The Impact of Neural Network Overparameterization on Gradient Confusion and Stochastic Gradient Descent

Karthik A. Sankararaman*¹, Soham De*², Zheng Xu¹, W. Ronny Huang¹, Tom Goldstein¹

¹Department of Computer Science, University of Maryland
²DeepMind, London
{kabinav, xuzh, rhuang, tomg}@cs.umd.edu, sohamde@google.com

*Both authors contributed equally to this work.

Abstract

The goal of this paper is to study why stochastic gradient descent (SGD) is efficient for neural networks, and how neural net design affects SGD. In particular, we investigate how overparameterization – an increase in the number of parameters beyond the number of training data – affects the dynamics of SGD. We introduce a simple concept called gradient confusion. When confusion is high, stochastic gradients produced by different data samples may be negatively correlated, slowing down convergence. But when gradient confusion is low, we show that SGD has better convergence properties than predicted by classical theory. Using theoretical and experimental results, we study how overparameterization affects gradient confusion, and thus the convergence of SGD, on linear models and neural networks. We show that increasing the number of parameters of linear models or increasing the width of neural networks leads to lower gradient confusion, and thus faster and easier model training. We also show how overparameterization by increasing the depth of neural networks results in higher gradient confusion, making deeper models harder to train. Finally, we observe empirically that techniques like batch normalization and skip connections reduce gradient confusion, which helps reduce the training burden of deep networks.
1 Introduction

Stochastic gradient descent (SGD) [RM51] and its variants with momentum [SMDH13, Nes83, Pol64] have become the standard optimization routine for neural networks due to their fast convergence and good generalization properties [WRS +17, KS17, SMDH13]. Yet the behavior of SGD on high-dimensional neural network models still eludes full theoretical understanding, both in terms of its convergence and generalization properties. In this paper, we study why SGD is so efficient at converging to low loss values on most standard neural networks, and how neural net architecture design affects training performance.

Classical stochastic optimization theory predicts that the learning rate of SGD needs to decrease over time for convergence to be guaranteed to the minimizer of a convex function [SZ13, Ber11]. For strongly convex functions for example, such results show that a decreasing learning rate schedule of $O(1/k)$ is required to guarantee convergence to within $\epsilon$-accuracy of the minimizer in $O(1/\epsilon)$ iterations, where $k$ denotes the iteration number. Typical stochastic optimization procedures experience a transient phase, where the optimizer makes progress towards a neighborhood of a minimizer, followed by a stationary phase, where the gradient noise starts to dominate the signal and the optimizer typically oscillates around the minimizer [DM92, Mur98, TA +17, CT18]. With decaying learning rates of the form $O(1/k)$ or $O(1/\sqrt{k})$, the convergence of SGD in the transient phase can be very slow, typically leading to poor performance on standard neural network problems.

Neural networks operate in a regime where the number of parameters is much larger than the number of training data. In this regime, SGD seems to converge quickly with constant learning rates. So quickly, in fact, that neural net practitioners often use a constant learning rate for the majority of training, with exponentially decaying learning rate schedules towards the end, without seeing the method stall [KSH12, SZ14, HZRS16, ZK16]. With constant learning rates, theoretical guarantees show that SGD converges quickly to a neighborhood of the minimizer (i.e., fast convergence in the transient phase), but then reaches a noise floor beyond which it stops converging; this noise floor depends on the learning rate and the variance of the gradients at the minimizer [MB11, NWS14]. Some more recent results have shown that when models can over-fit the data completely while being strongly convex, convergence without a noise floor is possible without decaying the learning rate [SR13, MBB17, BBM18, VBS18]. While these results do give insights into why constant learning rates followed by an exponential decay might work well in practice [CT18], they fail to fully explain the efficiency of SGD on neural nets, and how they relate to overparameterization.

Training performance is also highly affected by the neural network architecture. It is common knowledge among neural network practitioners that deeper networks train slower [BSF94, GB10]. This has led to several innovations over the years to get deeper networks to train more easily, such as careful initialization strategies [GB10, HZRS15, ZDM19], residual connections [HZRS16], and various normalization schemes like batch normalization [IS15] and weight normalization [SK16]. Furthermore, there is ample evidence to indicate that wider networks are easier to train [ZK16, NH17, LXS +19], and recent theoretical results suggest that the dynamics of SGD simplify considerably for very wide networks [JGH18, LXS +19]. Several prior works have investigated the difficulties of training deep networks [GB10, BFL +17], and the benefits of width [NH17,
LXS+19, DZPS18, AZLS18]. This work adds to the existing literature by identifying and analyzing a condition that enables us to establish a more direct relationship between layer width, network depth, problem dimensionality, and SGD dynamics on overparameterized networks.

**Our contributions.** The goal of this paper is to study why SGD is efficient for neural nets, and how neural net design affects SGD. Typical neural nets are overparameterized (i.e., the number of parameters exceed the number of training points). We ask how this over-parameterization, as well as the architecture of a neural net, affect the dynamics of SGD. We list the main contributions of this work below.

- We identify a condition, termed **gradient confusion**, that impacts the convergence properties of SGD on overparameterized models. When confusion is high, stochastic gradients produced by different data samples may be negatively correlated, causing slow convergence. On the other hand, when confusion is low, we show that convergence is accelerated, indicating a regime where constant learning rates work well in practice (Sections 2 and 3).

- We study the effect of overparameterization on gradient confusion (Section 4), and show that gradient confusion increases as the network depth increases, indicating the difficulty in training very deep networks. Our results require minimal assumptions, and hold for a large family of neural networks with non-linear activations and a large class of loss-functions.

- We study the effect of layer width and show that wider networks have lower gradient confusion for standard initialization procedures making them easier to train (Section 5). This result further indicates that for wider networks, the dynamics of SGD becomes closer to deterministic gradient descent on each training example separately.

- We perform extensive experiments on wide residual networks (WRNs) [ZK16], convolutional networks (CNNs) and multi-layer perceptrons (MLPs) for image classification tasks on CIFAR-10, CIFAR-100 and MNIST (Section 6 and Appendix A). We show that our theoretical results consistently hold across almost all our experiments. We further show that innovations like batch normalization and skip connections in residual networks help lower gradient confusion, thus indicating why standard neural networks that employ such techniques are so efficiently trained using SGD.

## 2 Gradient confusion

**Notations.** Throughout this paper, vectors are represented in bold lower-case while matrices in bold upper-case. We use $W_{i,j}$ to indicate the $(i, j)$ cell in matrix $W$ and $(W)_i$ for the $i^{th}$ row of matrix $W$. $\|W\|$ denotes the operator norm of $W$. $[N]$ denotes the set $\{1, 2, \ldots, N\}$ and $[N]_0$ denotes the set $\{0, 1, \ldots, N\}$.

**SGD basics.** Given $N$ training points (specified by the corresponding loss functions $\{f_i\}_{i \in [N]}$), we
use SGD to solve empirical risk minimization problems of the form,

\[
\min_{w \in \mathbb{R}^d} F(w) := \min_{w \in \mathbb{R}^d} \frac{1}{N} \sum_{i=1}^{N} f_i(w),
\]

using the following iterative update rule for \( T \) rounds:

\[
w_{k+1} = w_k - \alpha_k \nabla \tilde{f}_k(w_k).
\]

Here \( \alpha_k \) is the learning rate and \( \tilde{f}_k \) is a function chosen uniformly at random from \( \{f_i\}_{i \in [N]} \) at iteration \( k \in [T] \). In this paper, we consider constant learning rates \( \alpha_k = \alpha, \forall k \). We use \( w^* \) to denote the optimal solution, i.e., \( w^* = \arg \min_w F(w) \).

**Gradient confusion.** SGD works by iteratively selecting a random function \( \tilde{f}_k \), and modifying the parameters to move in the direction of the negative gradient of the objective term \( \tilde{f}_k \) without considering the effect on other terms. It may happen that the selected gradient \( \nabla \tilde{f}_k \) is negatively correlated with the gradient of another term \( \nabla f_j \). When the gradients of different mini-batches are negatively correlated, the objective terms disagree on which direction the parameters should move, and we say that there is **gradient confusion**\(^1\).

**Definition 2.1.** A set of objective functions \( \{f_i\}_{i \in [N]} \) has gradient confusion bound \( \eta \geq 0 \) if the pair-wise inner products between gradients satisfy, for a fixed \( w \in \mathbb{R}^d \),

\[
\langle \nabla f_i(w), \nabla f_j(w) \rangle \geq -\eta, \forall i \neq j \in [N].
\]

SGD converges fast when the gradient confusion is low. To see why, consider the case of training a logistic regression model on a dataset with orthogonal vectors. We have \( f_i(w) = \ell(y_i \cdot x_i^\top w) \), where \( \ell : \mathbb{R} \to \mathbb{R} \) is the logistic loss, \( \{x_i\}_{i \in [N]} \) is a set of orthogonal training vectors, and \( y_i \in \{-1, 1\} \) is the label for the \( i^{th} \) training example. We then have \( \nabla f_i(w) = \zeta_i x_i \), where \( \zeta_i = y_i \ell'(y_i \cdot x_i^\top w) \). Because of our orthogonal data assumption, the gradient confusion is 0 since \( \langle \nabla f_i(w), \nabla f_j(w) \rangle = \zeta_i \zeta_j \langle x_i, x_j \rangle = 0, \forall i, j \in [N] \) and \( i \neq j \). Because of gradient orthogonality, an update in the gradient direction \( f_i \) has no effect on the loss value of \( f_j \) for \( i \neq j \). In this case, SGD decouples into a (deterministic) gradient decent on each objective term separately, and we can expect to see the fast rates of convergence attained by deterministic gradient descent, rather than the slow rates of SGD.

Can we expect a problem to have low gradient confusion in practice? From the logistic regression problem, we have: \( |\langle \nabla f_i(w), \nabla f_j(w) \rangle| = |\langle x_i, x_j \rangle| \cdot |\zeta_i \zeta_j| \). This inner product is expected to be small for all \( w \); the logistic loss satisfies \( |\zeta_i \zeta_j| < 1 \), and for fixed \( N \) the quantity \( \max_{i,j} |\langle x_j, x_i \rangle| \) is \( O(1/\sqrt{d}) \) whenever \( \{x_i\} \) are randomly sampled from a sphere.\(^2\) Specifically, we have the following

\(^1\)This is related to gradient diversity [YPL+17] but with important differences, which we describe in Section 7.

\(^2\)More generally, this is true whenever \( x_i = \frac{1}{\sqrt{d}} y_i \), where \( y_i \) is an isotropic random vector [Ver18].
Lemma, which is often attributed to [MS86] (see Appendix B for a short proof).

Lemma 2.1 (Near orthogonality of random vectors). For vectors \{x_i\}_{i \in [N]} drawn uniformly from a unit sphere in d dimensions, and \(\nu > 0\),

\[
\Pr \left[ \max_{i,j} |x_i^\top x_j| > \nu \right] \leq N^2 \sqrt{\frac{\pi}{8}} \exp \left( - \frac{d-1}{2}\nu^2 \right).
\]

Thus, we would expect an average-case (i.e., random) linear model to have nearly orthogonal gradients, provided that we don’t train on too many training vectors (as the number of training vectors grow relative to the dimension, it becomes likely that we will see two training vectors with large negative correlation). In other words, we should expect a random linear model to have low gradient confusion when the number of parameters is "large" and the number of training data is "small" – i.e., when the model is over-parameterized. This is further evidenced by a toy example in Figure 1, where we show a slightly overparameterized linear regression model can have much faster convergence rates, as well as lower gradient confusion, compared to the underparameterized model.

Now consider more general neural net problems. There is evidence that the Hessian at the minimizer is very low rank for many standard overparameterized neural net models [SEG+17, Coo18, CCS+16, WZ+17]. What does this imply for the gradient confusion? For clarity in presentation, suppose each \(f_i\) has a minimizer at the origin (the same argument can be easily extended to the more general case). Suppose also that there is a Lipschitz constant for the Hessian of each function \(f_i\) that satisfies \(\|H_i(w) - H_i(w')\| \leq L_H \|w - w'\|\). Then \(\nabla f_i(w) = H_i w + e\), where \(e\) is an error term bounded as: \(\|e\| \leq \frac{1}{2} L_H \|w\|^2\), and we use the shorthand \(H_i\) to denote \(H_i(0)\). Then we have:

\[
|\langle \nabla f_i(w), \nabla f_j(w) \rangle| = |\langle H_i w, H_j w \rangle + \langle e, H_i w + H_j w \rangle + \|e\|^2 |
\leq \|w\|^2 \|H_i\| \|H_j\| + \|e\|\|w\| (\|H_i\| + \|H_j\|) + \|e\|^2
\leq \|w\|^2 \|H_i\| \|H_j\| + \frac{1}{2} L_H \|w\|^3 (\|H_i\| + \|H_j\|) + \frac{1}{4} L_H^2 \|w\|^4.
\]
If the Hessians are sufficiently random and low-rank (e.g., of the form $H_i = a_i a_i^\top$ where $a_i \in \mathbb{R}^{N \times r}$ are randomly sampled from a unit sphere), then one would expect the terms in this expression to be small for all $w$ within a neighborhood of the minimizer. Given that it has been observed that many standard neural nets have low rank Hessians at the minimizer, this indicates that the gradient confusion might be low for a large class of weights near the minimizer for these models.

