An Empirical Investigation of Training Speed and Generalisation

Candidate number: 1048593

University of Oxford

Submitted in partial completion of the

MSc in Computer Science

Trinity Term 2021

Contents

Abstract

In recent works, neural networks' training speed has been found to have an intriguing connection with their generalisation performance [Lyle et al., 2020, Ru et al., 2020, Jiang et al., 2020]. However, the mechanism causing this connection is not entirely clear for deep neural networks. In this dissertation, we came up with the robust flatness hypothesis and the gradient transfer hypothesis, either of which can potentially explain this connection. Additionally, we conducted a large number of neural network training experiments, aiming to find out the conditions under which the hypotheses might hold.

The experimental results confirmed that the connection between training speed and generalisation is robust: it can be observed with two popular neural architecture choices (SkipInit and ResNet), two optimiser choices (Stochastic Gradient Descent and Stochastic Gradient Langevin Dynamics), and with or without data augmentation by logit averaging. Further, we concluded that the robust flatness hypothesis might explain the connection except when Stochastic Gradient Langevin Dynamics is used. The gradient transfer hypothesis might explain the connection except when Stochastic Gradient Langevin Dynamics or the ResNet architecture is used. The conclusions reached are a step towards explaining the generalisation of neural networks via their training speed.

1 Introduction

Contents

1.1 Motivation

The problem of how neural networks generalise to unseen data has been pondered upon and is considered not fully understood. There have been numerous attempts at bounding the generalisation performances using classical methods from a learning theory perspective. Hardt et al. [2016] showed that when stochastic gradient descent is used as the optimisation method, parametric models have vanishing generalisation errors; Hochreiter and Schmidhuber [1997] and Keskar et al. [2017] demonstrated that the network flatness can explain generalisation from a lower function complexity perspective; McAllester [1999] drew from the probably approximately correct theory and the Bayesian theory of learning and derived a bound for generalisation; Jacot et al. [2018] used infinite-width network dynamics to explain generalisation. However, these methods suffer from serious flaws such as vacuous bounds, as pointed out by

Dziugaite and Roy [2017] and Neyshabur et al. [2017]; or paradoxical behaviours under equivalent network reparameterisation shown by Dinh et al. [2017]; or unrealistic assumptions such as the network being infinitely wide. The neural network community is still in search of plausible explanations of generalisation, with non-vacuous bounds, based on realistic assumptions.

1.2 Overview

Recently, there have been studies on using optimisation-based complexity measures to predict generalisation [Jiang et al., 2020, Lyle et al., 2020, Ru et al., 2020]. In particular, training speed as a generalisation measure has been studied for infinitely wide neural networks and linear models by Lyle et al. [2020]. Its predictive power of generalisation for deep neural networks was utilised for Neural Architecture Search by Ru et al. [2020]. However, there has been little research into the mechanism behind this predictive power for finitely wide neural networks. In this dissertation, we present an empirical study of training speed's predictive power for deep neural networks, as well as an investigation into the underlying mechanism. The central question this empirical investigation sets out to answer is whether there is a known connecting training speed and generalisation. And if yes, what are the necessary conditions for such mechanisms to apply in practice.

1.3 Outline

Chapter 2 We give a brief introduction to neural network training, generalisation, and why existing theories cannot fully explain generalisation satisfactorily while introducing notations that will be used throughout the dissertation. Then, we introduce theories looking at different aspects of neural networks that might explain the generalisation mystery. Training speed as a complexity measure and its connections with the prior theories are also covered in this chapter.

Chapter 3 We introduce the robust flatness hypothesis and the gradient transfer hypothesis, which can explain the connection between training speed and generalisation. The experimental setup for these two hypotheses is explained and the central question this dissertation tries to answer is brought up: what are the conditions under which these hypotheses might be true in the practice of neural network training.

Chapter 4 We present the experimental results. We then analyse the scope of the two hypotheses and various findings in the experiments.

Chapter 5 We conclude the dissertation by summarising the experimental insights and point out the future directions for pursuing a further understanding of training speed and generalisation.

1.4 Contributions

- We came up with two hypotheses, robust flatness and gradient transfer that might potentially explain the connection between training speed and generalisation, based on prior theories.
- We conducted experiments with 6 different groups of experiments over a wide range of hyperparameters, each isolating one individual neural network training technique used in practice, including data augmentation with transformation and logit averaging, the Stochastic Gradient Langevin Dynamics optimiser, the Multi-Layer Perceptron architecture, and Batch Normalization. The experiment groups are representative of a large regime of neural networks that solves visual tasks.
- Drawing on the experimental results, we confirmed the connection between training speed and generalisation, on a large family of finite neural networks. Further, we found the conditions under which each of our hypotheses might explain the connection.

2
Background

Contents

2.1 Neural Networks and Generalisation

Neural networks are parametric models often used to carry out standard statistical learning tasks such as regression and classification. In statistical learning, the objective is to estimate a set of parameters that can describe the data well from a finite set of training examples. For convenience, we lump the network weights into one single vector $\theta = [\theta_1^T, \theta_2^T, \ldots, \theta_L^T]^T$, where $\theta_1, \theta_2, \ldots, \theta_L$ are network weights in each of its L layers. The network implements a function $f_\theta : X \mapsto Y$.

We denote the finite training set by $D_{train} = \{(\mathbf{x_i}, y_i)\}_{i=1}^{m}$, where each data pair $(\mathbf{x_i}, y_i)$ is assumed to have been generated from an underlying distribution $(\mathbf{x_i}, y_i) \sim \mathcal{D}_{X,Y}$. The risk associated with the network function f_θ on a single datapoint $(\mathbf{x}, y)$ measures the dissimilarity between our prediction $f_\theta(\mathbf{x})$ and the ground truth y. It is denoted by $R\left(f_\theta\left(\mathbf{x}\right), y\right)$, where R measures the risk on a single pair of prediction and target. R is also called the loss function.

Definition 2.1 (Empirical Risk and Population Risk)**.** The empirical risk is the average risk on the training dataset, defined as

$$R_{emp}\left(\theta, D_{train}\right) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} R\left(f_\theta\left(\mathbf{x_i}\right), y_i\right). \tag{2.1.0.1}$$

And the population risk is the expected risk over the underlying data generating distribution, defined as

$$R_{pop}\left(\theta, \mathcal{D}_{X,Y}\right) \stackrel{\text{def}}{=} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{X,Y}}[R\left(f_\theta\left(\mathbf{x}\right), y\right)]. \tag{2.1.0.2}$$

Definition 2.2 (Empirical Risk Minimisation and Generalisation Error)**.** Given a fixed training set and underlying distribution, we wish to find a set of weights θ^* that minimises the population risk. However, in practice, we often do not know the underlying data generating distribution $\mathcal{D}_{X,Y}$ and use the empirical risk as a surrogate to minimise. This is called Empirical Risk Minimisation (ERM). The objective of ERM is to find a set of weights that minimises the empirical risk:

$$\hat{\theta} = \arg\min_{\theta} R_{emp}\left(\theta, D_{train}\right). \tag{2.1.0.3}$$

The generalisation error measures the difference between the population risk and the empirical risk, for the set of weights found:

$$\text{gen}\left(\hat{\theta}, D_{train}, \mathcal{D}_{X,Y}\right) \stackrel{\text{def}}{=} R_{pop}\left(\hat{\theta}, \mathcal{D}_{X,Y}\right) - R_{emp}\left(\hat{\theta}, D_{train}\right). \tag{2.1.0.4}$$

When it is clear from the context, we will omit D_{train} and $D_{X,Y}$.

