^{d} \|A_j\|$. Hence

$$\|f_{\theta_{t+\alpha}}(x) - f_{\theta_t}(x)\| \leq \sum_{2 \leq i \leq d} 5^d \|x\| \kappa \prod_{1 \leq j \leq d} \|A_j\|$$

$$\leq 5^d d \kappa \|x\| \prod_{1 \leq j \leq d} \|A_i\|.$$

Thus, given $\tau \leq \frac{\epsilon'}{2 \cdot 5^d d \max_{x \in S} \|x\| \prod_{1 \leq j \leq d} \|A_j\|}$, we know for any $\tau \leq t \leq 1 - \tau$ and for any $-\tau \leq \alpha \leq \tau$,

$$\|f_{\theta_{t+\alpha}}(x) - f_\theta(x)\| \leq \epsilon'/2. \tag{8}$$

There exists a set $Q = \{\theta_t\}$ with size $O(1/\tau)$ such that for any network on the path, the distance to the closest network in Q is no more than τ. If we can prove for any $\theta_t \in Q$, $\|f_\theta(x) - f_{\theta_t}(x)\| \leq \epsilon'/2$, we immediately know for any network $\theta_{t'}$ on the path $\|f_\theta(x) - f_{\theta_{t'}}(x)\| \leq \epsilon'$ by inequality (8).

By a union bound over Q, letting $\epsilon' = O\left(\sqrt{\frac{pc^2 d^2 \max_{x \in S}(\|f_\theta(x)\|^2) \log\left(\frac{mdh_d}{\delta\tau}\right)}{h_{\min} \min_{2 \leq i \leq d}(\mu_i^2 \mu_{i\rightarrow}^2)}}\right)$, we know with probability at least $1/2 - \delta$,

$$\|f_\theta(x) - f_{\theta_t}(x)\| \leq \epsilon'/2,$$

for any $\theta_t \in Q$.

Setting $\delta = 1/4$, we know there exists

$$\epsilon' = O\left(\sqrt{\frac{pc^2 d^3 \max_{x \in S}(\|f_\theta(x)\|^2) \log\left(\frac{mdh_d \max_{x \in S} \|x\| \prod_{1 \leq j \leq d} \|A_j\|}{\epsilon'}\right)}{h_{\min} \min_{2 \leq i \leq d}(\mu_i^2 \mu_{i\rightarrow}^2)}}\right)$$

such that with probability at least $1/4$,

$$\|f_\theta(x) - f_{\theta_t}(x)\| \leq \epsilon'$$

for any $x \in S$ and any $0 \leq t \leq 1$. Since the loss function is β-Lipschitz, we further know that for any $0 \leq t \leq 1$:

$$L(f_{\theta_t}) \leq L(f_\theta) + \beta\epsilon' = L(f_\theta) + \widetilde{O}(\sqrt{p}\epsilon).$$

$\square$

Now, we are ready to prove the main theorem.

Theorem 2. *Let θ^A and θ^B be two fully connected networks that are both ϵ-noise stable, there exists a path with 10 line segments in parameter space $\pi : [0,1] \rightarrow \Theta$ between θ^A and θ^B such that[6] $L(f_{\pi(t)}) \leq \max\{L(f_{\theta^A}), L(f_{\theta^B})\} + \widetilde{O}(\epsilon)$ for $0 \leq t \leq 1$.*

Proof of Theorem 2. Setting dropout probability $p = 3/4$, by Lemma 5 and Lemma 3, if $h_{\min} = \widetilde{\Omega}(1)$, we know there exist θ_1^A and θ_1^B such that

1. in both networks, each weight matrix from layer 2 to layer d has at least half of columns as zero vectors;

2. $L(f_{\theta_t^A}) \leq L(f_{\theta^A}) + \widetilde{O}(\epsilon)$ and $L(f_{\theta_t^B}) \leq L(f_{\theta^B}) + \widetilde{O}(\epsilon)$, for any $0 \leq t \leq 1$, where $\theta_t^A = \theta^A + t(\theta_1^A - \theta^A)$ and $\theta_t^B = \theta^B + t(\theta_1^B - \theta^B)$.

