Analyzing Monotonic Linear Interpolation in Neural Network Loss Landscapes

James Lucas [1 2] **Juhan Bae** [1 2] **Michael R. Zhang** [1 2] **Stanislav Fort** [3] **Richard Zemel** [1 2] **Roger Grosse** [1 2]

Abstract

Linear interpolation between initial neural network parameters and converged parameters after training with stochastic gradient descent (SGD) typically leads to a monotonic decrease in the training objective. This Monotonic Linear Interpolation (MLI) property, first observed by Goodfellow et al. (2014), persists in spite of the non-convex objectives and highly non-linear training dynamics of neural networks. Extending this work, we evaluate several hypotheses for this property that, to our knowledge, have not yet been explored. Using tools from differential geometry, we draw connections between the interpolated paths in function space and the monotonicity of the network — providing sufficient conditions for the MLI property under mean squared error. While the MLI property holds under various settings (e.g. network architectures and learning problems), we show in practice that networks violating the MLI property can be produced systematically, by encouraging the weights to move far from initialization. The MLI property raises important questions about the loss landscape geometry of neural networks and highlights the need to further study their global properties.

1. Introduction

A simple and lightweight method to probe neural network loss landscapes is to linearly interpolate between the parameters at initialization and the parameters found after training. More formally, consider a neural network with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ trained with respect to loss function $\mathcal{L}\colon \mathbb{R}^d \to \mathbb{R}$ on a dataset $\mathcal{D}$. Let the neural network be initialized with some parameters $\boldsymbol{\theta}_0$. Then, using a gradient descent optimizer, the network converges to some final parameters $\boldsymbol{\theta}_T$. A linear path is then constructed between these two parameters denoted $\boldsymbol{\theta}_\alpha = (1-\alpha)\boldsymbol{\theta}_0 + \alpha\boldsymbol{\theta}_T$. A surprising phenomenon, first observed by Goodfellow et al. (2014), is that the function $\mathcal{L}(\boldsymbol{\theta}_\alpha)$ typically monotonically decreases on the interval $\alpha \in [0, 1]$. We call this effect the *Monotonic Linear Interpolation (MLI) property* of neural networks.

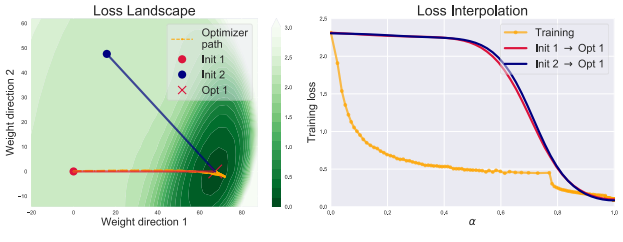

Figure 1. Monotonic linear interpolation for a ResNet-20 trained on CIFAR-10 from initialization to an optimum (**red**) and from an unrelated initialization to the same optimum (**blue**). On the left, we show a 2D slice of the loss landscape, defined by the two initializations and optimum, along with the optimization trajectory projected onto the plane (**orange**). On the right, we show the interpolated loss curves, with training loss shown relative to the proportion of distance travelled to the optimum.

The MLI property is illustrated in Figure 1. The interpolated path ($\boldsymbol{\theta}_\alpha$) exhibits the MLI property as the training loss monotonically decreases along this line. Even more surprising, linear interpolation between an unrelated random initialization and the same converged parameters also satisfies the MLI property.

Goodfellow et al. (2014) showed that the MLI property persists on various architectures, activation functions, and training objectives in neural network training. They conclude their study by stating that "the reason for the success of SGD on a wide variety of tasks is now clear: these tasks are relatively easy to optimize." In our work, we observe that networks violating the MLI property can be produced systematically and are also trained without significant difficulty. Moreover, since the publication of their research, there have been significant developments both in terms of the neural network architectures that we train today (He et al., 2016; Vaswani et al., 2017; Huang et al., 2017) and our theoretical understanding of them (Amari et al., 2020; Jacot et al., 2018; Draxler et al., 2018; Frankle & Carbin, 2018; Fort & Ganguli, 2019). Hence, with a wider lens that addresses these developments, we believe that further investigation of this phenomenon is likely to yield new insights into neural network optimization and their loss landscapes.

[1]University of Toronto [2]Vector Institute [3]Stanford University. Correspondence to: James Lucas <jlucas@cs.toronto.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

We study three distinct questions surrounding the MLI property. 1) How persistent is the MLI property? 2) Why does the MLI property hold? 3) What does the MLI property tell us about the loss landscape of neural networks? To address these questions, we provide an expanded empirical and theoretical study of this phenomenon.

To evaluate the persistence of the MLI property, we train neural networks with varying architectures, optimizers, datasets, initialization methods, objectives, and training mechanisms (e.g. batch normalization (Ioffe & Szegedy, 2015)). We find that the MLI property persists for the majority of these settings but can be consistently broken through mechanisms that encourage the weights to move far from initialization. As far as we know, ours is the first work to observe that MLI is not a stable property of the network architecture.

One hypothesis for the MLI property is that the networks are close to linear along the interpolation path. We formalize this notion using tools from differential geometry and provide sufficient conditions for neural networks trained under the MSE loss to satisfy the MLI property. In particular, we prove that if the length under the Gauss map (which we refer to as the *Gauss length*) of the interpolation trajectory in function space is small, then the network is guaranteed to have the MLI property. While the converse does not hold in general, we show that this quantity is correlated with monotonicity in practice. We connect this explanation to our prior observation that large distances moved in weight space encourage non-monotonic interpolations through a surprising power-law relationship between the distance moved and the average Gauss length.

Finally, we investigate the loss landscape of the neural networks we trained by evaluating the MLI property over alternative linear paths. For example, we examine the interpolation path connecting different initializations and final parameters (as in Figure 1). Surprisingly, when the MLI property holds for an initialization $\rightarrow$ final solution pair, the MLI property also holds for unrelated initializations to the same solution.

In summary, our primary contributions include:

- We prove a sufficient condition for neural networks minimizing MSE to satisfy the MLI property.
- We show that the MLI property does not always hold and that we can systematically control for/against it.
- We identify several common training mechanisms that provide this control and connect them to our novel theoretical results.
- We provide a novel insight into the neural networks' landscape through our analysis of the MLI property.