Without knowing $\mathcal{D}_{X,Y}$ but being able to sample from it, we generate a test set $D_{test} = \{\hat{\mathbf{x}}_\mathbf{i}, \hat{y}_i\}_{i=1}^{\hat{m}}$, where $(\hat{\mathbf{x}}_\mathbf{i}, \hat{y}_i) \sim \mathcal{D}_{X,Y}$, and estimate the population risk by calculating the empirical risk on the test set:

$$\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{X,Y}}[R\left(f_\theta\left(\mathbf{x}\right), y\right)] \approx R_{emp}\left(\hat{\theta}, D_{test}\right) = \frac{1}{\hat{m}}\sum_{i=1}^{\hat{m}} R\left(f_\theta\left(\hat{\mathbf{x}}_\mathbf{i}\right), \hat{y}_i\right). \tag{2.1.0.5}$$

Then we can estimate the generalisation error by

$$\begin{aligned}\operatorname{gen}\left(\hat{\theta}, D_{train}, \mathcal{D}_{X,Y}\right) &\approx R_{emp}\left(\hat{\theta}, D_{test}\right) - R_{emp}\left(\hat{\theta}, D_{train}\right) \\ &= \frac{1}{\hat{m}} \sum_{i=1}^{\hat{m}} R\left(f_\theta\left(\hat{\mathbf{x}}_\mathbf{i}\right), \hat{y}_i\right) - \frac{1}{m} \sum_{i=1}^{m} R\left(f_\theta\left(\mathbf{x}_\mathbf{i}\right), y_i\right).\end{aligned} \tag{2.1.0.6}$$

2.2 Existing Theories Cannot Explain Generalisation in Neural Networks

Classical theories based on the concepts of VC dimension [Vapnik, 1998], Rademacher complexity [Bartlett and Mendelson, 2002], and uniform stability [Mukherjee et al., 2006, Bousquet and Elisseeff, 2002, Poggio et al., 2004] etc. have been considered by the research community to explain generalisation. However, there are significant differences between neural networks and conventional models of learning. One major difference is that modern artificial neural networks often have a much larger effective capacity [Neyshabur et al., 2015b]. Zhang et al. [2017] showed that, as a consequence, some successful network architectures can perfectly fit the training data when their labels have been randomly permuted. They further showed that the effective capacities of several network architectures are so large that they shatter the training data: a near-zero training loss can be achieved even when the inputs are random Gaussian noises. This poses significant challenges to explaining generalisation with the classical theories of learning, as they give vacuous bounds for the generalisation error in the case of deep neural networks. For example, the randomization tests in [Zhang et al., 2017] suggest that the Rademacher complexity for the corresponding class of models is close to 1. Hence, bounds based on Rademacher complexity are not in general useful for explaining generalisation. The existing generalisation bounds are therefore of little use when it comes to explaining neural networks. Instead, some more recent theoretical results were specifically derived for the purpose of explaining neural network generalisation. They focus on properties and practices that have not been closely studied before, which we will review in detail in Section 2.3.

2.3 Theories and Empirical Studies of Neural Network Generalisation

We first review theories for neural network generalisation from the perspective of the optimisation algorithm, the network flatness, and the training dynamics. A subsection focusing on empirical studies of complexity measures for generalisation is presented afterwards.

2.3.1 Stochastic Gradient Descent

Gradient descent is a method of minimising an objective function $J(\theta)$ parameterised by model parameters θ by updating parameters in the opposite direction to the gradient $\nabla_\theta J(\theta)$. When training neural networks, we perform ERM as in Definition 2.2 and use $R_{emp}(\theta, D_{train})$ as our objective function. Partially because of the high computational cost of calculating the full batch gradient $\nabla_\theta R_{emp}(\theta, D_{train})$, in practice, Stochastic Gradient Descent is often used.

Definition 2.3 (Stochastic Gradient Method (SGM))**.** Let α_t be the learning rate at the t-th step of update. Consider weights of a network being updated with the following rule:

$$\theta = \theta - \alpha_t \cdot \nabla_\theta R_{emp}\left(\theta, \{\mathbf{x_j}, y_j\}_{j=t*s}^{(t+1)*s-1}\right), \tag{2.3.1.1}$$

where s is the batch size. When $s = 1$, this optimisation method is called Stochastic Gradient Descent (SGD) and when $s > 1$, it is referred to as mini-batch SGD.

SGD is often used in practice. The training data is randomly split into mini-batches of s training examples and the update rule becomes:

$$\theta = \theta - \eta \cdot \nabla_\theta R_{emp}\left(\theta, \{\mathbf{x_j}, y_j\}_{j=i}^{i+s-1}\right), \tag{2.3.1.2}$$

where $\{\mathbf{x_j}, y_j\}_{j=i}^{i+s-1}$ is one mini-batch. We should note that our notation of SGD might be called mini-batch gradient descent in other texts and should be differentiated from the notion of SGD where n is limited to be 1. We will now discuss works that establish connections between SGD and generalisation.

Uniform stability bounds Hardt et al. [2016] showed that parametric models have vanishing generalisation errors when they are trained by SGD. Their analysis is based on the algorithmic stability [Bousquet and Elisseeff, 2002] of the SGD.

Definition 2.4 (Uniform Stability, Definition 2.1 in [Hardt et al., 2016])**.** A randomised algorithm $A : (X, Y)^n \to \boldsymbol{\Theta}$ can learn a set of weights from a given training set. We say that A is ϵ-uniformly stable if for all datasets $D_1, D_2 \sim (X, Y)^n$ such that D_1 and D_2 differ by at most one example, we have

$$\sup_{(\boldsymbol{x},y)} \mathbb{E}_A \left[R\left(f_{A(D_1)}\left(\boldsymbol{x}\right), y \right) - R\left(f_{A(D_2)}\left(\boldsymbol{x}\right), y \right) \right] \leq \epsilon, \tag{2.3.1.3}$$

where the expectation is taken over the randomness of A and we use ϵ_{stab} to denote the infimum over all ϵ for which Inequality 2.3.1.3 holds.

Hardt et al. [2016] upper-bounded the generalisation error of a non-convex model by the number of training iterations SGD takes.

Theorem 2.5 (Theorem 3.12 in [Hardt et al., 2016])**.** Assume that $L\left(f_{(\cdot)}\left(\boldsymbol{x}\right), y \right) \in [0, 1]$ is an L-Lipschitz and β-smooth loss function for every $(\boldsymbol{x}, y)$: Suppose that we run SGD for T steps with monotonically non-increasing step sizes $\alpha_t \leq c/t$. Then, SGD has uniform stability with

$$\epsilon_{stab} \leq \frac{1 + 1/\beta c}{n - 1} \left(2cL^2\right)^{\frac{1}{\beta c + 1}} T^{\frac{\beta c}{\beta c + 1}} \tag{2.3.1.4}$$

With this result, it can be further shown that the expectation of the generalisation error is upper-bounded by the same quantity. Therefore, from this notion of algorithmic stability, we have a theory of why neural networks trained with SGD can generalise.

In modern practices, the hyper-parameters of neural networks often cause equation 2.3.1.4 to give a vacuous bound, yet they still exhibit good generalisation abilities. This suggests that uniform stability does not give a full account for the generalisation power of modern neural networks by itself.