---

[6]Here $\widetilde{O}(\cdot)$ hides log factors on relevant factors including $|S|, d, \|x\|, 1/\epsilon$ and $h_i \|A_i\|$ for layers $i \in [d]$.

Since the dropout fraction in both θ_1^A and θ_1^B is at least half, we can connect θ_1^A and θ_1^B as we did in Lemma 2, while ensuring the loss doesn't exceed $\max\{L(f_{\theta^A}), L(f_{\theta^B})\} + \widetilde{O}(\epsilon)$. Connecting θ^A to θ_1^A and connecting θ_B to θ_1^B each take one line segment. By the construction in Lemma 2, connecting two dropped-out networks θ_1^A and θ_1^B takes 8 line segments. Thus, overall the path between θ^A and θ^B contains 10 line segments. $\qquad\square$

Next, we show that if there exists a "narrow" neural network achiving small loss, we can get a lower energy barrier using a smaller dropout probability.

Theorem 3. *Suppose there exists a network θ^* with layer width h_i^* for each layer i that achieves loss $L(f_{\theta^*})$, and minimum hidden layer width $h_{min}^* = \widetilde{\Omega}(1)$. Let θ^A and θ^B be two ϵ-noise stable networks. For any dropout probability $1.5 \max_{1 \leq i \leq d-1}(h_i^*/h_i) \leq p \leq 3/4$, if for any $2 \leq i \leq d$, $1 \leq j \leq h_{i-1}$, $\|[A_i]_j\| = O(\sqrt{p})\|A_i\|_F$ then there exists a path with 13 line segments in parameter space $\pi : [0,1] \to \Theta$ between θ^A and θ^B such that $L(f_{\pi(t)}) \leq \max\{L(f_{\theta^A}) + \widetilde{O}(\sqrt{p}\epsilon), L(f_{\theta^B}) + \widetilde{O}(\sqrt{p}\epsilon), L(f_{\theta^*})\}$ for $0 \leq t \leq 1$.*

Proof of Theorem 3. Since $h_{min} \cdot \max_{1 \leq i \leq d-1}(h_i^*/h_i) \geq h_{min}^* = \widetilde{\Omega}(1)$, we have $h_{min} = \widetilde{\Omega}\left(\frac{1}{\max_{1 \leq i \leq d-1}(h_i^*/h_i)}\right)$. By Lemma 5 and Lemma 3, there exist θ_1^A and θ_1^B such that

1. in both networks, each weight matrix from layer 2 to layer d has at least h_i^* columns set to zero;

2. $L(f_{\theta_t^A}) \leq L(f_{\theta^A}) + \widetilde{O}(\sqrt{p}\epsilon)$ and $L(f_{\theta_t^B}) \leq L(f_{\theta^B}) + \widetilde{O}(\sqrt{p}\epsilon)$, for any $0 \leq t \leq 1$, where $\theta_t^A = \theta^A + t(\theta_1^A - \theta^A)$ and $\theta_t^B = \theta^B + t(\theta_1^B - \theta^B)$.

From the fact that at least h_i^* units in layer i of both θ_1^A and θ_1^B have been set to zero for $1 \leq i < d$—meaning that the corresponding rows of A_i and columns of A_{i+1} are zero—it follows from Lemma 2 that we can connect θ_1^A to an arbitrary permutation of θ^* using 8 segments while keeping the loss on the path no more than $\max\{L(f_{\theta_1^A}), L(f_{\theta^*})\}$. By choosing this permutation so that the non-zero units of θ^* do not intersect with those of θ_1^B, we can then connect θ^* to θ_1^B using just 3 segments as done in the first step of our path construction in Lemma 2 seeing as there is no need to permute θ^* a second time. Combining these paths together with the paths that interpolate between the original parameters θ^A and θ^B and their dropout versions θ_1^A and θ_1^B, we obtain a path in parameter space $\pi : [0,1] \to \Theta$ between θ^A and θ^B with 13 line segments such that $L(f_{\pi(t)}) \leq \max\{L(f_{\theta^A}) + \widetilde{O}(\sqrt{p}\epsilon), L(f_{\theta^B}) + \widetilde{O}(\sqrt{p}\epsilon), L(f_{\theta^*})\}$ for $0 \leq t \leq 1$. $\qquad\square$