2. Related Work

Monotonic linear interpolation. Goodfellow et al. (2014) were the first to observe that the MLI property persists on various architectures, activation functions, and training objectives in deep learning. In addition to their empirical evaluation, they provided a qualitative analysis of the MLI property in a toy model where they argued that the MLI property holds despite negative curvature about initialization and disconnected optima. Concurrent research (Frankle, 2020) extends the original work of Goodfellow et al. (2014) with evaluations on modern architectures trained with SGD.

Im et al. (2016) provided an empirical investigation of the loss landscape of neural networks via low dimensional projections, including those from initialization to converged solution. Similar to our work, they investigated the effect that varying optimizers and the use of batch normalization (Ioffe & Szegedy, 2015) has on the qualitative properties of the explored loss landscape.

In this work, we provide an expanded study of the MLI property. We first investigate the persistence of the MLI property on various tasks, including settings with modern architectures and techniques that were not invented at the time of the original investigation. Further, we show that despite the original work's claim, we can train networks that violate the MLI property without significant training difficulty. Our experiments yield new insights into neural networks' loss landscapes and uncover aspects of neural network training that correlate with the MLI property.

Linear connectivity. This work is connected to empirical and theoretical advancements in understanding the loss landscape of neural networks. Much of this recent work has involved characterizing mode connectivity of neural networks. In general, linear paths between modes cross regions of high loss (Goodfellow et al., 2014). However, Freeman & Bruna (2016); Garipov et al. (2018); Draxler et al. (2018) show that local minima found by stochastic gradient descent (SGD) can be connected via piecewise linear paths. Moreover, Freeman & Bruna (2016) show that these paths are typically of low curvature. Frankle et al. (2019) showed that linearly connected solutions may be found if networks share the same initialization. Fort et al. (2020) demonstrate the connection between linear connectivity and the advantage nonlinear networks enjoy over their linearized version. Kuditipudi et al. (2019) posit *dropout stability* as one possible explanation for mode connectivity, with Shevchenko & Mondelli (2019) extending these result to show that the loss landscape becomes increasingly connected and more dropout stable with increasing network depth. Finally, Nguyen (2019) shows that every sublevel set of an overparameterized network is connected, implying that all global minima are connected.

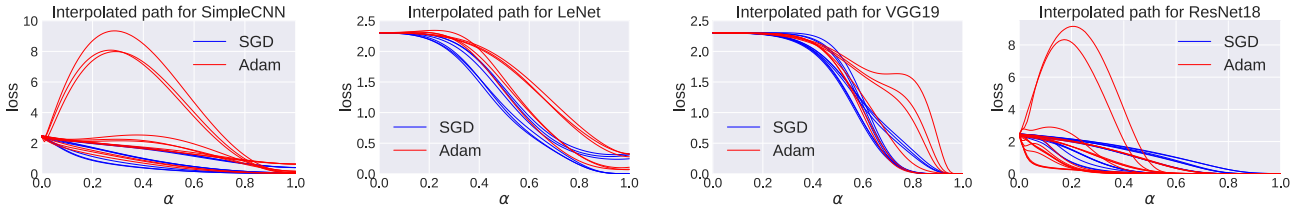

Figure 2. Training loss over the linear interpolation connecting initial and final parameters. Each curve represents a network trained on CIFAR-10 with different hyperparameter configurations (achieving at least 1.0 training loss). The MLI property holds for networks trained with SGD, but often fails for networks trained with Adam.

Note that the MLI property we study is distinct from mode connectivity, where paths are drawn between different final solutions instead of initialization → solution pairs. As far as we are aware, no prior work has explored connections between the MLI property and mode connectivity. This would make for exciting future work.

Loss landscape geometry. Recent analysis argues that there exists a small subspace at initialization in which the network converges (Gur-Ari et al., 2018; Fort & Ganguli, 2019; Papyan, 2020). Li et al. (2018) show that some of these spaces can be identified by learning in a random affine subspace of low dimension. Fort & Scherlis (2019) show that the success of these random spaces is related to the *Goldilocks zone* that depends on the Hessian at initialization. In a loose sense, the MLI can be considered a special case of these results, wherein a 1D space is sufficient for training to succeed. However, this is not the only mechanism in which neural network training can succeed — the solutions that violate the MLI property can have good generalization capability and are found without difficulty.

Venturi et al. (2018) provide necessary and sufficient conditions for the absence of spurious valleys in the loss landscapes of single hidden layer networks with 1D outputs. In particular, they show that as width grows a linear path from any point can get close to the set of global minima (but not necessarily the solution found by gradient descent). Interestingly, they also prove that for some worst-case data distributions and architectures there must always exist a barrier on any descent path.

It has long been argued that flatter minima lead to better generalization (Hochreiter & Schmidhuber, 1997a) with some caveats (Dinh et al., 2017). Recent work has shown that (full-batch) gradient descent with a large learning rate is able to find flatter minima by overcoming regions of initial high curvature (Lewkowycz et al., 2020). Intuitively, gradient descent breaks out of one locally convex region of the space and into another — suggesting that a barrier in the loss landscape has been surpassed. In this paper, we show that training with larger learning rates can lead to failure of the MLI property. And in doing so, identify a high loss barrier between the initial and converged parameters. Moreover, we show that these barriers do not appear when training with smaller learning rates.

Neural tangent kernel. Recent research has shown that over-parameterized networks appreciate faster and, in some cases, more linear learning dynamics (Lee et al., 2019; Matthews et al., 2018). The Neural Tangent Kernel (NTK) (Jacot et al., 2018) describes the learning dynamics of neural networks in their function space. Existing work argues that the NTK is near-constant in the infinite width setting (Sun, 2019), however recent work challenges this view in general (Liu et al., 2020). Fort et al. (2020) recently showed that the NTK evolves quickly early on during training but the rate of change decreases dramatically during training. In Appendix D.1, we draw connections between the NTK literature and the MLI property and show that sufficiently wide fully-connected networks exhibit the MLI property with high probability.