The implicit regularisation effects of SGD It is understood that SGD, in the limit of vanishing learning rates, follows the gradient flow on the full batch loss function [Yaida, 2019]. However, in practice, slightly larger learning rates are found to have better generalisation performances [LeCun et al., 2012, Keskar et al., 2017], which is not explained by generalisation bounds. Smith et al. [2021] showed that for SGD with random shuffling, the mean SGD iterate stays close to the gradient flow of a modified loss function. We denote the average mini-batch gradient norms by R_{reg}:

$$R_{reg} = \frac{1}{B}\sum_{b=0}^{B-1} \|\nabla_\theta R_{emp}\left(\theta, \{(\boldsymbol{x_j}, y_j)\}_{j=b*s}^{(b+1)*s-1}\right)\|^2. \tag{2.3.1.5}$$

The modified loss penalises this quantity R_{reg}. Namely, the modified loss is

$$\begin{aligned} R_{emp}^{SGD}(\theta, D_{train}) &= R_{emp}(\theta, D_{train}) + \eta R_{reg}(\theta, D_{train}, B) \\ &= R_{emp}(\theta, D_{train}) + \eta \cdot \frac{1}{B}\sum_{b=0}^{B-1} \|\nabla_\theta R_{emp}\left(\theta, \{(\boldsymbol{x_j}, y_j)\}_{j=b*s}^{(b+1)*s-1}\right)\|^2, \end{aligned} \tag{2.3.1.6}$$

where η is the learning rate, B is the number of mini-batches in one epoch, s is the batch size, and $R_{emp}\left(\theta, \{(\boldsymbol{x_j}, y_j)\}_{j=b*s}^{(b+1)*s-1}\right)$ is the empirical risk on the b-th mini-batch.

By proving that the mean SGD iterate is close to the gradient flow of a loss function that penalises the gradient norm, the implicit regularisation effects of SGD can potentially be linked to the flatness of minima. They also conducted experiments where a small learning rate and explicitly regularisation of the gradient norm were used to reproduce the generalisation performance with a larger learning rate.

2.3.2 Network Flatness

It was argued by both Hochreiter and Schmidhuber [1997] and Keskar et al. [2017] that generalisation might be explained by the flatness of the minima found by the optimisation algorithm, although Hochreiter and Schmidhuber [1997] and Keskar et al. [2017] used their own distinct notions of flatness. Dinh et al. [2017] refuted the idea that flatness in either definition can explain generalisation by itself, by introducing a reparametrisation on deep rectified networks that make equivalent

minima to change their sharpness arbitrarily. However, there are still interesting aspects of flatness that bear the hope of explaining generalisation.

Volume ϵ-sharpness In [Hochreiter and Schmidhuber, 1997] a flat minimum was defined as "a large connected region in weight space where the error remains approximately constant." They argued for explaining generalisation with network flatness from two perspectives: (1) It requires fewer bits of information to describe a flatter minimum (see Section 4.6 of [Hochreiter and Schmidhuber, 1997] for a justification). A smaller Minimum Description Length (MDL) for a set of weights indicates that the network function has a smaller Kolmogorov complexity [Li and Vitányi, 2019]. And a lower network function complexity can in turn attribute to a high generalisation performance [Rissanen, 1983]. (2) In a standard Bayesian view, a minimum with a larger posterior mass contributes more to the inference. Section 6.2 of [Buntine and Weigend, 1991] further suggests that minima with large posterior masses (fat minima) are also flat minima. Both the MDL and the Bayesian principles incentive us to prefer flatter minima.

Dinh et al. [2017] interpreted this flat minimum in the format of a complexity measure called volume ϵ-sharpness: Given $\epsilon > 0$, a minimum θ, and a loss R_{emp}, we can define $C\left(R_{emp}, \theta, \epsilon\right)$ as the largest connected set such that $\forall \boldsymbol{\theta'} \in C\left(R_{emp}, \theta, \epsilon\right), R_{emp}(\boldsymbol{\theta'}) < R_{emp}(\theta) + \epsilon$. The volume ϵ-sharpness is the volume of $C\left(R_{emp}, \theta, \epsilon\right)$.

Dinh et al. [2017] subsequently showed that for every minimum of deep rectified networks, there is a connected region of infinitely large volume, in which the loss remains approximately constant. Hence the volume ϵ-sharpness cannot explain generalisation of deep rectified networks by itself. However, there is some subtle difference in the definition of volume ϵ-sharpness by Dinh et al. [2017] and the original definition. Instead of using an unrestricted connected region around the minimum, Hochreiter and Schmidhuber [1997] considered a **box**, defined as "an L-dimensional hypercuboid" with its centre at the minimum. This restriction potentially makes volume ϵ-sharpness more robust to the reparametrisation by Dinh et al. [2017].

ϵ-sharpness Keskar et al. [2017] suggested that the drop in performance for large-batch-size optimisation could be attributed to that they tend to converge to sharp minima. Through experiments with different architectures, they found that larger batch sizes are correlated with sharper minima and worse generalisation. The flatness metric they used was called the ϵ-sharpness in [Dinh et al., 2017]: Let $B_2(\epsilon, \theta)$ be an Euclidean ball in the weight space of radius ϵ centered at a minimum θ. The ϵ-sharpness is then defined as proportional to

$$\frac{\max_{\theta' \in B_2(\epsilon,\theta)} \left(R_{emp}(\theta') - R_{emp}(\theta)\right)}{1 + R_{emp}(\theta')}. \tag{2.3.2.1}$$

Dinh et al. [2017] found that with the same reparametrisation scheme, every minimum in deep rectified networks is observationally equivalent to a minimum that generalises as well but with high ϵ-sharpness. Therefore, some limits of explaining generalisation with this flatness measure are also exposed. We note that these flatness measures only look at the final weights of the network, instead of the whole training trajectory.

PAC-Bayes Sharpness McAllester [1999] introduced the PAC-Bayesian framework. The PAC-Bayesian bound derived in the framework states that the expected generalisation error over the posterior weight distribution can be bounded by the Kullback–Leibler (KL) divergence between the prior and the posterior. Further, Dziugaite and Roy [2017] showed that by optimising the bound over Gaussian posteriors, non-vacuous bounds can be achieved. Neyshabur et al. [2017] demonstrated, with small-scale experiments, that isotropic Gaussian priors and posteriors make PAC-Bayesian sharpness a good measure of generalisation. To define the PAC-Bayes sharpness, we first define the 0-1 loss on a single datapoint (x, y):

$$\hat{R}(f_\theta(x), y) = \mathbb{1}\left[\arg\max_i f_\theta(x)_i = y\right]. \tag{2.3.2.2}$$

Hence, we can define the empirical 0-1 loss, or the empirical accuracy:

$$\hat{R}_{emp}(\theta, D_{train}) = \frac{1}{\mid D_{train} \mid} \sum_{i=1}^{|D_{train}|} \hat{R}(f_\theta(x_i, y_i)). \tag{2.3.2.3}$$

The PAC-Bayes sharpness can be calculated as such:

$$\hat{\sigma}^2 = \sup\left(\{\sigma^2 \mid \mathbb{E}_{\theta' \sim \mathcal{N}(\theta, \sigma^2 I)}\left[\hat{R}_{emp}(\theta', D_{train})\right] \leq 0.1\}\right),$$

$$\text{PAC-Bayes Sharpness} = \frac{1}{\hat{\sigma}^2}. \tag{2.3.2.4}$$

This form of PAC-Bayes sharpness defined by Equation 2.3.2.4 is also more computationally efficient than that proposed by Dziugaite and Roy [2017].

2.3.3 Learning Dynamics

We review three different perspectives regarding training dynamics to view generalisation: stiffness, neural tangent kernels, and the lottery ticket hypothesis.