C   Proofs for disconnected modes in two-layer nets

Proof of Theorem 4. Define our loss over parameter space such that $L(f_\theta) = \frac{1}{n}\sum l(y_i, f_\theta(\mathbf{x}_i))$, where $\mathbf{x}_i \in \mathbb{R}^{h+2}$ is our i^{th} data sample, $y_i \in \mathbb{R}$ the associated label, and $f_\theta(\mathbf{x}_i) = \mathbf{w}^T\phi(A\mathbf{x}_i)$ for $\theta = (\mathbf{w}, A) \in \mathbb{R}^{(h+2) \times h} \times \mathbb{R}^h$. We can represent the data samples as rows in a matrix $X \in \mathbb{R}^{n \times (h+2)}$—with $\mathbf{f}_i$ denoting the ith "feature" (i.e. column) of X—and the labels as elements of $\mathbf{y} \in \mathbb{R}^{n \times 1}$, as illustrated in Figure 4.

Choose k, l, m, n such that $k < l < m < n$ where $k > h$, $l - k > h$, $m - l > 2$ and $n - m > h$.

When $i \leq l$, let

$$x_{i,j} = \begin{cases} i, & j = 1 \\ i-1, & j = 2 \\ 1, & i \equiv j \pmod{h} \\ -1, & i \not\equiv j \pmod{h}, i \leq l \\ 0, & i \not\equiv j \pmod{h}, k < i \leq l. \end{cases}$$

When $l < i \leq m$, let

$$x_{i,j} = \begin{cases} -1, & j \leq 2, i \equiv j \pmod{h} \\ 0, & j \leq 2, i \not\equiv j \pmod{h} \\ 0, & j > 2, l < i \leq m. \end{cases}$$

$$X = \begin{array}{c} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \vdots \\ \mathbf{x}_k \\ \vdots \\ \mathbf{x}_l \\ \vdots \\ \vdots \\ \mathbf{x}_m \\ \vdots \\ \mathbf{x}_n \end{array} \begin{array}{cccccc} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 & \mathbf{f}_4 & \cdots & \mathbf{f}_{h+2} \\ \end{array}$$

$$\mathbf{y} = \begin{array}{c} y_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ y_l \\ \vdots \\ \vdots \\ \vdots \\ y_n \end{array} \begin{bmatrix} 1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 0 \\ \vdots \\ \vdots \\ \vdots \end{bmatrix}$$

Figure 4: Our dataset.

When $i > m$, let

$$x_{i,j} = \begin{cases} 0, & j \leq 2 \\ -1, & j > 2,\ i \equiv j \pmod h \\ 0, & j > 2,\ i \not\equiv j \pmod h. \end{cases}$$

Finally, let $y_i = 1$ when $i \leq l$ and 0 otherwise.

From the fact that $\phi(\mathbf{f}_1) - \phi(\mathbf{f}_2) = \sum_{j=3}^{h+2} \phi(\mathbf{f}_j) = \mathbf{y}$ it follows that there exist networks with both two active hidden units and h active hidden units that achieve minimal loss, with the former corresponding to the ground truth teacher network which generated our dataset.

Note in particular that the output layer weight corresponding to $\phi(\mathbf{f}_2)$ in the network with two active hidden units is negative, whereas in the network with h active hidden units the output layer weights are all positive. Thus, any path between the two networks must pass through a point in parameter space where at least one output layer weight is zero while the other $h - 1$ are positive. However, as shown in Lemma 6, there does not exist such a point in parameter space that achieves minimal loss. It follows that there exists a barrier in the loss landscape separating the original networks, both of which are global minima. Moreover, by adjusting k, l, and m we can somewhat arbitrarily raise or lower this barrier. $\qquad \square$

Lemma 6. *There does not exist a set of $h - 1$ positive weights w_i and vectors $\mathbf{h}_i \in \operatorname{span} X$ such that $\sum_{i=1}^{h-1} w_i \phi(\mathbf{h}_i) = \mathbf{y}$.*

Proof. We can think of each $\mathbf{h}_i$ as the output a particular hidden unit over all n samples in our dataset and w_i as the output layer weight associated to this hidden unit. We then have $\mathbf{h}_i = \sum a_{i,j} \mathbf{f}_j$, where the coefficients $a_{i,j}$ are elements of A.