Optimization algorithms. In this work, we investigate the role that optimization algorithms have on the MLI property (and thus the explored loss landscape more generally). Amari et al. (2020) recently showed that for linear regression, natural gradient descent (Amari, 1998) travels further in parameter space, as measured by Euclidean distance, compared to gradient descent. We verify this claim empirically for larger networks trained with adaptive optimizers and observe that this co-occurs with non-monotonicity along the interpolating path $\boldsymbol{\theta}_\alpha$.

3. The Monotonic Linear Interpolation Property

The Monotonic Linear Interpolation (MLI) property states that when a network is randomly initialized and then trained to convergence, the linear path connecting the initialization and converged solution is monotonically decreasing in the training loss. Specifically, we say that a network has the MLI property if, for all $\alpha_1, \alpha_2 \in [0, 1]$ with $\alpha_1 < \alpha_2$,

$$\mathcal{L}(\boldsymbol{\theta}_{\alpha_1}) \geq \mathcal{L}(\boldsymbol{\theta}_{\alpha_2}), \text{ where } \boldsymbol{\theta}_\alpha = \boldsymbol{\theta}_0 + \alpha(\boldsymbol{\theta}_T - \boldsymbol{\theta}_0). \quad (1)$$

Here, $\boldsymbol{\theta}_0$ denotes the parameters at initialization and $\boldsymbol{\theta}_T$ denotes the parameters at convergence.

3.1. Δ-Monotonicity

Goodfellow et al. (2014) found that the MLI property holds for a wide range of neural network architectures and learning problems. They provided primarily qualitative evidence of this fact by plotting $\mathcal{L}(\boldsymbol{\theta}_\alpha)$ with discretizations of $[0, 1]$ using varying resolutions. We instead propose a simple quantitative measure of non-monotonicity.

Definition 1. *(Δ-monotonicity) Consider a linear parameter interpolation parameterized by $\alpha, \boldsymbol{\theta}_0$, and $\boldsymbol{\theta}_T$ with corresponding loss function $\mathcal{L}$. The path is Δ-monotonic for $\Delta \geq 0$ if for all $\alpha_1, \alpha_2 \in [0, 1]$ with $\alpha_1 < \alpha_2$, we have $\mathcal{L}(\alpha_2) - \mathcal{L}(\alpha_1) < \Delta$.*

Intuitively, the above definition states that any bump due to increasing loss over the interpolation path should have a height upper-bounded by Δ. We are interested in the smallest $\Delta \geq 0$ for which this definition holds. Notably, this minimum Δ can be approximated well numerically by stepping along the interpolation path in fixed intervals to find α_1 and α_2 giving the largest positive gap $\mathcal{L}(\alpha_2) - \mathcal{L}(\alpha_1)$.

3.2. Weight-space perspective

It is natural to attempt to reason about the MLI property in terms of the parameters of the neural network. Intuitively, the MLI property suggests that, during optimization, the parameters move into a nearby basin of low loss without encountering any high-loss barriers in their path.

We can formalize this intuition for "Lazy Training" (Chizat et al., 2018), where the weights find a minimum near their initial value. Consider the second-order Taylor series expansion about the converged minimum $\boldsymbol{\theta}^*$,

$$\mathcal{L}(\boldsymbol{\theta}_0) \approx \mathcal{L}(\boldsymbol{\theta}^*) + (\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*)^\top \nabla^2_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}^*)(\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*). \quad (2)$$

Note that the first-order Taylor expansion term does not appear as $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}^*) = 0$. If the difference between the initial and converged parameters, $\|\boldsymbol{\theta}_0 - \boldsymbol{\theta}^*\|$, is sufficiently small, then this quadratic approximation holds well throughout the linear interpolation. In this case, the linear interpolation yields a monotonic decrease in the loss (Lemma 7, Appendix D).

Experimentally, we investigate the connection between the distance moved in weight space and the monotonicity of the resulting interpolation. We find that networks that move further in weight space during training are significantly more likely to produce non-monotonic initialization→optimum interpolations. Theoretically, we investigate the MLI property for wide neural networks where lazy training occurs provably (Lee et al., 2019). In this setting, we prove that the MLI property holds with high probability for networks of sufficient width (Theorem 8, Appendix D).

3.3. Function-space perspective

We typically train neural networks with a convex loss function applied to the network's output. While the parameter space of neural networks is extremely high-dimensional and exhibits symmetries, the function space is generally simpler and easier to reason about (Jacot et al., 2018). To that end, we let

$$\mathbf{z}(\alpha; \mathbf{x}) = f(\mathbf{x}; \boldsymbol{\theta}_\alpha) \in \mathbb{R}^k, \quad \alpha \in [0, 1] \quad (3)$$

denote the *logit interpolation* of a neural network f evaluated on data point $\mathbf{x}$ with parameters $\boldsymbol{\theta}_\alpha = \boldsymbol{\theta}_0 + \alpha(\boldsymbol{\theta}_T - \boldsymbol{\theta}_0)$.

One special case that guarantees the MLI property is that of linear functions, $f(\mathbf{x}; \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{x}$ (with $\mathcal{L}(\boldsymbol{\theta}_0) > \mathcal{L}(\boldsymbol{\theta}_T)$). In this case, the logit interpolations are also linear and, under a convex loss function, f will satisfy the MLI property (Boyd et al., 2004). In practice, we work with non-linear neural networks that have non-linear logit interpolations. However, we observed that the logit interpolations are often close to linear (in a sense that we formalize soon) and that this coincides with the MLI property (Figure 3). Therefore, we raise the question: Can we guarantee the MLI property for logit interpolations that are *close* to linear?

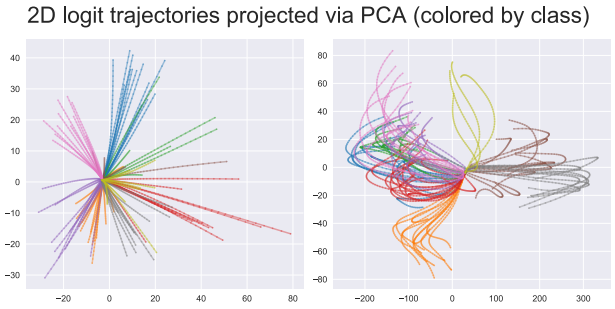

Figure 3. 2D projections (computed with PCA) of logit interpolations for fully-connected networks trained on Fashion-MNIST. Both networks achieve near-perfect final training accuracy. However, the first one (left) interpolates monotonically while the second one (right) does not. The only difference between these two networks is that the second was trained using batch normalization while the first was not.