The above arguments are rather informal, and ignore issues like the effect of the structure of neural networks. Furthermore, it is unclear whether we can expect low levels of gradient confusion in practice, and what effect non-zero confusion has on convergence rates. Below, we present a rigorous argument that low confusion levels accelerate SGD and help achieve faster convergence for non-convex problems. Then, we turn to the issue of over-parameterization, and study how it affects gradient confusion, and how this depends on neural network architecture. Finally, we use computational experiments to show that gradient confusion is low for standard neural nets used in practice, and that this effect contributes to the superior optimization performance of SGD.

3 SGD is efficient when gradient confusion is low

We now present a rigorous analysis of gradient confusion and its effect on SGD. Several prior papers have analyzed the convergence rates of constant learning rate SGD [NB01, MB11, NWS14, FB15, DFB17]. These results show that for strongly convex and Lipschitz smooth functions, SGD with a constant learning rate $\alpha$ converges linearly to a neighborhood of the minimizer. The noise floor it converges to depends on the learning rate $\alpha$ and the variance of the gradients at the minimizer, i.e., $\mathbb{E}_i \| \nabla f_i(w^*) \|^2$. To guarantee convergence to $\epsilon$-accuracy in such a setting, the learning rate needs to be small, i.e., $\alpha = O(\epsilon)$, and the method requires $T = O(1/\epsilon)$ iterations. Some more recent results show convergence of constant learning rate SGD without a noise floor and without small step sizes using an "overfitting" condition, i.e., where the model can completely overfit the data [SR13, MBB17, BBM18, VBS18]. The condition effectively translates to assuming $\mathbb{E}_i \| \nabla f_i(w^*) \|^2 = 0$, getting rid of the noise floor.

The gradient confusion bound is related to the overfitting condition. Cauchy-Swartz inequality implies that if $\mathbb{E}_i \| \nabla f_i(w^*) \|^2 = O(\epsilon)$, then $\mathbb{E}_{i,j} | \langle \nabla f_i(w^*), \nabla f_j(w^*) \rangle | = O(\epsilon), \forall i, j$. This implies that the gradient confusion at the minimizer is small when the variance of the gradients at the minimizer is small. Further note that when the variance of the gradients at the minimizer is small, i.e., $O(\epsilon)$, a direct application of the results in [MB11, NWS14] shows that constant learning rate SGD has fast convergence to $\epsilon$-accuracy in $T = O(\log(1/\epsilon))$ iterations, without the learning rate needing to be small.

Bounded gradient confusion does not, however, provide a bound on the variance of the gradients. Thus, it is instructive to derive convergence bounds of SGD explicitly in terms of the gradient confusion bound, to properly understand its effect. We begin by looking at the case where the objective satisfies the Polyak-Lojasiewicz (PL) inequality [Loj65], a condition related to, but weaker than, strong convexity, and used in recent work in stochastic optimization [KNS16, DYJG17]. Using the PL inequality, we provide tight bounds on the rate of convergence in terms of the optimality
gap. Then we look at a broader class of smooth non-convex functions, and analyze convergence to a stationary point.

We first make two standard assumptions about the objective function.

(A1) The individual functions $f_i$ are $L$-Lipschitz smooth:

$$f_i(w') \leq f_i(w) + \nabla f_i(w)^{\top}(w' - w) + \frac{L}{2}\|w' - w\|^2, \quad \forall i \in [N].$$

(A2) The individual functions $f_i$ satisfy the PL inequality:

$$\frac{1}{2}\|\nabla f_i(w)\|^2 \geq \mu (f_i(w) - f^*_i), \quad \forall i \in [N],$$

where $f^*_i = \min_w f_i(w)$.

We now state the following convergence result of constant learning rate SGD in terms of the gradient confusion bound.

**Theorem 3.1.** If the objective function satisfies A1 and A2, and has gradient confusion bound $\eta$ (eq. 3), SGD with updates of the form (2) converges linearly to a neighborhood of the minima of the problem (1) as:

$$\mathbb{E}[F(w_k) - F^*] \leq \rho^k (F(w_0) - F^*) + \frac{\alpha \eta}{1 - \rho},$$

where $\alpha < \frac{2}{NL}$, $\rho = 1 - \frac{2\mu}{N} \left(\alpha - \frac{NL\alpha^2}{2}\right)$ and $F^* = \min_w F(w)$.

**Proof.** See Appendix D.1 for the proof.

This result shows that SGD converges linearly to a neighborhood of a minimizer, and the size of this neighborhood depends on the level of gradient confusion. When $\eta = 0$, there is no confusion, and SGD converges directly to a minimizer. Further, when the gradient confusion is small, i.e., $\eta = O(\epsilon)$, then SGD has fast convergence to $O(\epsilon)$-accuracy in $T = O(\log(1/\epsilon))$ iterations, without requiring the learning rate to be vanishingly small.

**Convergence on general smooth non-convex functions.** We now show that low gradient confusion leads to fast convergence on more general smooth non-convex functions.

**Theorem 3.2.** If the objective satisfies A1 and the gradient confusion bound (eq. 3), then SGD converges to a neighborhood of a stationary point as:

$$\min_{k=1,\ldots,T} \mathbb{E}\|\nabla F(w_k)\|^2 \leq \frac{\rho(F(w_1) - F^*)}{T} + \alpha \rho \eta,$$

for learning rate $\alpha < 2/(NL)$, $\rho = \frac{2N}{2\alpha - N\alpha^2},$ and $F^* = \min_w F(w)$. 
Proof. See Appendix D.1 for the proof. \qed

Theorems 3.1 and 3.2, similar to previous constant learning rate results, predict an initial phase of optimization with fast convergence to the neighborhood of a minimizer or a stationary point. This behavior is often observed when optimizing neural nets [DM92, SMDH13], where a constant learning rate reaches a high level of accuracy on the model. Convergence slows down as the iterates approach the noise floor, and at this point drops in the objective function are achieved by exponentially decaying learning rate schedules [KSH12, SZ14, HZRS16, ZK16].

Note that the constants in Theorems 3.1 and 3.2 are slightly worse than those shown in previous work [MB11, NWS14]. This is possibly a product of the analysis and the constants can probably be improved. See Appendix C for further discussion of this, where we explore a strengthened gradient confusion bound that guarantees faster local convergence. That being said, we stress that our goal is not to study convergence rates per se, nor is it to prove state-of-the-art rate bounds for this class of problems. The main intention of these theorems is to show the direct effect that the gradient confusion bound has on the convergence rate and the noise floor that constant learning rate SGD converges to. As we show in the following sections, this new perspective in terms of the gradient confusion helps us more directly understand how overparameterization and neural net architecture design affects SGD dynamics and why.

4 Effect of overparameterization on gradient confusion

To draw a more rigorous connection between overparameterization and neural network structure, we analyze gradient confusion for generic (i.e., random) model problems using methods from high-dimensional probability. We first rigorously analyze the case where training data is randomly sampled from a unit sphere, and identify specific cases where gradient confusion is low with high-probability. We then show that similar results hold when we consider arbitrary fixed data and random weights instead. Our results require minimal additional assumptions, and hold for a large family of neural networks with non-linear activations and a large class of loss-functions. In particular, our results hold for fully connected networks (and convolutional networks in some cases) with the square-loss and logistic-loss functions, and commonly used non-linear activations such as sigmoid, tanh and ReLU.

4.1 Gradient confusion on a random data model

In this section, we consider synthetic training data of the form \( \mathcal{D} = \{ (x_i, C(x_i)) \}_{i \in [N]} \), for some labeling function \( C: \mathbb{R}^d \rightarrow [-1, 1] \), and with data points \( \{x_i\} \) drawn uniformly from the surface of a \( d \)-dimensional unit sphere. The concept being learned satisfies \( |C(x)| \leq 1 \) and \( \|\nabla_x C(x)\|_2 \leq 1 \) for \( \|x\| \leq 1 \). Note that this automatically holds for every model considered in this paper where the concept is realizable (i.e., where the model can express the concept function using its parameters), and more generally, this assumes a Lipschitz condition on the labels (i.e., the concept function...
We now prove concentration bounds for the gradient confusion on neural networks. When \( \|w\| \) is small, which force the weights to remain small during optimization. The operator norm of convolutional weights. The small-weights assumption is not just a theoretical concern, but also usually enforced in practice.

**Assumption 4.1**

Is the small-weights assumption reasonable? Without the small-weights assumption, the signal or the gradient is not possible in general, and thus, we consider this restricted class of networks. The small-weights assumption is not just a theoretical concern, but also usually enforced in practice. Neural networks are often trained with weight decay regularizers of the form \( \sum_i \|W_i\|^2_F \), which force the weights to remain small during optimization. The operator norm of convolutional layers have also recently been used as an effective regularizer as well for image classification tasks [SGL18]. We can also show more formally that Assumption 4.1 holds with high probability for randomly initialized weights.

We now prove concentration bounds for the gradient confusion on neural networks.

**Theorem 4.1** (Concentration bounds for arbitrary depth neural networks). Consider the problem of fitting neural networks (eq. 4) to data using either the square-loss or the logistic-loss function. Let \( \eta > 0 \) be a given constant. Let the weights satisfy Assumption 4.1 and the non-linearities in each layer satisfy properties (P1) and (P2). For some fixed constant \( c > 0 \), with probability at least

\[
1 - N^2 \exp \left( \frac{-c\eta^2}{16\zeta^4(\beta + 2)^4} \right),
\]
the gradient confusion bound in eq. (3) holds. For both the square-loss and the logistic-loss functions, the value of $\zeta_0 \leq 2\sqrt{\beta}$ (from Lemma D.1).

Note that the convergence rate results of SGD in Section 3 assume that the gradient confusion bound holds at every point along the path of SGD. On the other hand, Theorem 4.1 shows concentration bounds for the gradient confusion at a fixed weight $W$. Thus, to ensure that the above result is relevant for the convergence of SGD on overparameterized models, we now make the concentration bound in Theorem 4.1 uniform over all weights inside a ball $B_r$ of radius $r$.

**Corollary 4.1** (Uniform concentration around the minimizer). Select a point $W = (W_0, W_1, \ldots, W_\beta)$, satisfying Assumption 4.1. Consider a ball $B_r$ centered at $W$ of radius $r > 0$. If the data $\{x_i\}_{i \in [N]}$ are sampled uniformly from a unit sphere, then the gradient confusion bound in eq. (3) holds uniformly at all points $W' \in B_r$ with probability at least

$$1 - N^2 \exp\left(-\frac{c d \eta^2}{64 \zeta_0^4 (\beta + 2)^4}\right), \quad \text{if } r \leq \eta/4 \zeta_0^2,$$

$$1 - N^2 \exp\left(-\frac{c d \eta^2}{64 \zeta_0^4 (\beta + 2)^4} + \frac{8d \zeta_0^2 r}{\eta}\right), \quad \text{otherwise}.$$

For both the square-loss and the logistic-loss functions, the value of $\zeta_0 \leq 2\sqrt{\beta}$ (from Lemma D.1).

Thus, for a given dimension $d$ and number of samples $N$, when the depth $\beta$ decreases, the probability that the gradient confusion bound in eq. (3) holds increases, and vice versa. This helps explain why training very deep models is hard and typically slow with SGD, as observed previously [BSF94, GB10]. Note that this is also related to the shattered gradients phenomenon [BFL+17] that arises with depth (see Section 7 for more discussion). This naturally raises the question why modern deep neural networks are so efficiently trained using SGD. While careful initialization strategies prevent vanishing or exploding gradients making deeper networks trainable, as we show in Section 4.2, these strategies still suffer from high gradient confusion for very deep networks (see [ZDM19] for recent progress in this direction). Thus, in Section 6, we empirically explore how popular techniques like residual connections [HZRS16] and batch normalization [IS15] affect gradient confusion. We find that these techniques drastically lower gradient confusion, making very deep networks significantly easier to train using SGD.

Note that the results automatically hold for convolutional neural nets, since a convolutional operation on $x$ can be represented as a matrix multiplication $Ux$ for an appropriate Toeplitz matrix $U$.

### 4.2 Gradient confusion at initialization

We now show that similar results to Section 4.1 hold when assuming arbitrary bounded data and random weights (more specifically, we consider typical weight initializations used when training neural networks). We consider the scenario when the weights of the neural network are chosen at random from the interval, $[-1/\sqrt{\ell}, 1/\sqrt{\ell}]$, where $\ell$ is the layer width. This initialization strategy, and similar variants, are used almost universally for neural networks [GB10, HZRS15].
Figure 2: The effect of layer width on a 5-layer fully connected linear neural net with input dimension $d = 100$. The width across layers were kept the same. The output of the network was 1-dimensional and the squared-loss function was used. Each element of the input $x$ was distributed as $\mathcal{N}(0, 1/d)$, and each element of weight matrices were distributed as $\mathcal{N}(0, 2/(\text{fan-in} + \text{fan-out}))$, according to the Glorot normal initializer [GB10], where fan-in denotes the number of input units in the weight matrix, and fan-out denotes the number of output units in the weight matrix. As the width $\ell$ is increased from 50 to 1000, we see the mean and the variance of the gradient inner product decrease as $O(1/\ell)$, thus indicating that the gradient confusion decreases with increasing width.

**Theorem 4.2** (Neural nets with randomly chosen weights). Let $W_0, W_1, \ldots, W_\beta$ be weight matrices such that every entry in each of these matrices is an i.i.d. sample chosen uniformly from the interval, $[-1/\sqrt{\ell}, 1/\sqrt{\ell}]$, where $\ell$ is the maximum layer width. Then, for any dataset $x_1, x_2, \ldots, x_N$ with $\|x_i\| \leq 1$ for every $i \in [N]$, we have that the gradient confusion bound in eq. (3) holds with probability at least

$$1 - N^2 \exp\left(\frac{-cd(\eta - 4)^2}{64\zeta_0^4(\beta + 2)^4}\right).$$

For both the square-loss and the logistic-loss functions, the value of $\zeta_0 \leq 2\sqrt{3}$ (from Lemma D.1).