First, if there did exist w_i and $\mathbf{h}_i$ such that $\sum_{i=1}^{h-1} w_i \phi(\mathbf{h}_i) = \mathbf{y}$, then it must be the case for all i that $\mathbf{h}_i = \sum a_{i,j} \mathbf{f}_j$ where $a_{i,j} \geq 0$ for all j. Otherwise, there would be non-zero elements in some $\mathbf{h}_i$ between indexes $l + 1$ and n that would be impossible to eliminate in $\sum_{i=1}^{h-1} w_i \phi(\mathbf{h}_i)$ given that $w_i > 0$ for all i.

Second, any linear combination of $\mathbf{f}_1$ and $\mathbf{f}_2$ with positive coefficients would result in a vector whose first l elements are positive and increasing. In contrast, the first l elements of Y are constant. And so from the fact that there does not exist $a_{i,j} > 0$ such that the first l elements of $\sum a_{i,j}\mathbf{f}_j$ are decreasing—in particular because the first k elements and next $l - k$ elements of $\sum_{j=3}^{h+2} a_{ij}x_j$ are periodic with length h—it follows that $a_{i,1}, a_{i,2} = 0$ for all $\mathbf{h}_i$.

Thus, we need only consider linear combinations of $\mathbf{f}_3$ through $\mathbf{f}_{h+2}$ with positive coefficients as candidates for $\mathbf{h}_i$. To this end, note that if a particular $\mathbf{f}_j$ has zero coefficient in all of $\mathbf{h}_1$ through $\mathbf{h}_{h-1}$, then $\sum_{i=1}^{h-1} w_i \phi(\mathbf{h}_i)$ will have zeros in every index congruent to $j \mod h$ and therefore cannot equal $\mathbf{y}$. Hence by the pigeonhole principle, in order to have $\sum_{i=1}^{h-1} w_i \phi(\mathbf{h}_i) = \mathbf{y}$ there must be some i such that $\mathbf{h}_i = \sum_{j=3}^{h+2} a_{i,j}\mathbf{f}_j$ with at least two coefficients being non-zero. However, in any linear combination $\sum_{j=3}^{h+2} a_{i,j}\mathbf{f}_j$ where $a_{i,j}, a_{i,j'} > 0$ for at least two distinct j, j', the elements in indexes $k + 1$ to l will be greater than the elements in indexes 1 to k that are congruent to $j \mod h$ and $j' \mod h$. In contrast, the first l elements of $\mathbf{y}$ are constant. Hence, similar to the case of $\mathbf{f}_1$ and $\mathbf{f}_2$, there cannot exist $\mathbf{h}_i = \sum_{j=3}^{h+2} a_{i,j}\mathbf{f}_j$ and positive coefficients w_i such that $\sum_{i=1}^{h-1} w_i \phi(\mathbf{h}_i) = Y$. $\qquad\square$

D   Experimental details and further results

## D.1   Experimental details and hyperparameters

For all experiments on MNIST, we used a convolutional architecture consisting of 3 convolutional layers followed by a fully-connected output layer. Each convolutional layer consisted of 32 3×3 filters and used sufficient padding so as to keep the layer's output the same shape as its input. All networks were trained on an NVIDIA Tesla K20c GPU for 5000 iterations with a batch size of 64 using stochastic gradient descent with an initial learning rate of 0.1 and a decay rate of $1\mathrm{E}^{-6}$. No significant hyperparameter tuning was applied.

For the left and right plots in Figure 2, we report results averaged over 5 random trials and error bars corresponding to the standard deviation over these trials. For the center plot we simply computed the loss and accuracy over a linear path between a particular convolutional net and a single dropout version of itself. Specific to Figure 2, in applying dropout with probability p we randomly sample a subset of $\lfloor 32(1-p) \rfloor$ units and rescale these units by $1/(1-p)$ while setting the remaining units to zero. In the left plot, each trial consisted of sampling 20 such dropout networks and reporting the performance of the network achieving the lowest loss. Losses and accuracies in all plots were computed on a random batch of 4096 training images.