Measuring logit linearity. There is no standard method to measure the linearity of a curve, but there are several tools from differential geometry that are applicable. In this work, we focus on the length under the Gauss map, which we refer to as the *Gauss length*, a unit-free measure that is related to the curvature. In the case of curves, the Gauss length is computed by mapping the normalized tangent vectors of the curve onto the corresponding projective space (through the so-called Gauss map), and then measuring the length

of the curve in this space. This is described formally in the following definition.

Definition 2 (Gauss length)**.** *Given a curve* $\mathbf{z} : (0, 1) \to \mathbb{R}^d$*. Let* $\hat{\mathbf{v}}(\alpha) = \frac{\partial \mathbf{z}}{\partial \alpha} / \| \frac{\partial \mathbf{z}}{\partial \alpha} \|_2$ *denote the normalized tangent vectors. The length under the Gauss map (Gauss length) is given by:*

$$\int_0^1 \sqrt{\langle \partial_\alpha \hat{\mathbf{v}}(\alpha), \partial_\alpha \hat{\mathbf{v}}(\alpha) \rangle} d\alpha,$$

where $\partial_\alpha \hat{\mathbf{v}}(\alpha)$ *denotes the pushforward of the Gauss map acting on the acceleration vector.*

We refer readers to Lee (2006) or Poole et al. (2016) for a more thorough introduction to these concepts. Intuitively, the Gauss length measures how much the curve bends along its path, with a Gauss length of zero indicating a linear path. In Theorem 3, we prove that a sufficiently small Gauss length guarantees the MLI property for MSE loss.

Theorem 3 (Small Gauss length gives monotonicity)**.** *Let* $\mathcal{L}(\mathbf{z}) = \|\mathbf{z} - \mathbf{z}^*\|_2^2$ *for* $\mathbf{z}^* \in \mathbb{R}^d$*, and let* $\mathbf{z} : (0, 1) \to \mathbb{R}^d$ *be a smooth curve in* $\mathbb{R}^d$ *with* $\mathbf{z}(1) = \mathbf{z}^*$ *and* $\mathcal{L}(\mathbf{z}(0)) > 0$*. If the Gauss length of* $\mathbf{z}$ *is less than* $\pi/2$*, then* $\mathcal{L} \circ \mathbf{z}(\alpha)$ *is monotonically decreasing in* α*.*

See Appendix A for the proof. Informally, this theorem can be understood through a simple physical analogy. Imagine that you are standing on the inside surface of a uniform bowl and wish to increase your height before reaching the bottom. To do so, you must walk at an angle that is at least $\pi/2$ relative to the line connecting you to the bottom. Now, the smallest total rotation that guarantees your return to the bottom is at least $\pi/2$ radians.

Importantly, Theorem 3 applies to arbitrary smooth curves including those produced in the function space of neural networks when we interpolate in the weight space ($\mathbf{z}(\alpha; \mathbf{x})$ above). As an application of Theorem 3, in Appendix A.1, we give sufficient conditions for the MLI property to hold for two-layer linear models (whose loss landscape is non-convex with disconnected globally optimal manifolds (Kunin et al., 2019)). Furthermore, we prove that these sufficient conditions hold almost surely for models satisfying the *tabula rasa* assumptions of Saxe et al. (2019).

One notable departure from the theory in our experiments is that we consider the average loss over the dataset. In this case, individual logit trajectories may be non-monotonic while the network satisfies the MLI property. Nonetheless, we find the average Gauss length to be a good indicator for the monotonicity of the network as a whole.

4. Exploring & Explaining the MLI Property

In this section, we present our empirical investigation of the following questions: 1) How persistent is the MLI property? 2) Why does the MLI property hold? 3) What does the MLI property tell us about the loss landscape of neural networks?

For all experiments, unless specified otherwise, we discretize α in the interval $[0, 1]$ using 50 uniform steps. Here we report statistics from the training set throughout but note that the same observations hold for held-out datasets. Many additional results can be found in Appendix C.

A note on batch normalization. We experiment with networks that use batch normalization during training. These networks require additional care when interpolating network parameters as the running statistics will not align with the activation statistics during interpolation. Therefore, we opt to reset and *warm up* the running statistics for each interpolated set of parameters. This warm-up consists of computing the activation statistics over an epoch of the training data, meaning that each interpolation curve requires an additional 50 epochs (the number of discretizations of α) of data consumption to get accurate loss/accuracy estimates. Note that the learned affine transformation is interpolated as usual.

Experiment settings. We summarize the main settings here with full details of our experimental procedure given in Appendix B. We trained neural networks for reconstruction, classification, and language modeling. For the reconstruction tasks, we trained fully-connected deep autoencoders on MNIST (LeCun et al., 2010). For the classification tasks, we trained networks on MNIST, Fashion-MNIST (Xiao et al., 2017), CIFAR-10, and CIFAR-100 (Krizhevsky et al., 2009). On these datasets, we explored fully-connected networks, convolutional networks, and residual architectures (He et al., 2016). In the above cases, we provide substantial exploration over varying architectures and optimization. We provide a short study on the language modeling setting as well by training LSTM (Hochreiter & Schmidhuber, 1997b) and Transformer (Vaswani et al., 2017) architectures on WikiText-2 (Stephen et al., 2016) dataset. We also experimented with RoBERTa (Liu et al., 2019) on the Esperanto (Conneau et al., 2019) dataset. There, we verify the MLI property and visualize the loss landscape.