5 Effect of network layer width on gradient confusion

Consider a weight matrix $W \in \mathbb{R}^{p \times q}$ containing i.i.d. $\mathcal{N}(0, \nu^2)$ entries. Then for any vector $x \in \mathbb{R}^{q \times 1}$, we have $\mathbb{E}_W[\|Wx\|^2] = x^T \mathbb{E}_W[W^T W] x = \nu^2 p \|x\|^2$. Similarly, for any vector $y \in \mathbb{R}^{p \times 1}$, we see that $\mathbb{E}_W[\|W^T y\|^2] = \nu^2 q \|y\|^2$. Thus, a layer with weights $W$ can grow or shrink an input vector (which can be the outputs of the previous layer) on either the forward or the backward propagation at initialization time, depending on the variance $\nu^2$ of the entries in $W$.

The operator norm of the weight matrices $\|W\|$ being close to 1 is important for the trainability of neural networks, as it ensures that the signal is passed through the network without exploding or shrinking across layers [GB10]. Typical weight initialization techniques ensure that the operator norm is bounded by 1 with high probability (Appendix F). This indicates that we would expect the effect of width on convergence and gradient confusion to be much less pronounced than the effect of depth for typical neural net designs. This is also why, on assuming $\|W\| \leq 1$ for each weight
matrix $W$ in our results in Section 4, the dependence of gradient confusion on width goes away in general. A simple example that illustrates this is to consider the case where each weight matrix in the neural network has exactly one non-zero element, which is set to 1. The operator norm of each such weight matrix is 1, but the forward or backward propagated signals would not depend on the width.

That being said, it is interesting to consider how under specific weight initialization strategies, the layer width has an effect on the gradient confusion. To this end, consider a simple two-layer linear network with $g(x) := W_1 W_0 x$, where $W_0 \in \mathbb{R}^{\ell \times d}$, $W_1 \in \mathbb{R}^{1 \times \ell}$ and $x \in \mathbb{R}^d$. Further, assume that all elements of $W_1$ and $W_0$ are random, and distributed as $\mathcal{N}(0, 1/\ell)$, while all elements of $x \in \mathbb{R}^d$ are distributed as $\mathcal{N}(0, 1/d)$. Under this initialization, and on adapting Proposition A.1 for least square loss functions in [CWZ+18], we have that $\mathbb{E}_{W, x}[\langle \nabla f_i, \nabla f_j \rangle]$ is $\Theta\left(\frac{1}{\ell}\right)$, for any pair of data $x_i, x_j$. Further, using the same proof technique, we have a bound on the variance as: $\text{var}_{W, x}(\langle \nabla f_i, \nabla f_j \rangle) \leq \mathbb{E}_{W, x}[\|\nabla f_i\|^2 \|\nabla f_j\|^2] = O\left(\frac{1}{\ell^2}\right)$. Using Chebyshev’s inequality, we get:

$$\forall (i, j) \in [n], \quad \Pr[|\langle \nabla f_i, \nabla f_j \rangle| < \eta] \geq \Theta\left(1 - \frac{1}{\ell}\right). \quad (5)$$

This result shows that gradient confusion decreases with increasing width, and more generally, that the pairwise gradient inner product concentrates towards 0 with increasing width. Thus, this indicates that for wider networks, the dynamics of SGD becomes closer to decoupled gradient descent on each data sample separately. See Figure 2 for an additional simulation on a deeper neural network where the weights are initialized using the Glorot initializer [GB10].

Note that this result holds for the specific initialization strategy mentioned. Nonetheless, other recent work also provides evidence on the benefits of width, by showing that the learning dynamics of gradient descent simplify considerably for very wide networks under certain parameterizations [JGH18, LXS+19]. In the next section (and in Appendix A), we show empirical evidence that, given a sufficiently deep network, increasing the layer width often helps in lowering gradient confusion and speeding up convergence close to the minimizer for a range of neural network models.

6 Experimental results

We now present experimental results showing the effect of neural network architecture on convergence of SGD and gradient confusion. It is worth noting that Theorems 3.1 and 3.2 indicate that we would expect the effect of gradient confusion to be most prominent near the end of training, particularly when the gradient noise begins to dominate.

We performed experiments on wide residual networks (WRNs) [ZK16], convolutional networks (CNNs) and multi-layer perceptrons (MLPs) for image classification tasks on CIFAR-10, CIFAR-100 and MNIST. We present results for CNNs on CIFAR-10 in this section, and present all other results in Appendix A. We use CNN-$\beta$-$\ell$ to denote WRNs that have no skip connections or batch normalization, with a depth $\beta$ and width factor $\ell$.³ We turned off dropout and weight decay for all

³Width factor is the number of filters relative to the original ResNet model.
Figure 3: The effect of network depth with CNN-β-2 on CIFAR-10. Left plot: convergence curves of SGD, Middle plot: minimum of pairwise gradient cosine similarities at the end of training, Right plot: kernel density estimate of the pairwise gradient cosine similarities at the end of training (over all independent runs).

Figure 4: The effect of width with CNN-16-ℓ on CIFAR-10. Left plot: convergence curves of SGD (for cleaner figures, we plot results for width factors 2, 4 and 6 here), Middle plot: minimum of pairwise gradient cosine similarities at the end of training, Right plot: kernel density estimate of the pairwise gradient cosine similarities at the end of training (over all independent runs).

Our experiments. We used SGD as the optimizer without any momentum. Following [ZK16], we ran all experiments for 200 epochs with minibatches of size 128, and reduced the initial learning rate by a factor of 10 at epochs 80 and 160. We used the MSRA initializer [HZRS15] for the weights as is standard for this model, and used the same preprocessing steps for the CIFAR-10 images as described in [ZK16]. We ran each experiment 5 times, and we show the standard deviation across runs in our plots. We tuned the optimal initial learning rate for each model over a logarithmically-spaced grid and selected the run that achieved the lowest training loss value. To measure gradient confusion, at the end of every training epoch, we sampled 100 pairs of mini-batches each of size 128 (the same size as the training batch size). We calculated gradients on each mini-batch, and then computed pairwise cosine similarities. See Appendix A.2 for more details on the experimental setup and architectures used.

Effect of depth. To test our theoretical results, we consider CNNs with a fixed width factor of 2 and varying network depth. From Figure 3, we see that our theoretical results are backed by the experiments: increasing depth slows down convergence, and increases gradient confusion. We also
notice that with increasing depth, the density of pairwise gradient cosine similarities concentrates less sharply around 0 (indicating higher variance), which makes the network harder to train.

**Effect of width.** We now consider CNNs with a fixed depth of 16 and varying width factors. From Figure 4, we see that increasing width results in faster convergence and lower gradient confusion. We further see that the gradient cosine similarities concentrate around 0 with growing width, indicating that SGD becomes more deterministic with growing width. Note that the smallest network considered here is still overparameterized and achieves a high level of performance (see Appendix A.3).

**Effect of BatchNorm and skip connections.** To help understand why many standard deep nets are so efficiently trained using SGD, we test the effect of adding skip connections and batch normalization to CNNs of fixed width and varying depth. Figure 5 shows that adding skip connections or batch normalization individually help in training deeper models, but these models still suffer from worsening results and increasing gradient confusion as the network gets deeper. Both these techniques together keep the gradient confusion relatively low even for very deep networks, significantly improving trainability of deep models. Note that these observations are consistent with previous work [BFL+17, STIM18, YPR+19].

7 **Additional discussion of some related work**

The convergence of SGD on over-parameterized models has received a lot of attention. The authors of [ACH18] study the behavior of SGD on over-parameterized problems, and show that SGD on over-parameterized linear neural nets is similar to applying a certain preconditioner while optimizing. This can sometimes lead to acceleration when overparameterizing by increasing the depth of linear neural networks. In this paper, we show that this property does not hold in general (as mentioned briefly in [ACH18]), and that convergence typically slows down because of gradient confusion when training very deep networks.

The behavior of SGD on over-parameterized problems was also studied in [MBB17, BBM18,
VBS18, SR13], using an overfitting condition observed for some overparameterized neural nets (particularly for convnets), where the minimizer returned by the optimizer simultaneously minimizes each individual data sample, to show fast convergence of SGD. In contrast, we aim to establish a more direct relationship between width, depth, problem dimensionality, and the error floor of SGD convergence.

Other works have studied the impact of structured gradients on SGD. [BFL\textsuperscript{+}17] study the effects of shattered gradients at initialization for ReLU networks, which is when (non-stochastic) gradients at different (but close) locations in parameter space become negatively correlated. The authors show how gradients get increasingly shattered with depth in ReLU networks.

[Han18] show that the variance of gradients in fully connected networks with ReLU activations is exponential in the sum of the reciprocals of the hidden layer widths at initialization. Further, [HR18] show that this sum of the reciprocals of the hidden layer widths determines the variance of the sizes of the activations at each layer during initialization. When this sum of reciprocals is too large, early training dynamics are very slow, suggesting the difficulties of starting training on deeper networks, as well as the benefits of increased width.

In [YPL\textsuperscript{+}17], the authors define a property called gradient diversity. This quantity is related to gradient confusion, but with important differences. Gradient diversity also measures the degree to which individual gradients at different data samples are different from each other. This measure gets larger as the individual gradients become orthogonal to each other, and further increases as the gradients start pointing in opposite directions. In a large batch, higher gradient diversity is desirable, and this leads to improved convergence rates in distributed settings, as shown in [YPL\textsuperscript{+}17]. On the other hand, gradient confusion between two individual gradients is zero unless the inner product between them is negative. This is useful for studying convergence rates of small minibatch SGD since possible gradient updates do not conflict unless they are negatively correlated with each other. The choice of the definition of gradient diversity in [YPL\textsuperscript{+}17] also has important implications when its behavior is studied in over-parameterized settings. [CWZ\textsuperscript{+}18] extends the work of [YPL\textsuperscript{+}17], where the authors prove on 2-layer neural nets (and multi-layer linear neural nets) that gradient diversity increases with increased width and decreased depth. This does not however distinguish between the cases where gradients become more orthogonal vs. more negatively correlated. In this paper, we show that this can have very different effects on the convergence of SGD in overparameterized settings, and we view our works as complementary.

There has recently also been interest in analyzing conditions under which SGD converges to global minimizers of overparameterized linear and non-linear neural networks. [ACGH18] shows SGD converges linearly to global minimizers for linear neural nets under certain conditions. [DZPS18, AZLS18, ZCZG18, BGMSS17] also show convergence to global minimizers of SGD for non-linear neural nets. While all these results require the network to be sufficiently wide, they represent an important step in the direction of better understanding optimization on neural nets. This paper complements these recent results by studying how one property, low gradient confusion, contributes to SGD’s success on overparameterized neural nets.
8 Conclusions, limitations and future work

In this paper, we investigate how overparameterization affects the dynamics of SGD on neural networks. We introduce a concept called gradient confusion, and show that when gradient confusion is low, SGD has better convergence properties than predicted by classical theory. Further, using both theoretical and empirical results, we show that overparameterization by increasing the number of parameters of linear models or by increasing the width of neural network layers leads to lower gradient confusion, making the models easier to train. In contrast, overparameterization by increasing the depth of neural networks results in higher gradient confusion, making deeper models harder to train. We further show evidence of how techniques like batch normalization and skip connections in residual networks help in tackling this problem.

Note that many previous results have shown how deeper models are better at modeling higher complexity function classes than wider models, and thus depth is essential for the success of neural networks [ES16, Tel16, RPK+17]. Thus, our results indicate that, given a sufficiently deep network, increasing the network width is important for the trainability of the model, and will lead to faster convergence rates. This is further supported by other recent research [Han18, HR18] that show that the width should increase linearly with depth in a neural network to help dynamics at the beginning of training (i.e., at initialization). Our results also suggest the importance of further investigation into good initialization schemes for neural networks that make training very deep models possible. See [ZDM19] for some recent advances in this direction.

We consider the main limitation of this work to be the use of the random dataset that consists of i.i.d. samples of isoperimetric vectors to derive the concentration bounds in Section 4.1. Note that while this data model helps make the analysis tractable and gives insights into the effect of overparameterization, it is not clear to what extent real-world datasets follow such a model (i.e., i.i.d. data and isoperimetric property in the feature space). Nonetheless, these are standard assumptions used in learning theory since it allows for mathematical analysis. To make our observations on this model robust, we show that the same qualitative results hold (i) with arbitrary data and random weights in Section 4.2 (i.e., at initialization), and (ii) in the experimental sections (Section 6 and Appendix A). For future work, it would be interesting to extend this analysis to richer random data models. It would also be interesting to gain a better understanding of how layer width impacts long-term training dynamics, while the analysis here focuses on behavior near initialization.

An active area of research currently is to better understand how overparameterization and neural net architecture promote generalization [NLB+18, AZLL18, BHMM18, NMB+18]. An interesting topic for future work is whether there is a connection between gradient confusion, sharp and flat minimizers, and generalization for SGD. See [FNN19] for some recent work in this direction.

Acknowledgements

The authors thank Brendan O’Donoghue, James Martens, Sudha Rao, and Sam Smith for helpful discussions and for reviewing earlier versions of this manuscript.
References

[ACGH18] Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. A convergence analysis of gradient descent for deep linear neural networks. *arXiv preprint arXiv:1810.02281*, 2018.