Stiffness Fort et al. [2019] advocated to look at generalisation with a new measure: stiffness. We can define stiffness with respect to the loss function R, the function implemented by the network forward operation f_θ, the network weights θ and two datapoints $(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2)$ from some distributions D_1 and D_2 respectively. The gradients of the loss function with respect to the network's weights on the two datapoints are:

$$\boldsymbol{g_1} = \nabla_\theta R\left(f_\theta\left(\boldsymbol{x_1}\right), y_1\right), \quad \boldsymbol{g_2} = \nabla_\theta R\left(f_\theta\left(\boldsymbol{x_2}\right), y_2\right). \tag{2.3.3.1}$$

There are two meaningful quantities, the sign of the gradient product $\text{sign}\left(\boldsymbol{g_1} \cdot \boldsymbol{g_2}\right)$, and the cosine value of the angle between the two gradients $\frac{\boldsymbol{g_1} \cdot \boldsymbol{g_2}}{\|\boldsymbol{g_1} \cdot \boldsymbol{g_2}\|}$. By taking the expectation of these two quantities, we have the sign stiffness and the cosine stiffness:

$$\begin{aligned} S_{\text{sign}} &= \mathbb{E}_{(\boldsymbol{x_1}, y_1) \sim D_1, (\boldsymbol{x_2}, y_2) \sim D_2}[\text{sign}(g_1 \cdot g_2)], \\ S_{\cos} &= \mathbb{E}_{(\boldsymbol{x_1}, y_1) \sim D_1, (\boldsymbol{x_2}, y_2) \sim D_2}\left[\frac{\boldsymbol{g_1} \cdot \boldsymbol{g_2}}{\|\boldsymbol{g_1} \cdot \boldsymbol{g_2}\|}\right]. \end{aligned} \tag{2.3.3.2}$$

Stiffness is dependent on the distributions D_1 and D_2. If both D_1 and D_2 are the training distribution, the quantities are called train-train stiffness. If both are the validation distribution, they are called val-val stiffness. If one of D_1 and D_2 is the training distribution and the other one is the validation distribution,

they are called train-val stiffness. In practice, we use a Monte-Carlo estimator to estimate the stiffness values.

Gradient confusion Sankararaman et al. [2020] proposed a similar measure with stiffness which is correlated with training speed: gradient confusion. For a dataset with m datapoints $\{(\mathbf{x_i}, y_i)\}_{i=1}^{m}$, the gradient confusion is defined as:

$$\zeta = \max\left(0, -\min_{i \neq j} \left(\nabla_\theta R\left(f_\theta\left(\boldsymbol{x_i}\right), y_i\right) \cdot \nabla_\theta R\left(f_\theta\left(\boldsymbol{x_j}\right), y_j\right)\right)\right). \tag{2.3.3.3}$$

Both are quantities based on dot products of the loss function gradients, stiffness is an estimator of the average over the entire dataset, while gradient confusion is the opposite of the minimum value of the gradient dot products between any two datapoints in the dataset.
Sankararaman et al. [2020] observed that when gradient confusion is large, the gradient updates on different datapoints differ strongly, and training becomes slower as a consequence; vice versa, when gradient confusion is small, the gradient updates on different datapoints interact harmoniously, leading to faster convergence.

Gradient variance Inspired by Chaudhari and Soatto [2018] and Smith and Le [2018], the variance of the loss gradient can be used as a generalisation measure. Very similar to stiffness, it can be defined as follows:

$$\text{Gradient variance} = \mathbb{V}_{(\mathbf{x},y)\sim D_{train}}\left[\nabla_\theta R\left(f_\theta(\mathbf{x}), y\right)\right]. \tag{2.3.3.4}$$

Notice that the gradient variance measure can be taken at any stage of training. The empirical study of Jiang et al. [2020] suggested that the gradient variance at the end of the first epoch, as well as after training, can be useful generalisation measures. Jiang et al. [2020] further pointed out that the gradient variance at the end of training corresponds to a special kind of flatness of the local minimum found.

Neural Tangent Kernel Jacot et al. [2018] proved that under some assumptions, the evolution of an artificial neural network function follows the kernel gradient of a convex functional cost with respect to the Neural Tangent Kernel (NTK). Furthermore, in the infinite-width limit, the kernel is constant during training. To see this, we first recall the following observation:

Proposition 2.6 ("NTK at initialisation converges in probability to a deterministic limiting kernel." Proposition 1 in [Jacot et al., 2018])**.** Let the number of neurons in layers of a network of depth L be $n_1, \ldots, n_L$ and n_0 be the dimension of the input. Let σ be a Lipschitz nonlinearity. Define a set of covariances $\Sigma^{(L)}$ recursively by:

$$\begin{aligned} \Sigma^{(1)}(x, x') &= \frac{1}{n_0} x^T x' + \beta^2 \\ \Sigma^{(l+1)}(x, x') &= \mathbb{E}_{f \sim \mathcal{N}\left(0, \Sigma^{(l)}\right)} \left[\sigma\left(f\left(x\right)\right) \sigma\left(f\left(x'\right)\right)\right] + \beta^2, \end{aligned} \tag{2.3.3.5}$$

where β is an initialisation hyper-parameter.

Using Proposition 2.6, a key result can be proved to reveal insights about the NTK convergence for infinite-width networks:

Theorem 2.7 (Theorem 1 in [Jacot et al., 2018])**.** "For a network of depth L, the NTK $\Theta^{(L)}$ converges to a deterministic limiting kernel, when the layer widths $n_1, \ldots, n_{L-1} \to \infty$."

Let Id_{n_L} be an $n_L \times n_L$ identity matrix. Let $\dot{\sigma}$ be the derivative of the Lipschitz nonlinearity σ. We define the following quantity over the expectation of a centred Gaussian process f of covariance $\Sigma^{(L)}$:

$$\dot{\Sigma}^{(l+1)}(x, x') = \mathbb{E}_{f \sim \mathcal{N}(0, \Sigma^{(l)})} \left[\dot{\sigma}\left(f(x)\right) \dot{\sigma}\left(f(x')\right)\right]. \tag{2.3.3.6}$$

Then we define a scalar kernel $\Theta_\infty^{(L)} : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}$ recursively:

$$\begin{aligned} \Theta_\infty^{(1)}(x, x') &= \Sigma^{(1)}(x, x') \\ \Theta_\infty^{(L+1)}(x, x') &= \Theta_\infty^{(L)}(x, x') \dot{\Sigma}^{(L+1)}(x, x') + \Sigma^{(L+1)}(x, x'), \end{aligned} \tag{2.3.3.7}$$

Finally, the kernel that the NTK converges to at the infinite-width limit is

$$\Theta^{(L)} \to \Theta_\infty^{(L)} \otimes Id_{n_L}. \tag{2.3.3.8}$$

A second key result from [Jacot et al., 2018] is that "NTK stays asymptotically constant during training" with a more general definition of training.
Let $\mathcal{F} \subseteq \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ be the space of functions that can be implemented by the network and $F^{(L)} : \mathbb{R}^P \to F$ be the NN realisation function that maps network weights to the function implemented by the network. P is the total number of network weights. Consider a distribution on inputs $p^X \subseteq \mathbb{R}^{n_0}$, we first define a bilinear form of functions $\langle \cdot, \cdot \rangle$:

$$\forall f, g \in \mathcal{F}. \langle f, g \rangle_{p^X} = \mathbb{E}_{\boldsymbol{x} \sim p^X} \left[f(\boldsymbol{x})^\top g(\boldsymbol{x}) \right]. \tag{2.3.3.9}$$

The rule to update weights according to a training direction $d_t \in \mathcal{F}$ is then

$$\partial_t \theta_p(t) = \langle \partial_{\theta_p} F^{(L)}(\theta(t)), d_t \rangle_{p^X}. \tag{2.3.3.10}$$

Theorem 2.8 (Theorem 2 in [Jacot et al., 2018])**.** Assuming that the nonlinearity σ is Lipschitz twice differentiable with bounded second derivative. If the integral $\int_0^T \|d_t\|_{p^X} dt$ stays stochastically bounded, as network width goes to infinity $(n_1, \ldots, n_{L-1} \to \infty)$, we have, for $t \in [0, T]$, the time-dependent NTK converging:

$$\Theta^{(L)}(t) \to \Theta_\infty^{(L)} \otimes Id_{n_L}. \tag{2.3.3.11}$$

The neural network optimisation problem is highly non-convex, By observing the results above, we can consider the neural network training problem in the functional space instead of the network weight space. The convexity of the functional cost might help us to understand the nature of neural network generalisation.
We need to note that explanations based on neural tangent kernel theories operate in the NTK regime, which poses some restrictions to initialisation and consider training in the infinite-width limit. It is less clear how well the NTK theories generalise to outside this regime.