4.1. How persistent is the MLI property?

We first investigate the persistence of the MLI property. Goodfellow et al. (2014) showed that the MLI property persists in classification and language modeling tasks (with LSTMs (Hochreiter & Schmidhuber, 1997b)) when trained with SGD. However, several modern advances in neural network training remain unaddressed and the limits of the MLI property have not been characterized. We provide a secondary investigation of the MLI property on reconstruction, classification, and language modelling tasks using modern architectures and methods.

In summary, we found that the MLI property is persistent over most standard neural network training knobs, including (but not limited to): learning task, layer width, depth, activation function, initialization method and regularization. However, there were three mechanisms through which we regularly observed the failure of the MLI property: the use of large learning rates, the use of adaptive optimizers such as Adam (Kingma & Ba, 2014), and the use of batch normalization (Ioffe & Szegedy, 2015). For the remainder of this section, we focus on the effect of these mechanisms but refer readers to Appendix C for a wider view of our study. We defer further analysis of explanations for the MLI property to Section 4.2.

4.1.1. The effect of large learning rates

We found throughout that large learning rates were necessary to train networks that violated the MLI property. However, large learning rates alone were not always sufficient. In Table 1, we show the proportion of networks with non-monotonic interpolations over varying learning rate (including only those models that achieved better than 0.1 training loss). Models trained with SGD using smaller learning rates always exhibited the MLI property. On the other hand, models trained with SGD with larger learning rates often violated the MLI property. For example, 71% of the configurations with a learning rate of 1.0 were found to be non-monotonic. One hypothesis for this behaviour is due to the so-called catapult phase (Lewkowycz et al., 2020; Jastrzebski et al., 2020), where large learning rates encourage the parameters to overcome a barrier in the loss landscape. Additional results on the effect of using larger learning rates can be found in Appendix C.2.

4.1.2. The effect of adaptive optimizers

Prior work has only investigated the MLI property when training with SGD. To address this gap, we trained a wide variety of networks with adaptive optimizers (RMSProp (Hinton et al., 2012), Adam (Kingma & Ba, 2014), and K-FAC (Martens & Grosse, 2015)). Across all settings, we found that adaptive optimizers with large learning rates frequently led to models violating the MLI property.

MNIST autoencoders. For image reconstruction, we evaluated the MLI property for deep fully-connected autoencoders trained on MNIST. We trained autoencoders with SGD and Adam, with varying learning rates and with a varying number of hidden layer size.

In Figure 4, we show the training loss over the interpolated path for autoencoders with final loss (MSE) lower than 30. The majority of the networks trained with SGD retained the MLI property (with few failures at large learning rates). However, when trained with the Adam optimizer, a larger proportion of converged networks exhibited non-monotonic interpolations.

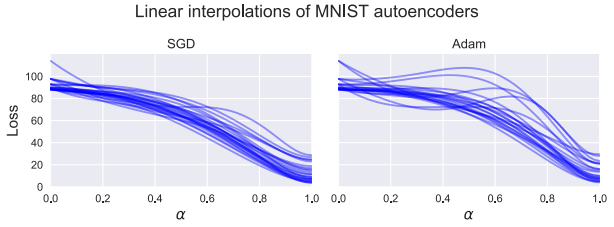

Figure 4. Training loss over linear interpolation of deep autoencoders trained on MNIST using SGD and Adam. Each interpolation line is for a training configuration with different hyperparameters (achieving better than 30 training loss).

MNIST & Fashion-MNIST classifiers. On the MNIST and Fashion-MNIST datasets, we explored varying dataset size, network size (depth/width of hidden layers), activation function, choice of optimizer, optimization hyperparameters, initialization methods, and the use of batch normalization. In Table 1, we compare two-layer networks trained with SGD and Adam. Models trained with SGD typically retained the MLI property but those trained with Adam frequently did not. In Appendix C.3, we show additional results for models trained with RMSProp and K-FAC (whose behaviour is qualitatively close to Adam) along with the interpolated loss curves.

CIFAR-10 & CIFAR-100 classifiers. On CIFAR-10 and CIFAR-100 datasets, we trained two-layer convolutional neural networks (SimpleCNN), LeNet (LeCun et al., 1989), AlexNet (Krizhevsky et al., 2012), VGG16, VGG19 (Simonyan & Zisserman, 2014), and ResNets (He et al., 2016) with different choices of optimizer and learning rate. In Figure 2, we show a broad overview of the interpolation paths for different architectures and optimizers. Overall, Adam-trained models violated the MLI property 3.2 times more often than SGD-trained models.

4.1.3. The effect of batch normalization

Batch normalization's invention and subsequent ubiquity postdate the initial investigation of the MLI property. Even now, the relationship between the MLI property and the use of batch normalization has not been investigated. We provide the first such study in this section. We found that the use of batch normalization greatly increased the rate at which trained networks failed to satisfy the MLI property.

MNIST & Fashion-MNIST classifiers. Table 1 shows the effect of batch normalization on the MLI property for fully connected classifiers trained on MNIST & Fashion-MNIST. The networks trained with batch normalization failed to satisfy the MLI property more frequently than

	LR:	0.001	0.003	0.01	0.03	0.1	0.3	1.0	3.0
SGD	BN	0.00 (20)	0.00 (24)	0.00 (24)	0.00 (24)	0.00 (24)	0.17 (24)	0.83 (24)	1.00 (16)
	No BN	0.00 (4)	0.00 (8)	0.00 (12)	0.00 (20)	0.20 (20)	0.00 (12)	0.00 (4)	0.00 (4)
Adam	BN	0.17 (24)	0.68 (22)	0.83 (24)	1.00 (24)	1.00 (16)	1.00 (16)	1.00 (4)	-
	No BN	0.00 (24)	0.20 (20)	0.00 (12)	0.00 (4)	-	-	-	-

Table 1. Proportion of trained MNIST & Fashion-MNIST classifiers (achieving better than 0.1 training loss) that had non-monotonic interpolations from initialization to final solution. The total number of runs with less than 0.1 training loss is displayed in parentheses next to the proportion. A dashed line indicates that no networks achieved 0.1 loss.