[ACH18] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *arXiv preprint arXiv:1802.06509*, 2018.

[AZLL18] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

[AZLS18] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. *arXiv preprint arXiv:1811.03962*, 2018.

[BBM18] Raef Bassily, Mikhail Belkin, and Siyuan Ma. On exponential convergence of sgd in non-convex over-parametrized learning. *arXiv preprint arXiv:1811.02564*, 2018.

[Ben12] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[Ber11] Dimitri P Bertsekas. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning*, 2010(1-38):3, 2011.

[BFL+17] David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? *arXiv preprint arXiv:1702.08591*, 2017.

[BGMSS17] Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. Sgd learns over-parameterized networks that provably generalize on linearly separable data. *arXiv preprint arXiv:1710.10174*, 2017.

[BHMM18] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine learning and the bias-variance trade-off. *arXiv preprint arXiv:1812.11118*, 2018.

[BLM13] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

[BSF94] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[CCS+16] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *arXiv preprint arXiv:1611.01838*, 2016.
[Coo18] Y Cooper. The loss landscape of overparameterized neural networks. *arXiv preprint arXiv:1804.10200*, 2018.

[CT18] Jerry Chee and Panos Toulis. Convergence diagnostics for stochastic gradient descent with constant learning rate. In *International Conference on Artificial Intelligence and Statistics*, pages 1476–1485, 2018.

[CWZ+18] Lingjiao Chen, Hongyi Wang, Jinman Zhao, Dimitris Papailiopoulos, and Paraschos Koutris. The effect of network width on the performance of large-batch training. *arXiv preprint arXiv:1806.03791*, 2018.

[DFB17] Aymeric Dieuleveut, Nicolas Flammarion, and Francis Bach. Harder, better, faster, stronger convergence rates for least-squares regression. *The Journal of Machine Learning Research*, 18(1):3520–3570, 2017.

[DM92] Christian Darken and John Moody. Towards faster stochastic gradient search. In *Advances in neural information processing systems*, pages 1009–1016, 1992.

[DR17] Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

[DYZG17] Soham De, Abhay Yadav, David Jacobs, and Tom Goldstein. Automated inference with adaptive batches. In *Artificial Intelligence and Statistics*, pages 1504–1513, 2017.

[DFP18] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[ES16] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *Conference on Learning Theory*, pages 907–940, 2016.

[FB15] Nicolas Flammarion and Francis Bach. From averaging to acceleration, there is only a step-size. In *Conference on Learning Theory*, pages 658–695, 2015.

[FNN19] Stanislaw Fort, Paweł Krzysztof Nowak, and Srin Narayanan. Stiffness: A new perspective on generalization in neural networks. *arXiv preprint arXiv:1901.09491*, 2019.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[Gra06] Robert M Gray. Toeplitz and circulant matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3):155–239, 2006.

[Han18] Boris Hanin. Which neural net architectures give rise to exploding and vanishing gradients? In *Advances in Neural Information Processing Systems*, pages 582–591, 2018.
[HR18] Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems*, pages 571–581, 2018.

[HZRS15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8580–8589, 2018.

[KNS16] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-lojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016.

[KS17] Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*, 2017.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[LBOM12] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[Loj65] Stanislaw Lojasiewicz. Ensembles semi-analytiques. *Lectures Notes IHES (Bures-sur-Yvette)*, 1965.

[LXS+19] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

[MB11] Eric Moulines and Francis R Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.

[MBB17] Siyuan Ma, Raef Bassily, and Mikhail Belkin. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. *arXiv preprint arXiv:1712.06559*, 2017.
[MS86] Vitali D Milman and Gideon Schechtman. *Asymptotic Theory of Finite Dimensional Normed Spaces*. Springer-Verlag, Berlin, Heidelberg, 1986.

[Mur98] Noboru Murata. A statistical study of on-line learning. *Online Learning and Neural Networks. Cambridge University Press, Cambridge, UK*, pages 63–92, 1998.

[NB01] Angelia Nedić and Dimitri Bertsekas. Convergence rate of incremental subgradient algorithms. In *Stochastic optimization: algorithms and applications*, pages 223–264. Springer, 2001.

[Nes83] Yuriii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k^2). 1983.

[NH17] Quynh Nguyen and Matthias Hein. The loss surface of deep and wide neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2603–2612. JMLR. org, 2017.

[NK19] Vaishnavh Nagarajan and J Zico Kolter. Generalization in deep networks: The role of distance from initialization. *arXiv preprint arXiv:1901.01672*, 2019.

[NLB+18] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-paramterization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*, 2018.

[NMB+18] Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tantia, Matthew Scicluna, Simon Lacoste-Julien, and Ioannis Mitliagkas. A modern take on the bias-variance tradeoff in neural networks. *arXiv preprint arXiv:1810.08591*, 2018.

[NWS14] Deanna Needell, Rachel Ward, and Nati Srebro. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Advances in Neural Information Processing Systems*, pages 1017–1025, 2014.

[OS18] Samet Oymak and Mahdi Soltanolkotabi. Overparameterized nonlinear learning: Gradient descent takes the shortest path? *arXiv preprint arXiv:1812.10004*, 2018.

[Pol64] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics, 4*(5):1–17, 1964.

[RM51] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[RPK+17] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl Dickstein. On the expressive power of deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2847–2854. JMLR. org, 2017.

[SEG+17] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.
Hanie Sedghi, Vineet Gupta, and Philip M Long. The singular values of convolutional layers. *arXiv preprint arXiv:1805.10408*, 2018.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909, 2016.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

Mark Schmidt and Nicolas Le Roux. Fast convergence of stochastic gradient descent under a strong growth condition. *arXiv preprint arXiv:1308.6370*, 2013.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? (no, it is not about internal covariate shift). *arXiv preprint arXiv:1805.11604*, 2018.

Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pages 71–79, 2013.

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

Panos Toulis, Edoardo M Airoldi, et al. Asymptotic and finite-sample properties of estimators based on stochastic gradients. *The Annals of Statistics*, 45(4):1694–1727, 2017.

Terence Tao. *Topics in random matrix theory*, volume 132. American Mathematical Soc., 2012.

Matus Telgarsky. Benefits of depth in neural networks. *arXiv preprint arXiv:1602.04485*, 2016.

Sharan Vaswani, Francis Bach, and Mark Schmidt. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. *arXiv preprint arXiv:1810.07288*, 2018.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press, 2018.

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pages 4151–4161, 2017.

Lei Wu, Zhanxing Zhu, et al. Towards understanding generalization of deep learning: Perspective of loss landscapes. *arXiv preprint arXiv:1706.10239*, 2017.
[YPL+17] Dong Yin, Ashwin Pananjady, Max Lam, Dimitris Papailiopoulos, Kannan Ramchandran, and Peter Bartlett. Gradient diversity: a key ingredient for scalable distributed learning. arXiv preprint arXiv:1706.05699, 2017.

[YPR+19] Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S Schoenholz. A mean field theory of batch normalization. arXiv preprint arXiv:1902.08129, 2019.

[ZCZG18] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. arXiv preprint arXiv:1811.08888, 2018.

[ZDM19] Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. arXiv preprint arXiv:1901.09321, 2019.

[ZK16] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. arXiv preprint arXiv:1605.07146, 2016.
Appendix

A Additional experimental results

In this section, we present more details about our experimental setup, as well as, additional experimental results on a range of models (MLPs, CNNs and Wide ResNets) and a range of datasets (MNIST, CIFAR-10, CIFAR-100).

A.1 MLPs on MNIST

To further test the main claims in the paper, we performed additional experiments on an image classification problem on the MNIST dataset using fully connected neural networks. We iterated over neural networks of varying depth and width, and considered both the identity activation function (i.e., linear neural networks) and the tanh activation function. We also considered two different weight initializations that are popularly used and appropriate for these activation functions:

- The Glorot normal initializer [GB10] with weights initialized by sampling from the distribution $\mathcal{N}(0, 2/(\text{fan-in} + \text{fan-out}))$, where fan-in denotes the number of input units in the weight matrix, and fan-out denotes the number of output units in the weight matrix.

- The LeCun normal initializer [LBOM12] with weights initialized by sampling from the distribution $\mathcal{N}(0, 1/\text{fan-in})$.

We considered the simplified case where all hidden layers have the same width $\ell$. Thus, the first weight matrix $W_0 \in \mathbb{R}^{\ell \times d}$, where $d = 784$ for the $28 \times 28$-sized images of MNIST; all intermediate weight matrices $\{W_p\}_{p \in [\beta-1]} \in \mathbb{R}^{\ell \times \ell}$; and the final layer $W_\beta \in \mathbb{R}^{10 \times \ell}$ for the 10 image classes in MNIST. We added biases to each layer, which we initialized to 0. We used softmax cross entropy as the loss function. We use MLP- $\beta$-$\ell$ to denote this fully connected network of depth $\beta$ and width $\ell$. We used the standard train-valid-test splits of 40000-10000-10000 for MNIST.

This relatively simple model gave us the ability to iterate over a large number of combinations of network architectures of varying width and depth, and different activation functions and weight initializations. Linear neural networks are an efficient way to directly understand the effect of changing depth and width without increasing model complexity over linear regression. Thus, we considered both linear and non-linear neural nets in our experiments.

We used SGD with constant learning rates for training with a mini-batch size of 128 and trained each model for 40000 iterations (more than 100 epochs). The constant learning rate $\alpha$ was tuned over a logarithmically-spaced grid:

$$\alpha \in \{10^0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}.$$ 

We ran each experiment 10 times (making sure at least 8 of them ran till completion), and picked the learning rate that achieved the lowest training loss value on average at the end of training. Our grid
search was such that the optimal learning rate never occurred at one of the extreme values tested.

To measure gradient confusion at the end training, we sampled 1000 pairs of mini-batches each of size 128 (the same size as the training batch size). We calculated gradients on each of these pairs of mini-batches, and then calculated the cosine similarity between them. To measure the worse-case gradient confusion, we computed the lowest gradient cosine similarity among all pairs. We explored the effect of changing depth and changing width on the different activation functions and weight initializations. We plot the final training loss achieved for each model and the minimum gradient cosine similarities calculated over the 1000 pairs of gradients at the end of training. For each point, we plot both the mean and the standard deviation over the 10 independent runs.

The effect of depth. We first present results showing the effect of network depth. We considered a fixed width of $\ell = 100$, and varied the depth of the neural network, on the log scale, as:

$$\beta \in \{3, 10, 30, 100, 300, 1000\}.$$

Figure 6 shows results on neural networks with identity and tanh activation functions for the two weight initializations considered (Glorot normal and LeCun normal). Similar to the experimental results in Section 6, and matching our theoretical results in Section 4, we notice the consistent trend of gradient confusion increasing with increasing depth. This makes the networks harder to train with increasing depth, and this is evidenced by an increase in the final training loss value. By depth $\beta = 1000$, the increased gradient confusion effectively makes the network untrainable when using tanh non-linearities.

The effect of width. We explored the effect of width by varying the width of the neural network while keeping the depth fixed at $\beta = 300$. We chose a very deep model, which is essentially untrainable for small widths (with standard initialization techniques) and helps better illustrate the effects of increasing width. We varied the width of the network, again on the log scale, as:

$$\ell \in \{10, 30, 100, 300, 1000\}.$$

Crucially, note that the smallest network considered here, MLP-300-10, still has more than 50000 parameters (i.e., more than the number of training samples), and the network with width $\ell = 30$ has almost three times the number of parameters as the high-performing MLP-3-100 network considered in the previous section. Figure 7 show results on linear neural nets and neural nets with tanh activations for both the Glorot normal and LeCun normal initializations. As in the experimental results of Section 6, we see the consistent trend of gradient confusion decreasing with increasing width. Thus, wider networks become easier to train and improve the final training loss value. We further see that when the width is too small ($\ell = 30$), the gradient confusion becomes drastically high and the network becomes completely untrainable.

A.2 Additional experimental details for CNNs and WRNs

In this section, we review the details of our setup for the image classification experiments on CNNs and WRNs on the CIFAR-10 and CIFAR-100 datasets.
Wide residual networks

The Wide ResNet (WRN) architecture [ZK16] for CIFAR datasets is a stack of three groups of residual blocks. There is a downsampling layer between two blocks, and the number of channels
(width of a convolutional layer) is doubled after downsampling. In the three groups, the width of convolutional layers is \( \{16\ell, 32\ell, 64\ell\} \), respectively. Each group contains \( \beta_r \) residual blocks, and each residual block contains two \( 3 \times 3 \) convolutional layers equipped with ReLU activation, batch normalization and dropout. There is a \( 3 \times 3 \) convolutional layer with 16 channels before the three groups of residual blocks. And there is a global average pooling, a fully-connected layer and a softmax layer after the three groups. The depth of WRN is \( \beta = 6\beta_r + 4 \).

For our experiments, we turned off dropout. Unless otherwise specified, we also turned off batch normalization. We added biases to the convolutional layers when not using batch normalization. We used the MSRA initializer [HZRS15] for the weights as is standard for this model, and used the same preprocessing steps for the CIFAR images as described in [ZK16]. This preprocessing step involves normalizing the images and doing data augmentation [ZK16]. We denote this network as WRN-\( \beta \)-\( \ell \), where \( \beta \) represents the depth and \( \ell \) represents the width factor of the network.