The Lottery Ticket Hypothesis and linear mode connectivity Techniques for pruning weights from neural networks without harming their performances [LeCun et al., 1989, Hassibi and Stork, 1992, Han et al., 2015, Li et al., 2017] can reduce the number of weights present by more than 90%. Based on this, Frankle and Carbin [2019] articulated the Lottery Ticket Hypothesis (LTH):

Definition 2.9 (The Lottery Ticket Hypothesis)**.** *"Dense, randomly-initialized, feed-forward networks contain subnetworks (winning tickets) that-when trained in isolation-reach test accuracy comparable to the original network in a similar number of iterations."*

The authors tested the LTH by running pruning algorithms on trained networks. Then the remaining weights were set to exactly where they had been initialised and subnetworks were re-trained. The authors found that the retrained subnetworks exhibited similar or even better generalisation than the full networks. The initialisation states of the remaining weights as well as the remaining subnetwork structure constitute a winning lottery ticket. Preserving both of them are essential for recovering the performance of the full network.

The authors also found that, interestingly, as pruning proceeds, the test accuracy first increases and then decreases, forming an *Occam's Hill* [Rasmussen and Ghahramani, 2000]: the full network has too much complexity that it overfits to the training data, while the overly pruned network has too little expressiveness that it cannot describe the data well. From an MDL perspective, a network that requires fewer bits of information to describe should generalise better. However, a sparse network can be hard to train from the start. Hence, the LTH provides an appealing theory for generalisation: the dense network is easy to train and contains combinatorily many lottery tickets. The optimisation algorithm finds a winning ticket that generalises well while minimising the empirical risk.

It is worth noting that on small networks, the iterative magnitude pruning (IMP) procedure can find matching (capable of being trained to the same performance of the full network) networks at non-trivial sparsities. For larger networks, there is little empirical evidence for the LTH [Liu et al., 2019, Gale et al., 2019].

Definition 2.10 (Linear Mode Connectivity)**.** Frankle et al. [2020] studied *"whether a neural network optimizes to the same, linearly connected minimum under different samples of SGD noise."*

Consider two sets of network weights θ_1 and θ_2. We can find interpolations of the network weights between them $\theta_{interp} = \alpha\theta_1 + (1-\alpha)\theta_2$, where $\alpha \in [0, 1]$. The linear interpolation instability, instability for short, is defined as $\max_\alpha R_{emp}(\theta_{interp}) - \frac{R_{emp}(\theta_1)+R_{emp}(\theta_2)}{2}$. For a network trained for k epochs, we can subject it to two different SGD noises and achieve two sets of final weights θ_1 and θ_2. If the instability between θ_1 and θ_2 is below a certain threshold, we say that the two sets of weights are linearly connected and the network is stable to SGD noise after k epochs.
With these notions, Frankle et al. [2020] showed that IMP subnetworks are only matching when they are stable to SGD noise. This distinguishes the known cases when IMP works and not works in the literature. For larger networks (e.g. ResNet-50 on ImageNet), the sparse IMP subnetworks become stable to SGD noise early in training, but not at initialisation. This generalises IMP to find subnetworks early in training, broadening the scope of the LTH.
Further, with the distinction of whether the subnetworks are stable to SGD noise, the training process is naturally divided into two stages. During these two different training regimes, the behaviours of neural network weights are qualitatively different. For example, training speed might be indicative of different properties in the two stages. This motivates our study of the neural network training procedure focusing on early and late training regimes individually.

2.3.4 Empirical Studies of Generalisation

We review some large-scale empirical studies on what measures are best for predicting generalisation.

Complexity Measures A lower complexity measure is supposed to reflect a simpler network and indicates a better generalisation performance. A lot of complexities measures based on different aspects of neural networks have been

proposed, with the aim of predicting generalisation. Jiang et al. [2020] categorised the measures according to what they are based on, in the Related Work section of their paper. We reiterate the basis of the complexities here:

- **Theoretically motivated measures:** PAC-Bayes [McAllester, 1999, Dziugaite and Roy, 2017, Neyshabur et al., 2017] based on the Probably Approximately Correct (PAC) theory and the Bayesian theory of learning; VC-dimension [Vapnik and Chervonenkis, 2015] based on the expressivenss of model families; Various norms of network weights [Neyshabur et al., 2015b, Bartlett et al., 2017, Neyshabur et al., 2017].

- **Empirically motivated measures:** Sharpness [Hochreiter and Schmidhuber, 1997, Keskar et al., 2017] based on loss surface smoothness; Fisher-Rao measure [Liang et al., 2019]; Distance between final weights and initial weights [Nagarajan and Kolter, 2019]; Path-norm [Neyshabur et al., 2015a] of network weight trajectories.

- **Optimisation-based measures:** Training speed [Hardt et al., 2016, Wilson et al., 2017, Lyle et al., 2020]; Magnitude of the gradient noise [Chaudhari and Soatto, 2018, Smith and Le, 2018].

The methodology employed by Jiang et al. [2020] and later adopted and improved by Dziugaite et al. [2020] was to carry out numerous experiments with a thorough hyper-parameter sweep while monitoring the complexity measures mentioned. By observing how the generalisation error and the complexity measure change when a hyperparameter changes, we seek a causal connection between the complexity measure and the generalisation error, independent of the hyper-parameter. Notice that there are two possible mechanisms of changes to the hyper-parameters causing changes in the generalisation error:

1. The hyper-parameter change causes the complexity measure monitored to change. And the complexity measure changes causes the generalisation error to change.

2. The hyper-parameter change causes both the complexity measure and the generalisation error to change. But the value of the complexity measure by itself does not affect generalisation.

For each complexity measure monitored, it has explanatory power of the generalisation if 1 is the true mechanism, while it does not if 2 is the true mechanism. In practice, due to the many variables of neural network training at play, it is likely that a mixture of the two mechanisms affects generalisation for most complexity measures. To eliminate the unwanted correlation between generalisation error and complexity measure from mechanism 2, Jiang et al. [2020] carried out a conditional independence test with the Inductive Causation algorithm [Verma and Pearl, 1990]. Dziugaite et al. [2020], however, took a different approach, and ensured the soundness of the causation found by requiring the complexity measure to be robust. This means that how trustworthy a causal link between the complexity measure and the generalisation error is weighted against how robust the link is in different hyper-parameter combinations.

2.4 Training Speed and Generalisation

In the search for an adequate theory, a strong correlation between training speed and generalisation was observed [Lyle et al., 2020, Ru et al., 2020]. Lyle et al. [2020] showed that for linear models, the connection between training speed and generalisation can be established via the quantity of marginal likelihood, from a Bayesian perspective. By the Neural Tangent Kernel [Jacot et al., 2018], the connection is also established for infinitely wide networks. For deep neural networks, the Sum over Training Loss (SOTL) exhibits a predictive ability of test cross entropy [Lyle et al., 2020] and subnetwork weights of better-performing subnetworks [Ru et al., 2020]. Therefore for deep neural networks, although the observation of the connection can be made, the theory behind it is far from clear. The difficulties to use the same theory for deep neural networks include (1) the training method in practice is not Bayesian (2) the marginal likelihood is not

tractable. In this dissertation, however, we set out to find the reason for this connection for deep neural networks.