		BN	BN-I	NBN-I	NBN-F
SGD	% (total)	0.54 (26)	0.00 (26)	0.00 (23)	0.11 (27)
	min Δ	0.794	0.000	0.000	0.076
Adam	% (total)	0.77 (22)	0.27 (30)	0.20 (20)	0.04 (23)
	min Δ	0.351	0.054	0.033	0.332

Table 2. Evaluation of effect of batch normalization, initialization, and choice of optimizer for residual networks trained on CIFAR-10 (achieving better than 1.0 training loss). We display the proportion of networks with non-monotonic interpolation and average min Δ such that the network is Δ-monotonic over varying training settings. Full explanation of table is given in main text.

those without. This is more pronounced with large learning rates and with Adam.

CIFAR-10 & CIFAR-100 classifiers. Next, we trained ResNet models on CIFAR-10 & CIFAR-100 classification tasks. We evaluated ResNet-{20,32,44,56} trained with Adam and SGD and with varying learning rates. We also varied the distribution over initial parameters and whether or not batch normalization was applied. The results for CIFAR-10 are displayed in Table 2 (CIFAR-100 results are similar, and are presented in Appendix C). The column headers, "BN" and "NBN" indicate batch normalization and no batch normalization respectively. The suffices "I" and "F" indicate two alternative initialization schemes, block-identity initialization (Goyal et al., 2017) and Fixup initialization (Zhang et al., 2019b). For each configuration, we report the percentage of models violating the MLI property and the average minimum Δ such that the model is Δ-monotonic (conditioning on $\Delta > 0$). Batch normalization led to significantly more networks with non-monotonic interpolations. We also observed that the initialization of the residual blocks plays an important role in shaping the loss landscape.

4.2. Why does MLI hold?

In Section 3, we discussed the parameter- and function-space perspectives of the MLI property. In our experiments, we explore these two perspectives on reconstruction and classification tasks. We also provide a similar analysis on the language modelling task in Appendix C. We computed the average Gauss length of the logit interpolations and the weight distance travelled. In both cases, these measures are predictive of MLI in practice, even for values exceeding the limits of our theory.

In Appendix C.1, we provide the full set of results for all settings we explored. Additionally, we provide an investigation of the relationship between the MLI property and generalization. In summary, we did not find a clear relationship between the success of the MLI property and the generalization ability of the neural network.

4.2.1. Weight distance vs. monotonicity

Throughout our experiments, we found that weight distance was negatively correlated with the monotonicity of the interpolated network. In Figure 5 (left), we show the relationship between the (normalized) distance travelled in weight space and the minimum Δ such that fully-connected classifiers are $\Delta-$monotonic.

First, we note that larger learning rates encourage greater movement in weight space — a finding that also extends to batch normalization and the use of adaptive optimizers. Second, we observed that the networks that travelled short distances during optimization consistently satisfied the MLI property. Conversely, networks with larger distances travelled in weight space were more likely to exhibit non-monotonic loss interpolations. In Appendix C.5, we show similar results for the autoencoders, CIFAR-10 & CIFAR-100 classifiers, LSTM & Transformers, and comparisons over batch normalization and adaptive optimizers.

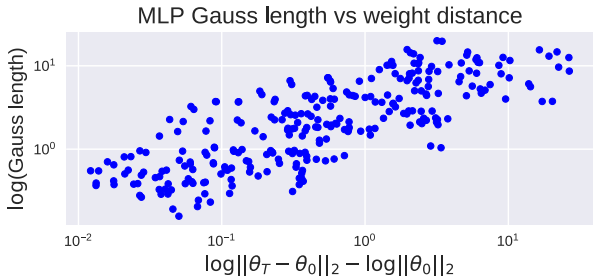

Figure 7. Power law relationship between Gauss length and weight distance travelled for MLP & Fashion-MNIST experiments. ($R^2 = 0.616$)

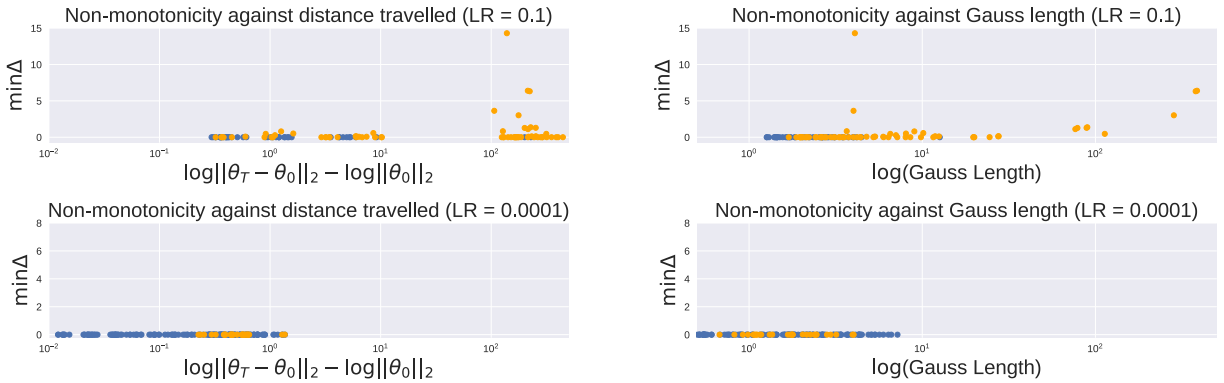

Figure 5. For each MNIST & Fashion-MNIST classifier, we compute the minimum Δ such that the interpolated loss is Δ-monotonic. We plot models trained with a learning rate of 0.1 and 0.0001 in the top and bottom rows respectively. On the left, we compare the distance moved in the weight space. On the right, we compare the Gauss length of the interpolated network outputs. **Blue** points represent networks where the MLI property holds and **orange** points are networks where the MLI property fails.