To study the effect of depth, we considered WRNs with width factor \( \ell = 2 \) and depth varying as:
\[
\beta \in \{16, 22, 28, 34, 40, 52, 76, 100\}.
\]

For cleaner figures, we sometimes plot a subset of these results: \( \beta \in \{16, 28, 40, 52, 76, 100\} \). To study the effect of width, we considered WRNs with depth \( \beta = 16 \) and width factor varying as:
\[
\ell \in \{2, 3, 4, 5, 6\}.
\]

**Convolutional neural nets**

The WRN architecture contains skip connections that, as we show, help in training deep networks. To consider VGG-like convolutional networks, we consider a family of networks where we remove the skip connections from WRNs. Following the WRN convention, we denote these networks as CNN-\( \beta \)-\( \ell \), where \( \beta \) denotes the depth and \( \ell \) denotes the width factor.

To study the effect of depth, we considered CNNs with width factor \( \ell = 2 \) and depth varying as:
\[
\beta \in \{16, 22, 28, 34, 40\}.
\]

To study the effect of width, we considered CNNs with depth \( \beta = 16 \) and width factor varying as:
\[
\ell \in \{2, 3, 4, 5, 6\}.
\]

**Hyperparameter tuning and other details**

We used SGD as the optimizer without any momentum. Following [ZK16], we ran all experiments for 200 epochs with minibatches of size 128, and reduced the initial learning rate by a factor of 10 at epochs 80 and 160. We turned off weight decay for all our experiments.

We ran each individual experiment 5 times. We ignored any runs that were unable to decrease the loss from its initial value. We also made sure at least 4 out of the 5 independent runs ran till
completion. When the learning rate is close to the threshold at which training is still possible, some runs may converge, while others may fail to converge. Thus, these checks ensure that we pick a learning rate that converges reliably in most cases on each problem. We show the standard deviation across runs in our plots.

We tuned the optimal initial learning rate for each model over a logarithmically-spaced grid:

$$\alpha \in \{10^1, 3 \times 10^0, 10^0, 3 \times 10^{-1}, 10^{-1}, 3 \times 10^{-2}, 10^{-2}, 3 \times 10^{-3}, 10^{-3}, 3 \times 10^{-4}, 10^{-4}, 3 \times 10^{-5}\},$$

and selected the run that achieved the lowest final training loss value (averaged over the independent runs). Our grid search was such that the optimal learning rate never occurred at one of the extreme values tested. We used the standard train-valid-test splits of 40000-10000-10000 for CIFAR-10 and CIFAR-100.

To measure gradient confusion, at the end of every training epoch, we sampled 100 pairs of mini-batches each of size 128 (the same size as the training batch size). We calculated gradients on each mini-batch, and then computed pairwise cosine similarities. To measure the worse-case gradient confusion, we computed the lowest gradient cosine similarity among all pairs. We also show the kernel density estimation of the pairwise gradient cosine similarities of the 100 minibatches sampled at the end of training (after 200 epochs), to see the concentration of the distribution. To do this, we combine together the 100 samples for each independent run and then perform kernel density estimation with a gaussian kernel on this data.

### A.3 Additional plots for CIFAR-10 on CNNs

In Section 6, we showed results for image classification using CNNs on CIFAR-10. In this section, we show some additional plots for this experiment. Figure 8 shows the effect of changing the depth, while Figure 9 shows the effect of changing the width factor of the CNN. We see that the final training loss and test set accuracy values show the same trends as in Section 6: deeper networks are harder to train, while wider networks are easier to train. As mentioned previously, Theorems 3.1 and 3.2 indicate that we would expect the effect of gradient confusion to be most prominent near the end of training, particularly when the gradient noise begins to dominate. From the plots we see that deeper networks have higher gradient confusion close to minimum, while wider networks have lower gradient confusion close to the minimum.

### A.4 CIFAR-100 on CNNs

We now consider image classifications tasks with CNNs on the CIFAR-100 dataset. Figure 10 shows the effect of varying depth, while Figure 11 shows the effect of varying width. We notice the same trends as in our results with CNNs on CIFAR-10. Interestingly, from the width results in Figure 11, we see that while there is no perceptible change to the minimum pairwise gradient cosine similarity, the distribution still sharply concentrates around 0 with increasing width. Thus more gradients become orthogonal to each other with increasing width, implying that SGD on very
Figure 8: The effect of network depth with CNN-\(\beta\)-2 on CIFAR-10. *Left plot:* final training loss values at the end of training. *Middle plot:* final test set accuracy values at the end of training. *Right plot:* curves showing the minimum of pairwise gradient cosine similarities during training.

Figure 9: The effect of width with CNN-16-\(\ell\) on CIFAR-10. *Left plot:* final training loss values at the end of training. *Middle plot:* final test set accuracy values at the end of training. *Right plot:* curves showing the minimum of pairwise gradient cosine similarities during training.

wide networks becomes closer to performing deterministic gradient descent on each individual data sample.

### A.5 Image classification with WRNs on CIFAR-10 and CIFAR-100

We now show results for image classification problems using wide residual networks (WRNs) on CIFAR-10 and CIFAR-100. The WRNs we consider do not have any batch normalization. Later we show results on the effect of adding batch normalization to these networks.

Figures 12 and 13 show results on the effect of depth using WRNs on CIFAR-10 and CIFAR-100 respectively. We again see the consistent trend of deeper networks having higher gradient confusion, making them harder to train. We further see that increasing depth results in the pairwise gradient cosine similarities concentrating less around 0, and leading to higher variance.

Figures 14 and 15 show results on the effect of width using WRNs on CIFAR-10 and CIFAR-100 respectively. We see that increasing width typically lowers gradient confusion and helps the network achieve lower loss values. The pairwise gradient cosine similarities also typically concentrate...
around 0 with higher width. We also notice from these figures that in some cases, increasing width might lead to diminishing returns, i.e., the benefits of increased width diminish after a certain point, as one would expect.

**A.6 Effect of batch normalization**

In Section 6 we showed results on the effect of adding batch normalization to CNNs and WRNs on an image classification task on CIFAR-10. In this section, we present similar results for image classification on CIFAR-100. Similar to Section 6, Figure 16 shows that adding skip connections or batch normalization individually help in training deeper models, but these models still suffer from worsening results and increasing gradient confusion as the network gets deeper. Both these techniques together keep the gradient confusion relatively low even for very deep networks, significantly improving trainability of deep models.
Figure 12: The effect of depth with WRN-β-2 (no batch normalization) on CIFAR-10. Left plot: training loss values at the end of training. Middle plot: minimum of pairwise gradient cosine similarities at the end of training. Right plot: kernel density estimate of the pairwise gradient cosine similarities at the end of training.

Figure 13: The effect of depth with WRN-β-2 (no batch normalization) on CIFAR-100. Left plot: training loss values at the end of training. Middle plot: minimum of pairwise gradient cosine similarities at the end of training. Right plot: kernel density estimate of the pairwise gradient cosine similarities at the end of training.

B Near orthogonality of random vectors

For completeness, we prove below a lemma on the near orthogonality of random vectors. This result is often attributed to [MS86].

**Lemma 2.1** (Near orthogonality of random vectors). For vectors \( \{x_i\}_{i \in [N]} \) drawn uniformly from a unit sphere in \( d \) dimensions, and \( \nu > 0 \),

\[
\Pr \left[ \max_{ij} |x_i^\top x_j| > \nu \right] \leq N^2 \sqrt{\frac{\pi}{8}} \exp \left( - \frac{d}{2} - \frac{1}{2} \nu^2 \right).
\]

*Proof.* Given a fixed vector \( x \), a uniform random vector \( y \) satisfies \( |x^\top y| \geq \nu \) only if \( y \) lies in one of two spherical caps: one centered at \( x \) and the other at \( -x \), and both with angular radius \( \cos^{-1}(\nu) \leq \pi/2 - \nu \). A simple result often attributed to Milman and Schechtman [MS86] bounds the
Figure 14: The effect of width with WRN-16-\(\ell\) (no batch normalization) on CIFAR-10. **Left plot:** training loss values at the end of training. **Middle plot:** minimum of pairwise gradient cosine similarities at the end of training. **Right plot:** kernel density estimate of the pairwise gradient cosine similarities at the end of training.

Figure 15: The effect of width with WRN-16-\(\ell\) (no batch normalization) on CIFAR-100. **Left plot:** training loss values at the end of training. **Middle plot:** minimum of pairwise gradient cosine similarities at the end of training. **Right plot:** kernel density estimate of the pairwise gradient cosine similarities at the end of training.

The probability of lying in either of these caps as

\[
\Pr[|x^\top y| \geq \nu] \leq \sqrt{\frac{\pi}{2}} \exp \left(-\frac{d-1}{2} \nu^2 \right). \tag{6}
\]

Because of rotational symmetry, the bound (6) holds if both \(x\) and \(y\) are chosen uniformly at random.

We next apply a union bound to control the probability that \(|x_i^\top x_j| \geq \nu\) for some pair \((i, j)\). There are fewer than \(N^2/2\) such pairs, and so the probability of this condition is

\[
\Pr[|x_i^\top x_j| \geq \nu, \text{ for all } i, j] \leq \frac{N^2}{2} \sqrt{\frac{\pi}{2}} \exp \left(-\frac{d-1}{2} \nu^2 \right).
\]
C Conditions for faster convergence

In Theorem 3.1, we show convergence to the neighborhood of a minimizer for problems satisfying the PL inequality and Lipschitz-smoothness. The learning rate for which the rate of convergence in the transient phase is optimal is given by $\alpha = \frac{1}{NL}$, with a corresponding decay rate of $\rho = 1 - \frac{\mu}{NL}$.

Faster convergence can be guaranteed if we strengthen the definition of confusion by examining the correlation between $\nabla f_i(w)$ and $\nabla f_j(w')$ for all $w$ and $w'$. Compared to Theorem 3.1, convergence is guaranteed with a larger learning rate that is independent of the training set size $N$, and faster geometric decay.

**Theorem C.1.** If the objective function satisfies A1 and A2, and satisfies the strengthened gradient confusion bound

$$\langle \nabla f_i(w), \nabla f_j(w') \rangle \geq -\eta, \; \forall i, j, w, w',$$

then SGD converges with

$$F(w_k) - F^* \leq \rho^k(F(w_0) - F^*) + \frac{\alpha\eta}{1 - \rho},$$

where the learning rate $\alpha < 2/L$ and $\rho = 1 - 2\mu\alpha/N + \mu L\alpha^2/N$.

**Proof.** We start by noting that, for $i \neq j$

$$f_i(w - \alpha \nabla f_j(w)) = f_i(w) + \int_{t=0}^{\alpha} \frac{\partial}{\partial t} f_i(w - t \nabla f_j(w)) \, dt$$

$$= f_i(w) - \int_{t=0}^{\alpha} \nabla f_j(w)^\top f_i(w - t \nabla f_j(w)) \, dt$$

$$\leq f_i(w) + \int_{t=0}^{\alpha} \eta \, dt$$

$$\leq f_i(w) + \alpha \eta.$$
We then have
\[
NF(w_{k+1}) = \tilde{f}_k(w^k - \alpha \nabla \tilde{f}_k(w^k)) + \sum_{i \neq \tilde{k}} f_i(w^k - \alpha \nabla \tilde{f}_k(w^k)) \\
\leq \tilde{f}_k(w^k) - \alpha \langle \nabla \tilde{f}_k(w^k), \nabla \tilde{f}_k(w^k) \rangle + \frac{L\alpha^2}{2} \| \nabla \tilde{f}_k(w^k) \|^2 + \sum_{i \neq \tilde{k}} f_i(w^k) + \alpha \eta \\
\leq NF(w^k) - \left( \alpha - \frac{L\alpha^2}{2} \right) \| \nabla \tilde{f}_k(w^k) \|^2 + N\alpha \eta.
\]

Re-arranging and applying A2 we get
\[
F(w_{k+1}) \leq F(w^k) - \frac{2\mu}{N} \left( \alpha - \frac{L\alpha^2}{2} \right) (\tilde{f}_k(w^k) - \tilde{f}^*_k) + \alpha \eta.
\]
Taking expectations, subtracting \( F^* \) from both sides, and using \( E_i[f_i^*] \leq F^* \), we get
\[
E[F(w_{k+1}) - F^*] \leq \left( 1 - \frac{2\mu \alpha}{N} + \frac{\mu L\alpha^2}{N} \right) E[F(w_k) - F^*] + \alpha \eta.
\]
Unrolling this expression gives us
\[
E[F(w_{k+1}) - F^*] \leq \rho^{k+1}(F(w_0) - F^*) + \frac{\alpha \eta}{1 - \rho},
\]
where \( \rho = 1 - 2\mu \alpha / N + \mu L\alpha^2 / N \), which completes the proof.

Note that with the strengthened gradient confusion bound, under the optimal learning rate of \( \alpha = 1/L \), the decay rate for the linear convergence phase \( \rho \) is given by \( \rho = 1 - \frac{\mu}{NL} \). This is a much faster rate compared to Theorem 3.1, which under the optimal learning rate of \( \alpha = \frac{1}{NL} \) gives a decay rate of \( \rho = 1 - \frac{\mu}{NL} \).