Lyle et al. [2020] and Ru et al. [2020] observed that for neural networks, SOTL and its variants are correlated with the generalisation performance, both early and late in training. This was used in [Ru et al., 2020] to predict generalisation and help Neural Architecture Search (NAS) before the network is done training. Investigating the mechanism through which this metric correlates with the generalisation performance is the goal of this dissertation. We will first introduce the theoretical background for the connection between SOTL and marginal likelihood, and how it might be viewed in the theories covered above. Then we will raise some hypotheses to test in this dissertation.

2.4.1 Marginal Likelihood and Training Statistics

Consider the training dataset D_{train} and let $D_{<i} = \{(\boldsymbol{x_j}, y_j)\}_{j=1}^{i-1}$. When doing Bayesian model selection, we want to compute the posterior mass $P(M \mid D_{train})$ for each model M. The marginal likelihood $P(D_{train} \mid M)$ is used as the likelihood function. It can be written as

$$P(D_{train} \mid M) = \int_{\theta} P(D_{train} \mid \theta) P(\theta \mid M) d\theta = \mathbb{E}_{P(\theta|M)} P(D_{train} \mid \theta). \quad (2.4.1.1)$$

This is Equation (1) in [Lyle et al., 2020].

When describing the model evidence $P(D_{train}|M)$ for a single model, we omit M and write $P(D_{train})$. For a model parameterised by θ, we have the following form of the log marginal likelihood:

$$\log P(D_{train}) = \log \prod_{i=1}^{n} P(D_i|D_{<i}) = \sum_{i=1}^{n} \log \left[\mathbb{E}_{P(\theta|D_{<i})} P(D_i|\theta) \right]. \quad (2.4.1.2)$$

From this equation, we see that a model which places more interim weight on parameters that yield high likelihoods will have a larger marginal likelihood.

Now consider two quantities:

$$\begin{aligned} \hat{\mathcal{L}}(D_{train}) &= \sum_{i=1}^{n} \frac{1}{k} \sum_{j=1}^{k} \log P(D_i|\boldsymbol{\theta_j^i}). \\ \hat{\mathcal{L}}_k(D_{train}) &= \sum_{i=1}^{n} \log \frac{1}{k} \sum_{j=1}^{k} P(D_i|\boldsymbol{\theta_j^i}). \end{aligned} \quad (2.4.1.3)$$

[Lyle et al., 2020] proved that both these quantities are estimators of a lower bound of Bayesian models' marginal likelihoods. Using the negative log likelihood as a loss function, they arrived at the conclusion that lower bounds of the model's marginal likelihood can be estimated by training statistics. Also, in the infinite-sample, infinite-training-time limit, the same rankings as the estimator are achieved by gradient descent on a linear ensemble.

In [Ru et al., 2020], SOTL and its variants were clearly defined. Empirically, Ru et al. [2020] showed that there is a strong correlation between SOTL and test accuracy and devised the neural architecture search based on this correlation. We consider the setting where we train a neural network parameterised by θ for T epochs. Denote the function implemented by the network by f_θ. Let there be B mini-batches in one epoch. We denote the network weights before training on the b-th mini-batch at the t-th epoch by $\theta_{t,b}$. Let $(\boldsymbol{X_b}, \boldsymbol{y_b})$ be the data and label on the b-th mini-batch. We can write the Sum over Training Loss in the recent E epochs (SOTL-E) as:

$$\text{SOTL-E} = \sum_{t=T-E+1}^{T} \left[\frac{1}{B} \sum_{b=1}^{B} R\left(f_{\boldsymbol{\theta_{t,b}}}\left(\boldsymbol{X_b}\right), \boldsymbol{y_b} \right) \right]. \tag{2.4.1.4}$$

When $E = T$, this recovers the original SOTL:

$$\text{SOTL} = \sum_{t=1}^{T} \left[\frac{1}{B} \sum_{b=1}^{B} R\left(f_{\boldsymbol{\theta_{t,b}}}\left(\boldsymbol{X_b}\right), \boldsymbol{y_b} \right) \right]. \tag{2.4.1.5}$$

An additional training speed estimator based on the moving average of training loss (TSE-EMA) can be written as:

$$\text{TSE-EMA} = \sum_{t=1}^{T} \gamma^t \left[\frac{1}{B} \sum_{b=1}^{B} R\left(f_{\boldsymbol{\theta_{t,b}}}\left(\boldsymbol{X_b}\right), \boldsymbol{y_b} \right) \right], \tag{2.4.1.6}$$

where $\gamma < 1$ is a decay constant. This averaging method weighs losses at the earlier stage of training more than those at the later stage.

3

Hypotheses and the Experimental Method

Contents

In the last chapter we saw the connections that could exist between training speed and other entities in the literature that explain generalisation from different aspects. We can also see how training speed can be connected to marginal likelihood and thus explain generalisation from a Bayesian perspective for linear models and infinitely wide models in [Lyle et al., 2020]. We wish to understand how training speed is related to generalisation for neural networks in practice, the setting of which is often non-Bayesian. In this chapter, we develop two hypotheses for the mechanism through which training speed is connected with generalisation for deep neural networks: namely, the robust flatness hypothesis and the gradient transfer hypothesis.

We use Figure 3.1 to express three possible relationships between training speed, a quantity whose identity we wish to find, and generalisation in our hypothesised mechanisms: Here the variable X denotes network flatness or the amount of gradient

transfer, in each of our two hypotheses respectively. Notice that it is difficult to distinguish between these two mechanisms, thus we only hypothesise about the identity of variable X. Figure 3.1a denotes a mechanism in which the variable X causes both training speed and generalisation; figure 3.1b denotes a mechanism in which training speed causes variable X, which in turn causes generalisation; figure 3.1c denotes a mechanism in which a different unknown factor than X causes both training speed and the variable X and X causes generalisation.

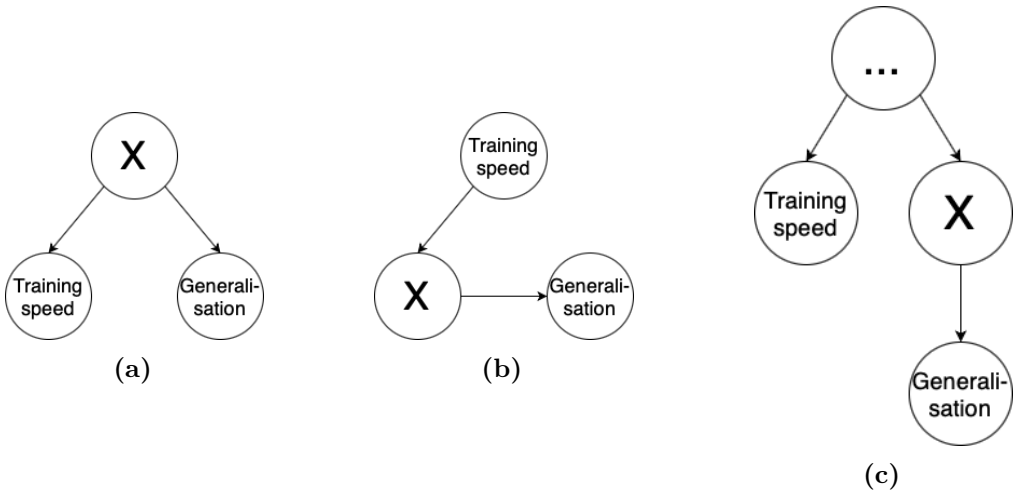


Figure 3.1: Three possible mechanisms through which training speed is related to generalisation. The node X denotes the factor we are hypothesising about. The node $\cdots$ in (c) stands for an unknown factor.