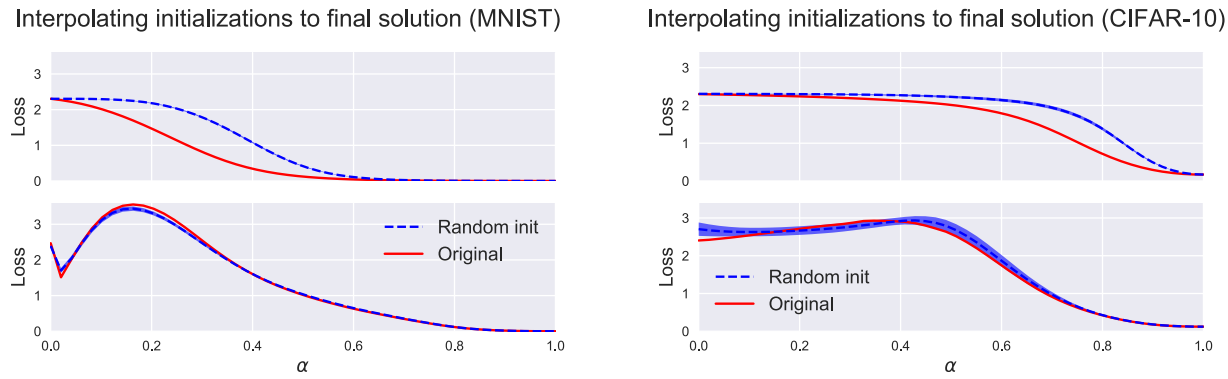

Figure 6. Classifier interpolation loss on training set between 15 different random initializations and an optimum. The top row shows interpolation towards a final solution that is monotonic with its original initialization. The bottom row shows this interpolation for a non-monotonic original pair. For the random initializations, mean loss is shown with standard deviation (± 1) as filled region.

4.2.2. Gauss Length vs. Monotonicity

We also observed a negative correlation between the Gauss length of the logit interpolations and the minimum Δ such that the loss interpolation is Δ-monotonic. In Figure 5 (right), we make this comparison for classifiers trained on MNIST & Fashion-MNIST. As our analysis predicts, small Gauss lengths lead to monotonic interpolations. And beyond the strict limits of our theoretical analysis, we find that as the Gauss length increases, the non-monotonicity also increases.

We also observed that larger learning rates lead to much larger Gauss lengths. As with the weight distance, this finding extends to batch normalization and the use of adaptive optimizers too (see Appendix C.6). In Appendix C.4, we conduct an ablation study to investigate the relationship between Gauss length and the choice of optimizer by changing the optimizer in the middle of training (SGD $\rightarrow$ Adam and Adam $\rightarrow$ SGD). Switching to Adam at any point during training leads to large Gauss length and weight distance without a significant spike in the training loss — with little variation due to the time of the optimizer switch.

4.2.3. Gauss Length vs Weight Distance

When the distance moved in weight space is small, we would expect a small Gauss length as a linearization of the network provides a good approximation. However, it is not obvious what relationship (if any) should be expected more generally. Surprisingly, we consistently observed a power-law relationship between the average Gauss length and the distance moved in weight space (Figure 7). We observed this relationship across all of the experimental settings that we explored. Full results are presented in Appendix C.7.

Thus far, we focused on the monotonicity of paths connecting the initialization and final network solution. In this section, we ask: are (non-)monotonic interpolations unique to the initialization and final solution pair?

To this end, we evaluated linear interpolations between learned network parameters and unrelated random initializations (Figure 6). For a fully-connected MNIST classifier and a ResNet-20 trained on CIFAR-10, we found that random initializations display the same interpolation behaviour as

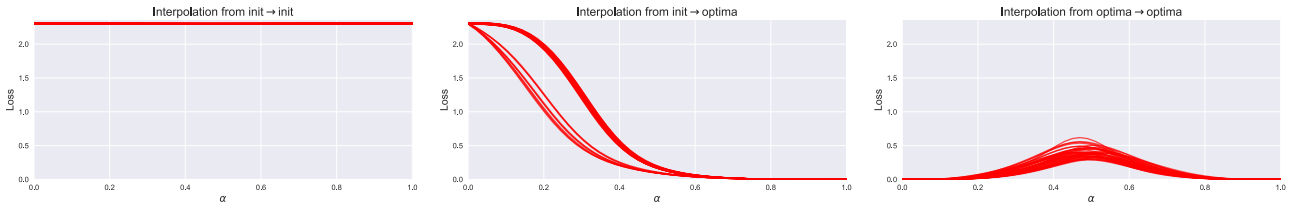

Figure 8. Linear interpolation for 10 FashionMNIST classifiers with less than 0.01 final loss. Left: Interpolating between all pairs of initializations. Middle: Interpolating from all initializations to all optima. Right: Interpolating between all pairs of optima.

the original initialization-solution pair. This suggests that the MLI property is not tied to a particular pair of parameters but rather is a global property of the loss landscape. We also explored linear interpolations between pairs of initializations, initialization to optima pairs, and pairs of optima in Figure 8. No barriers were observed between the pairs of initializations or the initialization → optimum pairs, but barriers are present between the optima. This highlights the rich structure present in the loss landscape of these models and aligns well with the qualitative predictions of Fort & Jastrzebski (2019).

Finally, we provide visualizations of the loss landscape via 2D projections of the parameter space. While low-dimensional projections of high-dimensional spaces are often misleading, in the case of linear interpolations, the entire path lies in the projected plane. Therefore, these visualizations give us valuable insight into connectivity in the loss landscape for multiple initialization → final solution paths.

In Figure 9, we show 2D projections of the loss landscape for RoBERTa (Liu et al., 2019) trained as a language model on Esperanto (Conneau et al., 2019) using the HuggingFace library (Wolf et al., 2020). We trained two models and plotted the initial points and optima for both. Both initial points are monotonically connected to both minima.

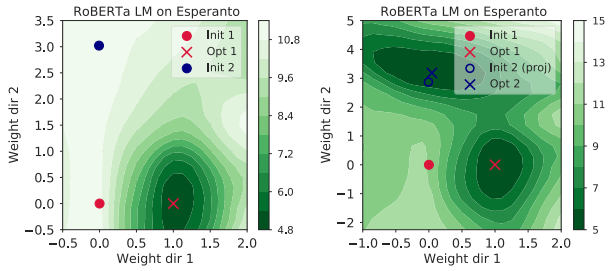

Figure 9. Two-dimensional sections of the weight space for RoBERTa trained as a language model on Esperanto. Left: plane defined by two initializations and the optima reached from one of them is shown. Right: plane defined by "Init 1" and two optima are shown (with "Init 2" projected onto the plane).