How likely is such a strengthened gradient confusion bound to hold? Note that the informal analysis on low-rank Hessians in Section 2 can be extended to show that this strengthened gradient confusion bound (eq. (7)) would be likely to be small for all \( w, w' \) within a neighborhood of the minimizer. In this case, the inner product between two gradients at weights \( w \) and \( w' \) is:
\[
|\langle \nabla f_i(w), \nabla f_j(w') \rangle| \\
\leq \| w \| \| w' \| \| H_i \| \| H_j \| + \frac{1}{2} L_H (\| H_i \| \| w \| \| w' \|)^2 + \| H_j \| \| w \|^2 \| w' \| + \frac{1}{4} L_H^2 \| w \|^2 \| w' \|^2.
\]

Using the same arguments as before, we would expect the terms in this expression to be small if the Hessians are sufficiently random and low-rank (e.g., of the form \( H_i = a_i a_i^T \) where \( a_i \in \mathbb{R}^{N \times r} \) are randomly sampled from a unit sphere), for all \( w, w' \) within a neighborhood of the minimizer. As mentioned before, there is evidence that the Hessian at the minimizer is very low rank for many standard overparameterized neural net models [SEG+17, Coo18, CCS+16, WZ+17]. This indicates that the strengthened gradient confusion bound, and thus a faster rate of convergence of SGD, may indeed hold for many standard problems.
D Missing proofs

D.1 Proofs of Theorems 3.1 and 3.2

This section presents proofs for the convergence theorems of SGD presented in Section 3, under the assumption of low gradient confusion. For clarity of presentation, we re-state each theorem before its proof.

**Theorem 3.1.** If the objective function satisfies A1 and A2, and has gradient confusion bound \( \eta \) (eq. 3), SGD with updates of the form (2) converges linearly to a neighborhood of the minima of the problem (1) as:

\[
E[F(w_k) - F^*] \leq \rho^k (F(w_0) - F^*) + \frac{\alpha \eta}{1 - \rho},
\]

where \( \alpha < 2/NL \), \( \rho = 1 - \frac{2 \mu}{N} \left( \alpha - \frac{NL \alpha^2}{2} \right) \) and \( F^* = \min_w F(w) \).

**Proof.** Let \( \tilde{i} \in [N] \) denote the index of the realized function \( \tilde{f}_k \) in the uniform sampling from \( \{ f_i \}_{i \in [N]} \) at step \( k \). From Assumption (A1), we have

\[
F(w_{k+1}) \leq F(w_k) + \langle \nabla F(w_k), w_{k+1} - w_k \rangle + \frac{L}{2} \| w_{k+1} - w_k \|^2
\]

\[
= F(w_k) - \alpha \langle \nabla F(w_k), \nabla \tilde{f}_k(w_k) \rangle + \frac{L \alpha^2}{2} \| \nabla \tilde{f}_k(w_k) \|^2
\]

\[
= F(w_k) - \left( \frac{\alpha}{N} - \frac{L \alpha^2}{2} \right) \| \nabla \tilde{f}_k(w_k) \|^2 - \frac{\alpha}{N} \sum_{i \neq \tilde{i}} \langle \nabla f_i(w_k), \nabla \tilde{f}_k(w_k) \rangle
\]

\[
\leq F(w_k) - \left( \frac{\alpha}{N} - \frac{L \alpha^2}{2} \right) \| \nabla \tilde{f}_k(w_k) \|^2 + \frac{\alpha(N - 1) \eta}{N},
\]

\[
\leq F(w_k) - \left( \frac{\alpha}{N} - \frac{L \alpha^2}{2} \right) \| \nabla \tilde{f}_k(w_k) \|^2 + \alpha \eta,
\]

where the second-last inequality follows from Definition 2.1. Let the learning rate \( \alpha < 2/NL \). Then, using Assumption (A2) and subtracting by \( F^* = \min_w F(w) \) on both sides, we get

\[
F(w_{k+1}) - F^* \leq F(w_k) - F^* - 2 \mu \left( \frac{\alpha}{N} - \frac{L \alpha^2}{2} \right) (\tilde{f}_k(w_k) - \tilde{f}_k^*) + \alpha \eta,
\]

where \( \tilde{f}_k^* = \min_w \tilde{f}_k(w) \). It is easy to see that by definition we have, \( \mathbb{E}_i[f_i^*] \leq F^* \). Moreover, from assumption that \( \alpha < \frac{2}{NL} \), it implies that \( \left( \frac{\alpha}{N} - \frac{L \alpha^2}{2} \right) > 0 \). Therefore, taking expectation on both sides we get,

\[
\mathbb{E}[F(w_{k+1}) - F^*] \leq \left( 1 - \frac{2 \mu \alpha}{N} + \mu L \alpha^2 \right) \mathbb{E}[F(w_k) - F^*] + \alpha \eta.
\]

33
Writing $\rho = 1 - \frac{2\mu\alpha}{N} + \mu L\alpha^2$, and unrolling the iterations, we get

\[
\mathbb{E}[F(\mathbf{w}_{k+1}) - F^*] \leq \rho^{k+1}(F(\mathbf{w}_0) - F^*) + \sum_{i=0}^{k} \rho^i \alpha \eta
\]

\[
\leq \rho^{k+1}(F(\mathbf{w}_0) - F^*) + \sum_{i=0}^{\infty} \rho^i \alpha \eta
\]

\[
= \rho^{k+1}(F(\mathbf{w}_0) - F^*) + \frac{\alpha \eta}{1 - \rho}.
\]

\[\square\]

**Theorem 3.2.** If the objective satisfies A1 and the gradient confusion bound (eq. 3), then SGD converges to a neighborhood of a stationary point as:

\[
\min_{k=1, \ldots, T} \mathbb{E} \|\nabla F(\mathbf{w}_k)\|^2 \leq \frac{\rho(F(\mathbf{w}_1) - F^*)}{T} + \alpha \rho \eta,
\]

for learning rate $\alpha < 2/(NL)$, $\rho = \frac{2N}{2\alpha - NL\alpha^2}$, and $F^* = \min_w F(w)$.

**Proof.** From Theorem 3.1, we have:

\[
F(\mathbf{w}_{k+1}) \leq F(\mathbf{w}_k) - \left(\frac{\alpha}{N} - \frac{L\alpha^2}{2}\right) \|\nabla \tilde{f}_k(\mathbf{w}_k)\|^2 + \alpha \eta.
\]

(8)

Now we know that:

\[
\mathbb{E}\|\nabla \tilde{f}_k(\mathbf{w}_k)\|^2 = \mathbb{E}\|\nabla \tilde{f}_k(\mathbf{w}_k) - \nabla F(\mathbf{w}_k)\|^2 + \mathbb{E}\|\nabla F(\mathbf{w}_k)\|^2 \geq \mathbb{E}\|\nabla F(\mathbf{w}_k)\|^2.
\]

Thus, taking expectation and assuming the step size $\alpha < 2/(NL)$, we can rewrite eq. (8) as:

\[
\mathbb{E}\|\nabla F(\mathbf{w}_k)\|^2 \leq \frac{2N}{2\alpha - NL\alpha^2} \mathbb{E}[F(\mathbf{w}_k) - F(\mathbf{w}_{k+1})] + \frac{2N \eta}{2 - NL\alpha}.
\]

Taking an average over $T$ iterations, and using $F^* = \min_w F(w)$, we get:

\[
\min_{k=1, \ldots, T} \mathbb{E}\|\nabla F(\mathbf{w}_k)\|^2 \leq \frac{1}{T} \sum_{k=1}^{T} \mathbb{E}\|\nabla F(\mathbf{w}_k)\|^2 \leq \frac{2N}{2\alpha - NL\alpha^2} \frac{F(\mathbf{w}_1) - F^*}{T} + \frac{2N \eta}{2 - NL\alpha}.
\]

\[\square\]

**D.2 Proofs of helper lemmas**

**Lemma D.1.** Consider the set of loss-functions $\{f_i(W)\}_{i \in [N]}$ where all $f_i$ are either the square-loss function or the logistic-loss function. Consider a feed-forward neural network as defined in eq. (4) whose weights $W$ satisfy Assumption 4.1,. Consider the gradient $\nabla_W f_i(W)$ of each function $f_i$. Note that we can write $\nabla_W f_i(W) = \zeta_{x_i}(W) \nabla_W g(W)(x_i)$, where we define $\zeta_{x_i}(W) = \partial f_i(W)/\partial g_W$. Then we have the following properties.
1. When \( \|x\| \leq 1 \) we have \( \|\nabla_W g_W(x)\| \leq 1 \).

2. There exists a constant \( \zeta_0 > 0 \) such that \( |\zeta_{x_i}(W)| \leq \zeta_0 \), \( \|\nabla_{x_i} \zeta_{x_i}(W)\|_2 \leq \zeta_0 \), \( \|\nabla_W \zeta_{x_i}(W)\|_2 \leq \zeta_0 \).

We show that \( \zeta_0 \leq 2\sqrt{\beta} \), where \( \beta \) is the depth of the neural network.

**Proof.** Let \( W \) denote the tuple \( (W_p)_{p \in [\beta]_0} \). Consider \( |\zeta_{x_i}(W)| = |f'(W)| \). In the case of square-loss function this evaluates to \( |g_W(x) - C(x)| \leq 2 \). In case of logistic regression, this evaluates to \( \frac{-1}{1 + \exp(C(x)g_W(x))} \leq 1 \). Now we consider \( \|\nabla_{x_i} \zeta_{x_i}(W)\| \). Consider the squared loss function. We then have the following.

\[
\|\nabla_{x_i} \zeta_{x_i}(W)\| = \|\nabla_{x_i} f'(W)\| = \|\nabla_{x_i} g_W(x_i) - C(x_i)\| \leq \|\nabla_{x_i} g_W(x_i)\| + 1.
\]

Likewise, consider the logistic-loss function. We then have the following.

\[
\|\nabla_{x_i} \zeta_{x_i}(W)\| \leq \left\| \frac{C(x_i)^2}{(1 + \exp(C(x_i)g_W(x_i)))^2} \exp(C(x_i)g_W(x_i)) \right\| \|\nabla_{x_i} g_W(x_i)\| \leq \|\nabla_{x_i} g_W(x_i)\|.
\]

Thus, it suffices to bound \( \|\nabla_{x_i} g_W(x_i)\| \). Using Assumption 4.1 and the properties (P1), (P2) of \( \sigma \), this can be upper-bounded by 1.

Consider \( \nabla_{W_p} \zeta_{x_i}(W) \) for some layer index \( p \in [\beta]_0 \). We will show that \( \|\nabla_{W_p} \zeta_{x_i}(W)\|_2 \leq 2 \). Then it immediately follows that \( \|\nabla_W \zeta_{x_i}(W)\|_2 \leq 2\sqrt{\beta} \). In the case of a squared loss function. We have the following.

\[
\|\nabla_{W_p} \zeta_{x_i}(W)\| = \|\nabla_{W_p} f'(W)\| = \|\nabla_{W_p} g_W(x_i) - C(x_i)\| \leq \|\nabla_{W_p} g_W(x_i)\| + 1.
\]

Likewise, consider the logistic-loss function. We then have the following.

\[
\|\nabla_{W_p} \zeta_{x_i}(W)\| \leq \left\| \frac{C(x_i)^2}{(1 + \exp(C(x_i)g_W(x_i)))^2} \exp(C(x_i)g_W(x_i)) \right\| \|\nabla_{W_p} g_W(x_i)\| \leq \|\nabla_{W_p} g_W(x_i)\|.
\]

Since \( \|\nabla_{W_p} g_W(x_i)\| \leq 1 \), we have that \( \|\nabla_{W_p} \zeta_{x_i}(W)\| \leq 2 \) in both the cases. Thus, \( \zeta_0 = 2\sqrt{\beta} \).
D.3 Proofs of Theorem 4.1 and Corollary 4.1

In this section, we will analyze the case when we have non-linear activations at each layer.

Theorem 4.1 (Concentration bounds for arbitrary depth neural networks). Consider the problem of fitting neural networks (eq. 4) to data using either the square-loss or the logistic-loss function. Let $\eta > 0$ be a given constant. Let the weights satisfy Assumption 4.1 and the non-linearities in each layer satisfy properties (P1) and (P2). For some fixed constant $c > 0$, with probability at least

$$1 - N^2 \exp \left( \frac{-cd\eta^2}{16\zeta_0^4(\beta + 2)^4} \right),$$

the gradient confusion bound in eq. (3) holds. For both the square-loss and the logistic-loss functions, the value of $\zeta_0 \leq 2\sqrt{\beta}$ (from Lemma D.1).

Proof. We show two key properties, namely bounded gradient and non negative expectation. We will then use both these properties to complete the proof.

Bounded gradient. For every $i \in [n]$ define $\zeta_{x_i}(W) := f'(W)$. For every $p \in [\beta]$ define $H_p$ as follows.

$$H_p(x) := \sigma(W_p \cdot \sigma(W_{p-1} \cdot \sigma(\ldots \sigma(W_0 \cdot x) \ldots)).$$

Fix an $i \in [N]$. Then we have the following recurrence

$$g_{\beta}(x_i) := \sigma'(H_{\beta}(x_i))$$
$$g_p(x_i) := (W_{p+1}^\top \cdot g_{p+1}(x_i)) \cdot \text{Diag}(\sigma'(H_p(x_i))) \quad \forall p \in \{0, 1, \ldots, \beta - 1\}.$$

Then the gradients can be written in terms of the above quantities as follows.

$$\nabla_W f_i(W) = g_p(x_i) \cdot H_{p-1}(x_i)^\top \quad \forall p \in [\beta].$$

We can write $h_W(x_i, x_j)$ as follows.

$$\zeta_{x_i}(W)\zeta_{x_j}(W) \left( \sum_{p \in [\beta]} \text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i)^\top \cdot g_p(x_j) \cdot H_{p-1}(x_i)^\top] \right).$$

We will now bound $\|\nabla_{(x_i, x_j)} h_W(x_i, x_j)\|_2$. Consider $\nabla_{x_i} h_W(x_i, x_j)$. This can be written as follows.

$$\left( \nabla_{x_i} \zeta_{x_i}(W) \right) \zeta_{x_j}(W) \sum_{p \in [\beta]} \text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i)^\top \cdot g_p(x_j) \cdot H_{p-1}(x_i)^\top] +$$
$$\zeta_{x_i}(W) \zeta_{x_j}(W) \sum_{p \in [\beta]} \left[ \nabla_{x_i} \left( H_{p-1}(x_i) \cdot g_p(x_i)^\top \cdot g_p(x_j) \cdot H_{p-1}(x_i) \right) \right]^\top.$$
Observe that each of the entries in the diagonal matrix $\text{Diag}(\sigma'(H_p(x_i)))$ is at most 1. Thus, we have that $\|\text{Diag}(\sigma'(H_p(x_i)))\| \leq 1$.