3.1 Two hypotheses

The Robust Flatness Hypothesis The Robust Flatness Hypothesis states that the correlation between training speed and generalisation is due to network flatness, i.e. $X =$ flatness. This connection should be observed when the training is robust, i.e. not diverged. We have seen in Subsection 2.3.2 how in various ways network flatness can impact generalisation. There are also multiple ways in which network flatness is connected with training speed: A larger learning rate might cause both a flatter network minimum and a faster training speed (which corresponds to Figure 3.1c). The correlation between flatness and training speed is mediated by the learning rate; Huang et al. [2019] argued that the high dimensionality of network parameter space

leads to a much larger volume of this space being associated with flatter minima which in turn means that a faster training speed is implied since they are easier to find (which corresponds to Figure 3.1a). This way, the correlation is immediate.

The Gradient Transfer Hypothesis The Gradient Transfer Hypothesis states that the correlation between training speed and generalisation is due to gradient transfer, i.e. X = the amount of gradient transfer. To define gradient transfer informally, it is the quantity measuring how well the gradient update on one mini-batch affects other mini-batches. Several formal definitions were introduced in Subsection 2.3.3. From a first-order approximation, this quantity is proportional to the dot product of the network gradients on two mini-batches. It is intuitively straightforward to understand the connection between training speed and gradient transfer: all other things being equal, more gradient transfer will lead to a faster training speed. In the first epoch of training, when all training images are shown to the model for the first time, different batches simulate the sampling of training and test batches when generalisation is calculated. In later batches, because all data points have been seen at least once, the simulation becomes approximated.

3.2 Method

In the practice of training deep neural networks, there are many practical techniques aiming to enhance the performance of models trained. The mechanisms by which they influence the generalisation of networks are known to different degrees. To understand how they affect the connection between training speed and generalisation, we conducted experiments to collect data for a large family of deep neural networks, employing a subset of all possible training techniques. For example, we investigated the effects of neural architecture choices (with/without convolutional layers, and with/without batch normalizations), data augmentation, and the noise in the stochastic gradient estimators. Using these results, we aim to find the conditions under which at least one of the candidate mechanisms we hypothesised might explain the connection between training speed and generalisation for deep neural networks.

The objective of our experiments is to collect data from networks that are representative of how deep learning models are used in practice. To this end, we generated a large family of experiments, to cover popular network architectures, optimisers, data augmentation schemes, etc. Namely, we wish to investigate the effects of the following factors during network training:

1. Using data augmentation by image transformations;
2. Using Logit averaging [Nabarro et al., 2021];
3. Using Stochastic gradient Langevin dynamics [Welling and Teh, 2011];
4. Not using convolutional layers;
5. Using Batch Normalization [Ioffe and Szegedy, 2015].

3.2.1 Experimental setup

We chose the CIFAR10 [Krizhevsky et al., 2009] visual classification task. The CIFAR10 dataset contains 60,000 RGB images of 32×32 pixels. The task is to classify the images into one of ten categories. Of the 60,000 images, 50,000 are used for training and 10,000 are used for validation. See Figure 3.2 for 10 examples from each category.

We employed the following strategy to conduct controlled experiments: we first trained multiple batch-norm-free networks with different widths, depths, and learning rates, using SGD as the optimiser without data augmentation. The network training is terminated once the loss on the training set reaches 0.01 or when it has been trained for 400 epochs. This group of networks is our control group. Then, we trained 5 experimental groups of networks corresponding to the 5 factors we wish to investigate. For each experimental group, we used the same setting as the control group, except for one hyperparameter whose effect we want to investigate. Each group contains 45 experiments with different hyperparameter combinations. We present the full results for the experiments in Appendix A. We start by showing the average test accuracies for each group of the experiments, as primary results.

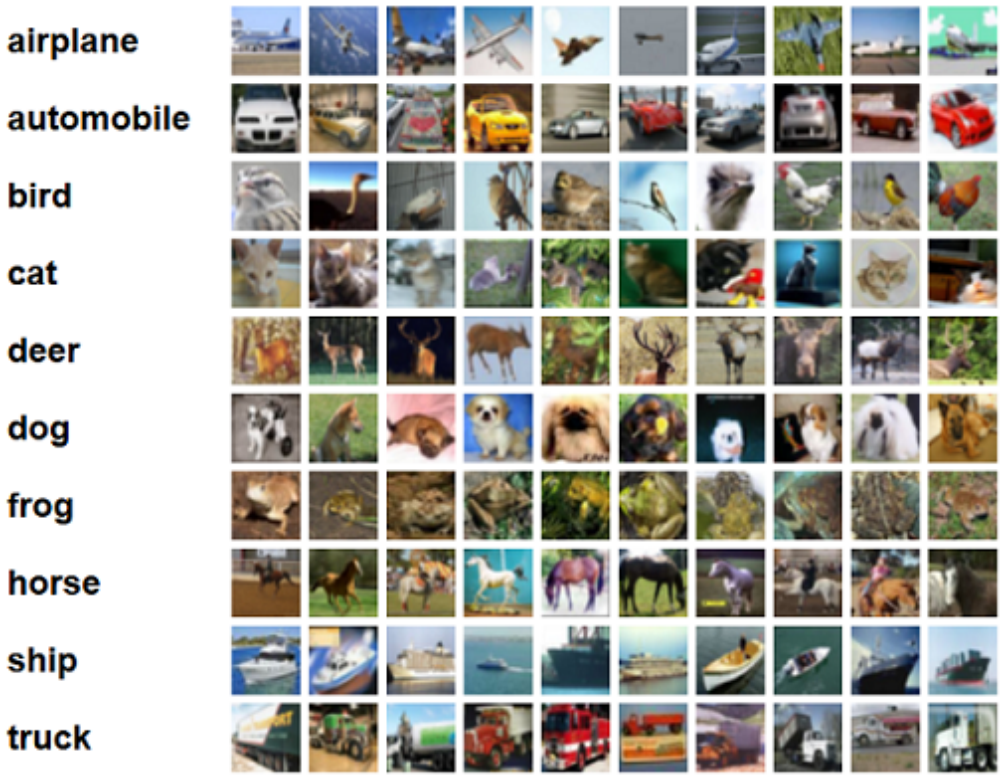


Figure 3.2: Example images from the CIFAR10 dataset. Retrieved from the nutsml documentation.

To determine how good a generalisation complexity measure is, Jiang et al. [2020] used two evaluation criteria: conditional mutual information and Kendall's rank-correlation coefficient. We introduce them below and analyse the experimental results using these two criteria as our main tools.

3.2.2 Sizes of neural architectures

We present the sizes of the neural architectures used in the experiments, in terms of the number of parameters, in Table 3.1.

3.2.3 Important quantities

In this section, we list the complexity measures that we monitor in the experiments, and explain why they are meaningful. Also, we introduce the quantities used in the analysis of causal mechanisms and correlations.

Architecture	Base	MLP	ResNet
Depth=2, width=8	171,574	205,440	268,336
Depth=2, width=12	385,230	314,304	602,952
Depth=2, width=16	684,134	427,264	1,071,200
Depth=3, width=8	268,345	209,536	461,872
Depth=3, width=12	602,961	323,520	1,038,408
Depth=3, width=16	1,071,209	443,648	1,845,344
Depth=4, width=8	365,116	213,632	558,640
Depth=4, width=12	820,692	332,736	1,256,136
Depth=4, width=16	1,458,284	460,032	2,232,416

Table 3.1: Number of trainable parameters in each of the architectures used.