5. Conclusion

Goodfellow et al. (2014) first showed that linear interpolation between initial and final network parameters monotonically decreases the training loss. In this work, we provided the first evidence that this so-called, Monotonic Linear Interpolation (MLI), is not a stable property of neural network training. In doing so, we provided a deeper theoretical understanding of the MLI property and properties of the loss landscape in general. Our empirical investigation of the MLI property explored variations in datasets, architecture, optimization, and other training mechanisms. We identified several mechanisms that systematically produce trained networks that violate the MLI property, and connected these mechanisms to our theoretical explanations of the MLI property. Additional results indicate that the MLI property is not unique to the initialization→solution pair produced by training, but rather is a global property of the loss landscape connecting arbitrary initialization→solution pairs. The empirical and theoretical analysis we presented highlights the intriguing properties of neural network loss landscapes.

6. Acknowledgements

We thank Yani Ioannou, Will Grathwohl, Jake Snell, Eleni Triantafillou, and David Madras for valuable feedback on this paper. We would also like to thank Jimmy Ba, Guodong Zhang, and our many other colleagues for their helpful discussions throughout this research. Specifically, we appreciate Bryn Elesedy asking about connectivity from unrelated initializations. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute (`www.vectorinstitute.ai/partners`).

References

Agarwal, N., Anil, R., Hazan, E., Koren, T., and Zhang, C. Disentangling adaptive gradient methods from learning rates. *arXiv preprint arXiv:2002.11803*, 2020.

Amari, S.-I. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Amari, S.-i., Ba, J., Grosse, R., Li, X., Nitanda, A., Suzuki, T., Wu, D., and Xu, J. When does preconditioning help

or hurt generalization? *arXiv preprint arXiv:2006.10732*, 2020.

Boyd, S., Boyd, S. P., and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets. *arXiv preprint arXiv:1703.04933*, 2017.

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

Fort, S. and Ganguli, S. Emergent properties of the local geometry of neural loss landscapes. *arXiv preprint arXiv:1910.05929*, 2019.

Fort, S. and Jastrzebski, S. Large scale structure of neural network loss landscapes. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Fort, S. and Scherlis, A. The Goldilocks zone: Towards better understanding of neural network loss landscapes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3574–3581, 2019.

Fort, S., Dziugaite, G. K., Paul, M., Kharaghani, S., Roy, D. M., and Ganguli, S. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. *arXiv preprint arXiv:2010.15110*, 2020.

Frankle, J. Revisiting" qualitatively characterizing neural network optimization problems". *arXiv preprint arXiv:2012.06898*, 2020.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Frankle, J., Dziugaite, G. K., Roy, D. M., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. *arXiv preprint arXiv:1912.05671*, 2019.

Freeman, C. D. and Bruna, J. Topology and geometry of half-rectified network optimization. 2016.

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

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Gur-Ari, G., Roberts, D. A., and Dyer, E. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hinton, G., Srivastava, N., and Swersky, K. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8), 2012.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural Computation*, 9(1):1–42, 1997a.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997b.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Im, D. J., Tao, M., and Branson, K. An empirical analysis of the optimization of deep network loss surfaces. *arXiv preprint arXiv:1612.04010*, 2016.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8571–8580, 2018.

Jastrzebski, S., Szymczak, M., Fort, S., Arpit, D., Tabor, J., Cho*, K., and Geras*, K. The break-even point on optimization trajectories of deep neural networks. In *International Conference on Learning Representations*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Krizhevsky, A. et al. Learning multiple layers of features from tiny images. 2009.

Kuditipudi, R., Wang, X., Lee, H., Zhang, Y., Li, Z., Hu, W., Ge, R., and Arora, S. Explaining landscape connectivity of low-cost solutions for multilayer nets. In *Advances in Neural Information Processing Systems*, pp. 14574–14583, 2019.

Kunin, D., Bloom, J., Goeva, A., and Seed, C. Loss landscapes of regularized linear autoencoders. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3560–3569. PMLR, 09–15 Jun 2019.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

LeCun, Y., Cortes, C., and Burges, C. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

Lee, J. M. *Riemannian manifolds: an introduction to curvature*, volume 176. Springer Science & Business Media, 2006.

Lewkowycz, A., Bahri, Y., Dyer, E., Sohl-Dickstein, J., and Gur-Ari, G. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018.

Liu, C., Zhu, L., and Belkin, M. On the linearity of large non-linear models: when and why the tangent kernel is constant. *Advances in Neural Information Processing Systems*, 33, 2020.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Martens, J. and Grosse, R. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015.

Matthews, A. G. d. G., Rowland, M., Hron, J., Turner, R. E., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

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

Papyan, V. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning Research*, 21(252):1–64, 2020.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. Exponential expressivity in deep neural networks through transient chaos. *arXiv preprint arXiv:1606.05340*, 2016.

Saxe, A. M., McClelland, J. L., and Ganguli, S. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.

Schaul, T., Zhang, S., and LeCun, Y. No more pesky learning rates. In *International Conference on Machine Learning*, pp. 343–351, 2013.

Shevchenko, A. and Mondelli, M. Landscape connectivity and dropout stability of sgd solutions for over-parameterized neural networks. *arXiv preprint arXiv:1912.10095*, 2019.

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

Stephen, M., Caiming, X., James, B., and Socher, R. 2016.

Sun, R. Optimization for deep learning: theory and algorithms. *arXiv preprint arXiv:1912.08957*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

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

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M.,

Lhoest, Q., and Rush, A. M. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics.

Wu, Y., Ren, M., Liao, R., and Grosse, R. Understanding short-horizon bias in stochastic meta-optimization. *arXiv preprint arXiv:1803.02021*, 2018.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Zhang, G., Li, L., Nado, Z., Martens, J., Sachdeva, S., Dahl, G., Shallue, C., and Grosse, R. B. Which algorithmic choices matter at which batch sizes? insights from a noisy quadratic model. In *Advances in Neural Information Processing Systems*, pp. 8196–8207, 2019a.

Zhang, H., Dauphin, Y. N., and Ma, T. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019b.