We have the following relationship.

\[
\|g_p(x_i)\| \leq 1
\]
\[
\|g_p(x_i)\| \leq \|W_{p+1}^T\|\|g_{p+1}(x_i)\|\|\text{Diag}(\sigma'(H_p(x_i)))\| \leq 1 \quad \forall p \in \{0, 1, \ldots, \beta - 1\}.
\]

Moreover we have,

\[
\|\text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i) \cdot H_{p-1}(x_i)^T]\| \leq \|H_{p-1}(x_i)\|\|g_p(x_i)\|^2\|H_{p-1}(x_i)\| \leq 1.
\]

Consider $\|\nabla x_i (H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i))\|$ for every $p \in [\beta]_0$.

This can be upper-bounded by,

\[
\|\nabla x_i H_{p-1}(x_i)\|\|g_p(x_i)\|^2\|H_{p-1}(x_i)\| + \|H_{p-1}(x_i)\|\|\nabla x_i g_p(x_i)\|^2\|g_p(x_j)\|^2\|H_{p-1}(x_i)\|.
\]

Note that $\nabla x_i H_{p-1}(x_i) = g_1(x_i) \cdot \text{Diag}(\sigma'(W_0 \cdot x_i)) \cdot W_0^T \cdot g_p(x_i)^T$. Thus, $\|\nabla x_i H_{p-1}(x_i)\| \leq 1$.

We will now show that $\|\nabla x_i g_p(x_i)\| \leq \beta - p + 1$. We prove this inductively. Consider the base case when $p = \beta$.

\[
\|\nabla x_i g_\beta(x_i)\| = \|\nabla x_i \sigma'(H_\beta(x_i))\| \leq 1 = \beta - \beta + 1.
\]

Now, the inductive step.

\[
\|\nabla x_i g_p(x_i)\| \leq \|\nabla x_i g_{p+1}(x_i)\| + \|\nabla x_i \text{Diag}(\sigma'(H_p(x_i)))\| \leq \beta - p \leq \beta - p + 1.
\]

Thus, using eq. (10) and the above arguments, we obtain, $\|\nabla (x_i, x_j) h_W(x_i, x_j)\|_2 \leq \zeta_0^2(\beta + 1) + \zeta_0^2(\beta + 1)(\beta + 2) \leq 2\zeta_0^2(\beta + 2)^2$ and thus, $\|\nabla (x_i, x_j) h_W(x_i, x_j)\|_2 \leq 4\zeta_0^2(\beta + 2)^2$.

**Non-negative expectation.**

\[
\mathbb{E}_{x_i, x_j}[h(x_i, x_j)] = \mathbb{E}_{x_i, x_j}[\langle \nabla f_i(W), \nabla f_j(W) \rangle]
\]
\[
= \langle \mathbb{E}_{x_i}[\nabla f_i(W)], \mathbb{E}_{x_j}[\nabla f_j(W)] \rangle
\]
\[
= \|\mathbb{E}_{x_i}[\nabla f_i(W)]\|^2 \geq 0.
\]

We have used the fact that $\nabla f_i(W)$ and $\nabla f_j(W)$ are identically distributed and independent.

**Concentration of Measure.** We combine the two properties as follows. From **Non-negative Expectation** property and Eq. (25), we have that

\[
\text{Pr}[h_W(x_i, x_j) \leq -\eta] \leq \text{Pr}[h_W(x_i, x_j) \leq \mathbb{E}_{(x_i, x_j)}[h_W(x_i, x_j)] - \eta] \leq \exp\left(\frac{-cd\eta^2}{16\zeta_0^4(\beta + 2)^4}\right).
\]

(11)

To obtain the probability that some value of $h_w(\nabla_w f_i, \nabla_w f_j)$ lies below $-\eta$, we use a union bound. There are $N(N-1)/2 < N^2/2$ possible pairs of data points to consider, and so this probability is bounded above by $N^2 \exp\left(\frac{-cd\eta^2}{16\zeta_0^4(\beta + 2)^4}\right)$. \qed
D.3.1 Proof of Corollary 4.1

Before we prove Corollary 4.1 we prove the following helper lemma.

Lemma D.2. Suppose \( \max_W \| \nabla_W f_i(W) \| \leq M \), and both \( \nabla_W f_i(w) \) and \( \nabla_W f_j(W) \) are Lipschitz in \( W \) with constant \( L \). Then \( h_W(x_i, x_j) \) is Lipschitz in \( W \) with constant \( 2LM \).

Proof. We view \( W \) as flattened vector. We now prove the above result for these two vectors. For two vectors \( w, w' \),

\[
|h_w(x_i, x_j) - h_{w'}(x_i, x_j)| = |\langle \nabla_w f_i(w), \nabla_w f_j(w) \rangle - \langle \nabla_{w'} f_i(w'), \nabla_{w'} f_j(w') \rangle| \\
= |\langle \nabla_w f_i(w) - \nabla_{w'} f_i(w'), \nabla_w f_j(w) \rangle - \langle \nabla_{w'} f_i(w'), \nabla_w f_j(w) \rangle| \\
\leq |\langle \nabla_w f_i(w) - \nabla_{w'} f_i(w'), \nabla_{w'} f_j(w') \rangle - \langle \nabla_{w'} f_i(w'), \nabla_{w'} f_j(w') \rangle| \\
\leq \| \nabla_w f_i(w) - \nabla_{w'} f_i(w') \| \| \nabla_{w'} f_j(w') \| \\
\leq 2LM \| w - w' \|.
\]

Here the first inequality uses the triangle inequality, the second inequality uses the Cauchy-Schwartz inequality, and the third and fourth inequalities use the assumptions that \( \nabla_w f_i(w) \) and \( \nabla_w f_j(w) \) are Lipschitz in \( w \) and have bounded norm. 

We are now ready to prove the corollary, which we restate here. The proof uses a standard "epsilon-net" argument; we identify a fine net of points within the ball \( B_r \). If the gradient confusion is small at every point in this discrete set, and the gradient confusion varies slowly enough with \( W \), when we can guarantee small gradient confusion at every point in \( B_r \).

Corollary 4.1 (Uniform concentration around the minimizer). Select a point \( W = (W_0, W_1, \ldots, W_\beta) \), satisfying Assumption 4.1. Consider a ball \( B_r \) centered at \( W \) of radius \( r > 0 \). If the data \( \{x_i\}_{i \in [N]} \) are sampled uniformly from a unit sphere, then the gradient confusion bound in eq. (3) holds uniformly at all points \( W' \in B_r \) with probability at least

\[
1 - N^2 \exp \left( -\frac{cd\eta^2}{64\zeta_0^2(\beta + 2)^4} \right), \quad \text{if } r \leq \eta/4\zeta_0^2,
\]

\[
1 - N^2 \exp \left( -\frac{cd\eta^2}{64\zeta_0^2(\beta + 2)^4} + \frac{8d\zeta_0^2r}{\eta} \right), \quad \text{otherwise}.
\]

For both the square-loss and the logistic-loss functions, the value of \( \zeta_0 \leq 2\sqrt{\beta} \) (from Lemma D.1).

Proof. Define the function \( h^+(W) = \max_{ij} h_W(x_i, x_j) \). Our goal is to find conditions under which \( h^+(W) > -\eta \) for all \( W \) in a large set. To derive such conditions, we will need a Lipschitz
constant for \( h^+(W) \), which is no larger than the maximal Lipschitz constant of \( h_W(x_i, x_j) \) for all \( i, j \). We have that \( \|\nabla_W f_i\| = \|\nabla_W (W) x_i\| \leq \zeta_0 \). Now we need to get a \( W \)-Lipschitz constants for \( \nabla_{x_i} f_i = \zeta(x)(W) x_i \). By Lemma D.1, we have \( \|\nabla_W (\zeta(x)(W)) x_i\| = \|\nabla_W W (\zeta(x)) x_i\| \leq \zeta_0 \). Using Lemma D.2, we see that \( 2\zeta_0^2 \) is a Lipschitz constant for \( h_W(x_i, x_j) \), and thus also \( h^+(W) \).

Now, consider a minimizer \( W \) of the objective, and a ball \( B_r \) around this point of radius \( r \). Define the constant \( \epsilon = \frac{\eta}{4\zeta_0^2} \), and create an \( \epsilon \)-net of points \( N_\epsilon = \{W_i\} \) inside the ball. This net is sufficiently dense that any \( W' \in B_r \) is at most \( \epsilon \) units away from some \( W_i \in N_\epsilon \). Furthermore, because \( h^+(W) \) is Lipschitz in \( W \), \( |h^+(W') - h^+(W_i)| \leq 2\zeta_0^2\epsilon = \eta/2 \).

We now know the following: if we can guarantee that

\[
h^+(W_i) \geq -\eta/2, \text{ for all } W_i \in N_\epsilon, \tag{12}
\]

then we also know that \( h^+(W') \geq -\eta \) for all \( W' \in B_r \). For this reason, we prove the result by bounding the probability that \( (12) \) holds. It is known that \( N_\epsilon \) can be constructed so that \( |N_\epsilon| \leq (2r/\epsilon + 1)^d = (8\zeta_0^2r/\eta + 1)^d \) (see [Ver18], Corollary 4.1.13). Theorem 4.1 provides a bound on the probability that each individual point in the net satisfies condition \( (12) \). Using a union bound, we see that all points in the net satisfy this condition with probability at least

\[
1 - N^2 \left( \frac{8\zeta_0^2r}{\eta} + 1 \right)^d \exp \left( -\frac{cd(\eta/2)^2}{16\zeta_0^2} \right) \tag{13}
\]

\[
= 1 - N^2 \exp(d \log(8\zeta_0^2r/\eta + 1)) \exp \left( -\frac{cd\eta^2}{64\zeta_0^2} \right) \tag{14}
\]

\[
\geq 1 - N^2 \exp(8d\zeta_0^2r/\eta) \exp \left( -\frac{cd\eta^2}{64\zeta_0^2} \right) \tag{15}
\]

\[
= 1 - N^2 \exp \left( -\frac{cd\eta^2}{64\zeta_0^2} + \frac{8d\zeta_0^2r}{\eta} \right) \tag{16}
\]

Finally, note that, if \( r < \epsilon \), then we can form a net with \( |N_\epsilon| = 1 \). In this case, the probability of satisfying \( (12) \) is at least

\[
1 - N^2 \exp \left( -\frac{cd(\eta/2)^2}{64\zeta_0^2} \right). \square
\]

### D.4 Proof of Theorem 4.2

**Theorem 4.2** (Neural nets with randomly chosen weights). Let \( W_0, W_1, \ldots, W_\beta \) be weight matrices such that every entry in each of these matrices is an i.i.d. sample chosen uniformly from the interval, \([-1/\sqrt{\ell}, 1/\sqrt{\ell}]\), where \( \ell \) is the maximum layer width. Then, for any dataset \( x_1, x_2, \ldots, x_N \) with \( \|x_i\| \leq 1 \) for every \( i \in [N] \), we have that the gradient confusion bound in eq. (3) holds with probability at least

\[
1 - N^2 \exp \left( \frac{-cd(\eta - 4)^2}{64\zeta_0^4(\beta + 2)^4} \right).
\]

For both the square-loss and the logistic-loss functions, the value of \( \zeta_0 \leq 2\sqrt{\beta} \) (from Lemma D.1).
The proof strategy is similar to that of Theorem 4.1. We will first show that the gradient of the function $h(.,.)$ as defined in eq. (9) with respect to the weights is bounded. We obtain the following analogue of eq. (10).

$$
(\nabla W \zeta_x_i(W)) \zeta_x_j(W) \left( \sum_{p \in [\beta]_0} \text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)^T] \right) + 
$$

$$
(\nabla W \zeta_x_j(W)) \zeta_x_i(W) \left( \sum_{p \in [\beta]_0} \text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)^T] \right) + 
$$

$$
\zeta_x_i(W) \zeta_x_j(W) \sum_{p \in [\beta]_0} \left[ \nabla W (H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)) \right]^T. \hspace{1cm} (17)
$$

As in the case of the proof for Theorem 4.1, we will upper-bound the $\ell_2$-norm of the above expression. In particular, we show the following.