On CIFAR-10, we trained VGG-11 networks on an NVIDIA Titan X GPU for 300 epochs with SGD with a batch size of 128, with weight decay 5e-4, momentum 0.9, and an initial learning rate of 0.05 which is decayed by factor of 2 every 30 epochs. We used channel-wise dropout at all convolutional layers. The dropout rates are $p = 0.25$ at the first three layers and are $p = 0.5$ at the others. Ordinary dropout with $p = 0.5$ is used at every fully-connected layers except for the last one (the softmax layer).

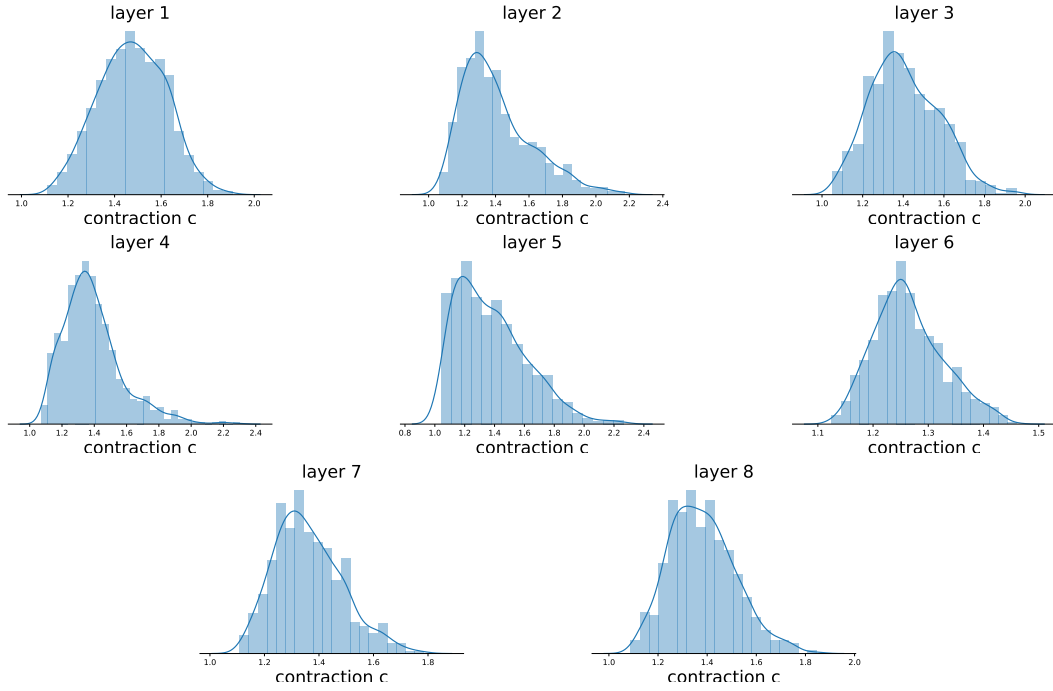## D.2 Verification of noise stability conditions
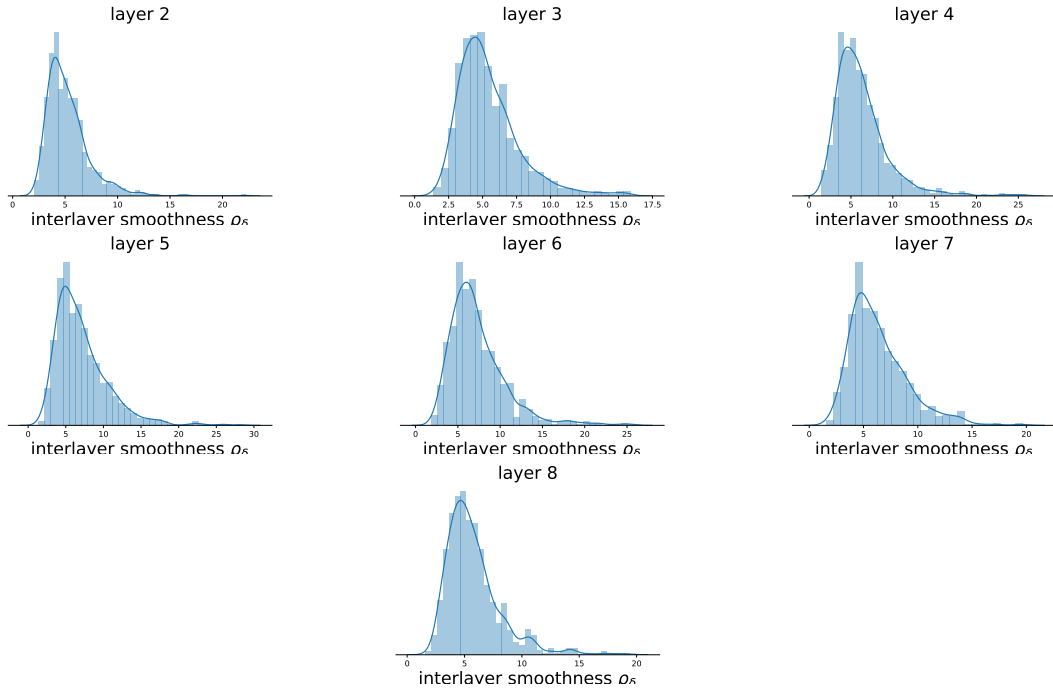
D.2.1 Layer cushion
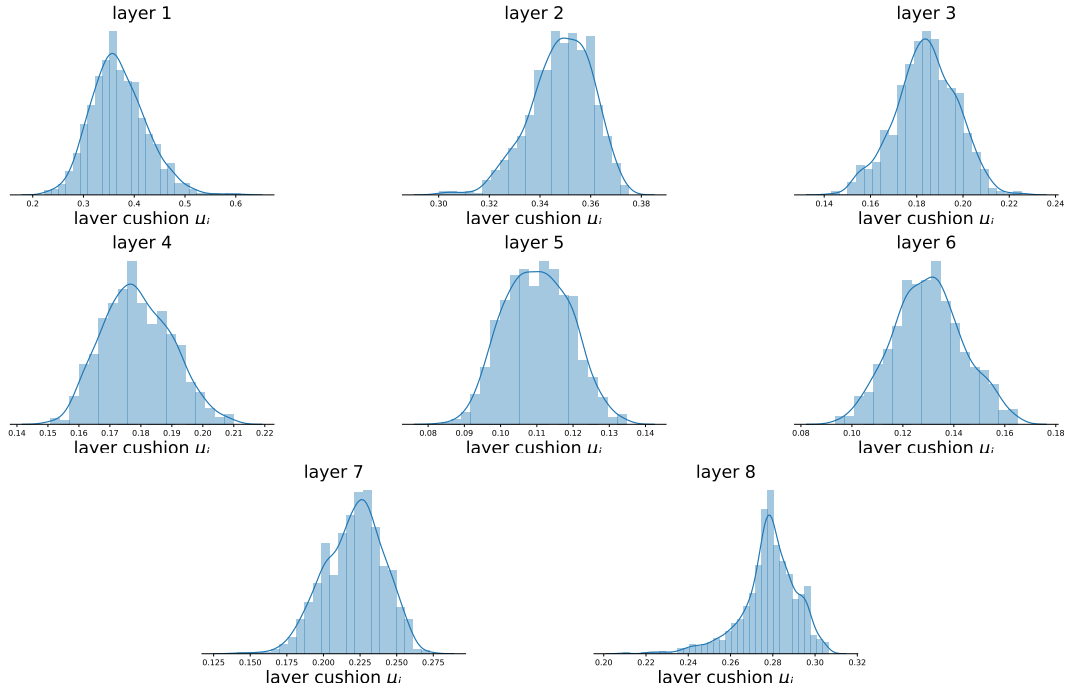
D.2.2 Interlayer cushion
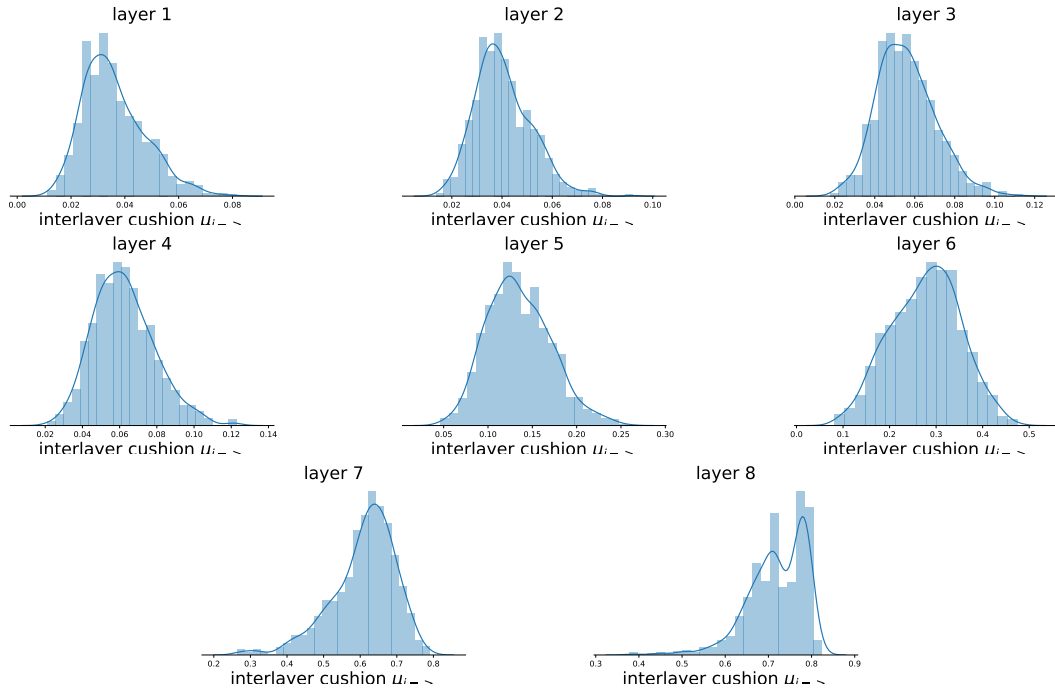
D.2.3 Activation contraction