$$
\left\| (\nabla W \zeta_x_i(W)) \zeta_x_j(W) \left( \sum_{p \in [\beta]_0} \text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)^T] \right) \right\|_2 \leq 2\zeta_0^2(\beta + 2)^2. \hspace{1cm} (18)
$$

$$
\left\| (\nabla W \zeta_x_j(W)) \zeta_x_i(W) \left( \sum_{p \in [\beta]_0} \text{Tr}[H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)^T] \right) \right\|_2 \leq 2\zeta_0^2(\beta + 2)^2. \hspace{1cm} (19)
$$

$$
\left\| \zeta_x_i(W) \zeta_x_j(W) \sum_{p \in [\beta]_0} \left[ \nabla W (H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)) \right]^T \right\|_2 \leq 4\zeta_0^2(\beta + 2)^2. \hspace{1cm} (20)
$$

Equations (18) and (19) follow from the the fact that $\| (\nabla W \zeta_x_i(W)) \|_2 \leq \zeta_0$ and the arguments in the proof for Theorem 4.1. We will now show the proof sketch for eq. (20). For every $p \in [\beta]_0$, consider $\| \nabla W (H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)) \|$. Using the symmetry between $x_i$ and $x_j$, the expression can be upper-bounded by,

$$
2\| \nabla W H_{p-1}(x_i) \| \| g_p(x_i) \| \| g_p(x_j) \| \| H_{p-1}(x_i) \| + 2\| H_{p-1}(x_i) \| \| \nabla W g_p(x_i) \| \| g_p(x_j) \| \| H_{p-1}(x_i) \|.
$$

As before we can use an inductive argument to find the upper-bound and thus, we obtain the following which implies eq. (20).

$$
\| \nabla W (H_{p-1}(x_i) \cdot g_p(x_i)^T \cdot g_p(x_j) \cdot H_{p-1}(x_i)) \| \leq 4(\beta + 2)^2.
$$

Next, we show that the expected value can be lower-bounded by $-4$ as in the case of linear models above (Theorem 4.2). Combining these two gives us the desired result. Consider $\mathbb{E}_W[h_W(x_i, x_j)]$. We compute this expectation iteratively as follows.

$$
\mathbb{E}_W[h_W(x_i, x_j)]
$$
We will briefly describe some technical lemmas we require in our analysis. The following Chernoff-

We now prove the following inequality.

\[
= \mathbb{E}_{W_0}[\mathbb{E}_{W_1}[\ldots \mathbb{E}_{W_\beta}[h_W(x_i, x_j)]]] \\
\geq -\zeta_0^2 \mathbb{E}_{W_0} \left[ \mathbb{E}_{W_1} \left[ \ldots \mathbb{E}_{W_\beta} \left[ \sum_{p \in [\beta]_0} \text{Tr}(H_{p-1}(x_i) \cdot g_p(x_i)^\top \cdot g_p(x_j) \cdot H_{p-1}(x_i)^\top) \right] \right] \right] .
\]

We now prove the following inequality.

\[
\mathbb{E}_{W_0} \left[ \mathbb{E}_{W_1} \left[ \ldots \mathbb{E}_{W_\beta-1} \left[ \text{Tr}(H_{\beta-1}(x_i) \cdot H_{\beta-1}(x_i)^\top) \right] \right] \right] \\
+ \mathbb{E}_{W_0} \left[ \mathbb{E}_{W_1} \left[ \ldots \mathbb{E}_{W_\beta} \left[ \sum_{p \in [\beta]_0 \setminus \{\beta\}} \text{Tr}(H_{p-1}(x_i) \cdot g_p(x_i)^\top \cdot g_p(x_j) \cdot H_{p-1}(x_i)^\top) \right] \right] \right] .
\]

The first sum in the above expression can be upper-bounded by 1, since \(|\sigma(x)| \leq 1\). We will now show that the second sum is 0. Consider the inner-most expectation. The weights \(W_\beta\) appears only in the expression \(g_p(x_j)^\top \cdot g_p(x_j)\). Moreover, note that every entry in \(W_\beta\) is an i.i.d. normal random variable with mean 0. Thus, the second summand simplifies to,

\[
\mathbb{E}_{W_0} \left[ \mathbb{E}_{W_1} \left[ \ldots \mathbb{E}_{W_{\beta-1}} \left[ \sum_{p \in [\beta]_0 \setminus \{\beta, \beta-1\}} \text{Tr}(H_{p-1}(x_i) \cdot g_p(x_i)^\top \cdot g_p(x_j) \cdot H_{p-1}(x_i)^\top) \right] \right] \right] .
\]

Applying the above argument repeatedly we obtain that the second summand (eq. 23) is 0.

Thus, we obtain the inequality in eq. (21) which implies that \(\mathbb{E}[h_W(x_i, x_j)] \geq -\zeta_0^2\). Using the fact that \(\zeta_0 = 2\) we obtain the statement.

\section{Technical lemmas}

We will briefly describe some technical lemmas we require in our analysis. The following Chernoff-

\textbf{Lemma E.1} \textit{(Concentration of Lipshitz function over a sphere). Let} \(x \in \mathbb{R}^d\) \textit{be sampled uniformly from the surface of a} \(d\)-\textit{dimensional sphere. Consider a Lipshitz function} \(\ell : \mathbb{R}^d \to \mathbb{R}\) \textit{which is differentiable everywhere. Let} \(|\nabla \ell|_2\) \textit{denote} \(\sup_{x \in \mathbb{R}^d} |\nabla \ell(x)|_2\). \textit{Then for any} \(t \geq 0\) \textit{and some fixed constant} \(c \geq 0\), \textit{we have the following.}

\[
\text{Pr} \left[ |\ell(x) - \mathbb{E}[\ell(x)]| \geq t \right] \leq 2 \exp \left( -\frac{cdt^2}{\rho^2} \right) ,
\]

\textit{where} \(\rho \geq |\nabla \ell|_2\).
We will rely on the following generalization of Lemma E.1. We would like to point out that the underlying metric is the Euclidean metric and thus we use the $\| \cdot \|_2$-norm.

**Corollary E.1.** Let $x, y \in \mathbb{R}^d$ be two mutually independent vectors sampled uniformly from the surface of a $d$-dimensional sphere. Consider a Lipschitz function $\ell : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ which is differentiable everywhere. Let $\| \nabla \ell \|_2$ denote $\sup_{(x,y)\in\mathbb{R}^d \times \mathbb{R}^d} \| \nabla \ell(x,y) \|_2$. Then for any $t \geq 0$ and some fixed constant $c \geq 0$, we have the following.

\[
\Pr \left[ \left| \ell(x,y) - \mathbb{E}[\ell(x,y)] \right| \geq t \right] \leq 2 \exp \left( -\frac{c dt^2}{\rho^2} \right), \tag{25}
\]

where $\rho \geq \| \nabla \ell \|_2$.

**Proof.** This corollary can be derived from Lemma E.1 as follows. Note that for every fixed $\tilde{y} \in \mathbb{R}^d$, eq. (24) holds. Additionally, we have that the vectors $x$ and $y$ are mutually independent. Hence we can write the LHS of eq. (25) as the following.

\[
\int_{(\tilde{y})_1=-\infty}^{(\tilde{y})_1=\infty} \cdots \int_{(\tilde{y})_d=-\infty}^{(\tilde{y})_d=\infty} \Pr \left[ \left| \ell(x,y) - \mathbb{E}[\ell(x,y)] \right| \geq t \mid y = \tilde{y} \right] \phi(\tilde{y}) d(\tilde{y})_1 \cdots d(\tilde{y})_d.
\]

Here $\phi(\tilde{y})$ refers to the pdf of the distribution of $y$. From independence, the inner term in the integral evaluates to $\Pr \left[ \left| \ell(x,\tilde{y}) - \mathbb{E}[\ell(x,\tilde{y})] \right| \geq t \right]$. We know this is less than or equal to $2 \exp \left( -\frac{c dt^2}{\| \nabla \ell \|_2^2} \right)$. Therefore, the integral can be upper bounded by the following.

\[
\int_{(\tilde{y})_1=-\infty}^{(\tilde{y})_1=\infty} \cdots \int_{(\tilde{y})_d=-\infty}^{(\tilde{y})_d=\infty} 2 \exp \left( -\frac{c dt^2}{\| \nabla \ell \|_2^2} \right) \phi(\tilde{y}) d(\tilde{y})_1 \cdots d(\tilde{y})_d.
\]

Since $\phi(\tilde{y})$ is a valid pdf, we get the required eq. (25). \qed

Additionally, we will use the following facts about a normalized Gaussian random variable.

**Lemma E.2.** For a normalized Gaussian $x$ (i.e., an $x$ sampled uniformly from the surface of a unit $d$-dimensional sphere) the following statements are true.

1. $\forall p \in [d] \text{ we have that } \mathbb{E}[(x)_p] = 0$.
2. $\forall p \in [d] \text{ we have that } \mathbb{E}[(x)_p^2] = 1/d$.

**Proof.** Part (1) can be proved by observing that the normalized Gaussian random variable is spherically symmetric about the origin. In other words, for every $p \in [d]$ the vectors $(x_1, x_2, \ldots, x_p, \ldots, x_d)$ and $(x_1, x_2, \ldots, -x_p, \ldots, x_d)$ are identically distributed. Hence $\mathbb{E}[x_p] = \mathbb{E}[-x_p]$ which implies that $\mathbb{E}[x_p] = 0$.

Part (2) can be proved by observing that for any $p, p' \in [d]$, $x_p$ and $x_{p'}$ are identically distributed. Fix any $p \in [d]$. We have that $\sum_{p' \in [d]} \mathbb{E}[x_{p'}^2] = d \times \mathbb{E}[x_p^2]$. Note that we have
\[
\sum_{p' \in [d]} \mathbb{E}[x_{p'}^2] = \int_{(x)_1 = -\infty}^{(x)_1 = \infty} \cdots \int_{(x)_d = -\infty}^{(x)_d = \infty} \sum_{p'' \in [d]} x_{p''}^2 \phi(x) d(x)_1 \cdots d(x)_d = 1.
\]

Therefore \( \mathbb{E}[x_p^2] = 1/d \).

We use the following well-known Gaussian concentration inequality in our proofs (e.g., Chapter 5 in [BLM13]).

**Lemma E.3** (Gaussian Concentration). Let \( x = (x_1, x_2, \ldots, x_d) \) be i.i.d. \( N(0, \nu^2) \) random variables. Consider a Lipshitz function \( \ell : \mathbb{R}^d \to \mathbb{R} \) which is differentiable everywhere. Let \( \|\nabla \ell\|_2 \) denote \( \sup_{x \in \mathbb{R}^d} \|\nabla \ell(x)\|_2 \). Then for any \( t \geq 0 \), we have the following.

\[
\Pr \left[ \|\ell(x) - \mathbb{E}[\ell(x)]\| \geq t \right] \leq 2 \exp \left( -\frac{t^2}{2\nu^2\rho^2} \right),
\]

where \( \rho \geq \|\nabla \ell\|_2 \).

## F Additional discussion of the small weights assumption

Without the small-weights assumption (Assumption 4.1), the signal propagated forward or the gradients \( \nabla W f_i \) could potentially blow up in magnitude, making the network untrainable. Proving non-vacuous bounds in case of such blow-ups in magnitude of the signal or the gradient is not possible in general, and thus, we assume this restricted class of weights.

Note that the small-weights assumption is not just a theoretical concern, but also usually enforced in practice. Neural networks are often trained with weight decay regularizers of the form \( \sum_i \|W_i\|_2^\gamma \), which keep the weights small during optimization. The operator norm of convolutional layers have also recently been used as an effective regularizer as well for image classification tasks [SGL18].

**Assumption 4.1 holds for random initialization.** We can show formally that Assumption 4.1 holds with high probability for random initializations that are frequently used in practice [Ben12]. Consider a random weight matrix \( W \in \mathbb{R}^{\ell_1 \times \ell_2} \) where every entry is an independent sample from \( \mathcal{N}(0, \frac{1}{2\ell}) \). Then we can show that the operator norm of \( W \) is upper-bounded by 1 with high probability. In particular we have the following, the proof for which follows directly from Theorem 2.3.8 and Proposition 2.3.10 in [Tao12] with appropriate scaling. Define \( \ell := \max\{\ell_1, \ell_2\} \). Then with probability at least \( 1 - \Omega(\exp(-\ell^2)) \) we have that \( \|W\| \leq 1 \).

This argument can also be extended to convolutional neural nets. Consider a 2D input \( X \in \mathbb{R}^{\ell \times \ell} \) and a filter \( F \in \mathbb{R}^{k \times k} \). The convolution operation \( X * F = Hx \) where \( H \in \mathbb{R}^{(\ell-k+1)^2 \times \ell^2} \) is a Toeplitz matrix (assuming a stride of 1 and no padding) and \( x \in \mathbb{R}^{\ell^2 \times 1} \) is the flattened input vector [Gra06]. Define \( d_{\min} := \min\{(\ell - k + 1)^2, \ell^2\} \). Consider a randomly initialized filter where every entry of the filter is a \( \mathcal{N}(0, \frac{1}{2d_{\min}}) \) random matrix. It then holds that with probability at least
$1 - \Omega(\exp(-d_{\text{min}}^2))$, we have $\|H\| \leq 1$. This again follows from Theorem 2.3.8 and Proposition 2.3.10 in [Tao12] and the fact that each column in $H$ has at most $d_{\text{min}}$ non-zero entries each of which is an independent gaussian random variable.

While, in general, there is no reason to believe that such a small-weights assumption would continue to hold during optimization without explicit regularizers like weight decay, some recent work has shown evidence that the weights do not move too far away during training from the random initialization point for overparameterized neural nets [NLB+18, DR17, NK19, ZCG18, AZLS18, DZPS18, OS18]. It is worth noting though that all these results have been shown under some restrictive assumptions, such as very wide networks.