D.2.4 Interlayer smoothness

E Tools

We use matrix concentration bounds to bound the noise produced by dropping out one single layer (Lemma 5).

Lemma 7 (Matrix Bernstein; Theorem 1.6 in (Tropp, 2012))**.** *Consider a finite sequence $\{Z_k\}$ of independent, random matrices with dimension $d_1 \times d_2$. Assume that each random matrix satisfies*

$$\mathbb{E}[Z_k] = 0 \text{ and } \|Z_k\| \leq R \text{ almost surely.}$$

Define

$$\sigma^2 := \max\left\{\big\|\sum_k \mathbb{E}[Z_k Z_k^*]\big\|, \big\|\sum_k \mathbb{E}[Z_k^* Z_k]\big\|\right\}.$$

Then, for all $t \geq 0$,

$$\Pr\left\{\big\|\sum_k Z_k\big\| \geq t\right\} \leq (d_1 + d_2) \exp\left(\frac{-t^2/2}{\sigma^2 + Rt/3}\right).$$

As a corollary, we have:

Lemma 8 (Bernstein Inequality: Vector Case)**.** *Consider a finite sequence $\{v_k\}$ of independent, random vectors with dimension d. Assume that each random vector satisfies*

$$\|v_k - \mathbb{E}[v_k]\| \leq R \text{ almost surely.}$$

Define

$$\sigma^2 := \sum_k \mathbb{E}\big[\|v_k - \mathbb{E}[v_k]\|^2\big].$$

Then, for all $t \geq 0$,

$$\Pr\left\{\big\|\sum_k (v_k - \mathbb{E}[v_k])\big\| \geq t\right\} \leq (d + 1) \cdot \exp\left(\frac{-t^2/2}{\sigma^2 + Rt/3}\right).$$