Training speed based measures There are a group of 3 entities that are estimators of the training speed: SOTL, SOTL-E_50, and TSE-EMA. SOTL-E_50 is the value of SOTL-E when $E = 50$. SOTL, SOTL-E, and TSE-EMA are defined in equations 2.4.1.5, 2.4.1.4, and 2.4.1.6, respectively. These are the indicators of training speed that we wish to study. SOTL-E_50 and TSE-EMA are variants of SOTL, focusing on later and earlier stages. By observing their values, we can investigate the importance of training speed at different stages in predicting generalisation.

Sharpness-based measures We consider two measures related to sharpness: PAC-Bayes sharpness (Equation 2.3.2.4) and value sensitivity. We experimented with the other sharpness-based measures introduced in Subsection 2.3.2 in the preliminary experiments and found that these two are the best at predicting generalisation. Value sensitivity is an entity closely related to PAC-Bayes sharpness. Concretely, we define it as:

$$\text{Value sensitivity} = \mathbb{E}_{\theta' \sim \mathcal{N}(\theta, 0.01I)} \left[R_{emp} \left(\theta', D_{train} \right) \right]. \tag{3.2.3.1}$$

Gradient variance based measures We monitor the gradient variance measure defined by Equation 2.3.3.4, at two points in training: when the neural network

has seen all the training data once (at the end of epoch 1), and when the network has finished training. We denote these two quantities by "Initial gradient variance" and "Final gradient variance" respectively.

Average test accuracy To give brief indications of how well each group of experiments would perform in reality, we calculate the average test accuracies across each group.

Normalised Conditional Mutual Information (NCMI) We represent two possible sets of causal relations between the network parameters θ, the complexity measure μ, and the generalisation error g in Figure 3.3. We wish to have a criterion that tells us how good a generalisation measure is by indicating the relative likelihood of the mechanism in Figure 3.3a to that of Figure 3.3b.

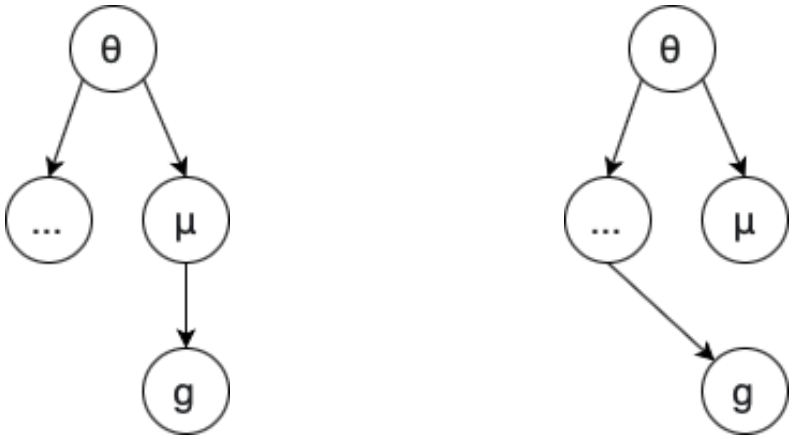


Figure 3.3: (a) The complexity measure μ can perfectly explain the generalisation error. **(b)** The generalisation error g might be explained by some hidden factor and cannot be explained by μ.

Inspired by the Inductive Causation (IC) Algorithm [Verma and Pearl, 1990], Jiang et al. [2020] perform a conditional independence test for each complexity measure. They do so by reading the mutual information between μ and g, conditioned on a set of hyperparameters. In our analysis, we place the condition on a single hyperparameter out of the three that we vary for each group: learning rate, network depth, and network width. We present the weighted average of

the conditional mutual information, as well as its minimum, across different hyperparameter combinations.

Concretely, let h be a hyperparameter, S be a value of h, and U_S be the set of experiments whose hyperparameter h has value S. Let $\theta : \Theta$ be the parameters of the network, $\mu : \Theta \to \mathbb{R}$ be a complexity measure function and $g : \Theta \to \mathbb{R}$ be the generalisation error function. We can then define $V_\mu :\equiv \text{sign}(\mu(\theta_1) - \mu(\theta_2)) : \Theta_1 \times \Theta_2 \to \{-1, 1\}$ and $V_g :\equiv \text{sign}(g(\theta_1) - g(\theta_2)) : \Theta_1 \times \Theta_2 \to \{-1, 1\}$. Now, we have

$$\mathcal{I}(\mu, g \mid h) = \sum_S \sum_{U_S} p(U_S) \sum_{V_\mu \in \pm 1} \sum_{V_g \in \pm 1} p(V_\mu, P_g \mid U_S) \log \left(\frac{p(V_\mu, P_g \mid U_S)}{p(V_\mu \mid U_S) p(P_g \mid U_S)} \right). \tag{3.2.3.2}$$

We call this quantity the conditional mutual information (CMI). The higher the CMI, the more likely that there exists an edge between μ and g, under the hyperparameter h.

Since we wish to arrive at a criterion with a fixed range, we can first calculate the conditional entropy of generalisation:

$$H(V_g \mid h) = -\sum_S \sum_{U_S} p(U_S) \sum_{V_g \in \pm 1} p(V_g \mid U_S) \log p(V_g \mid U_S), \tag{3.2.3.3}$$

and use it to normalise the conditional mutual information:

$$\hat{\mathcal{I}}(\mu, g \mid h) = \frac{\mathcal{I}(\mu, g \mid h)}{H(V_g \mid h)}. \tag{3.2.3.4}$$

We denote the above entity by Normalised Conditional Mututal Information (NCMI). According to the IC algorithm, an edge is kept if there is no hyperparameter h such that the two nodes are independent. We calculate the minimum value of the normalised conditional mutual information across hyperparameters for each group of experiments:

$$\mathcal{K}(\mu) = \min_h \hat{\mathcal{I}}(\mu, g \mid h). \tag{3.2.3.5}$$

In addition, we also calculate the average conditional mutual information, denoted by $\bar{\mathcal{K}}(\mu)$:

$$\bar{\mathcal{K}}(\mu) = \sum_h p(h) \hat{\mathcal{I}}(\mu, g \mid h). \tag{3.2.3.6}$$

Kendall's rank-correlation coefficient Ideally, for each of the generalisation complexity measures, we wish that they can be positively correlated with the generalisation error (i.e. the smaller the value of the complexity measure, the lower the generalisation error). However, the NCMI introduced in the last part does not tell us anything about the positivity of the correlation. Hence, we introduce Kendall's Rank-Correlation Coefficient [Kendall, 1938]. It uses ranking to evaluate the quality of complexity measures.

Concretely, let U be the set of all experiments in the group, $\theta_1, \theta_2, \ldots, \theta_{|U|}$ be the network parameters of each of the experiments, $\mu : \Theta \to \mathbb{R}$ be the complexity measure function, and $g : \Theta \to \mathbb{R}$ be the generalisation error function. Kendall's rank coefficient is defined as:

$$\tau(U; \mu, g) :\equiv \frac{1}{\mid U \mid (\mid U \mid -1)} \sum_{\theta_1 \in U} \sum_{\theta_2 \in U \setminus \theta_1} \operatorname{sign}\left(\mu(\theta_1) - \mu(\theta_2)\right) \operatorname{sign}\left(g(\theta_1) - g(\theta_2)\right). \tag{3.2.3.7}$$

It can easily be verified that Kendall's rank coefficient falls into the range of $[-1, 1]$. When the ranking of the complexity measure is the same as that of the generalisation error, the coefficient is 1. When the ranking of the complexity measure is the reverse of the generalisation error, the coefficient is -1.

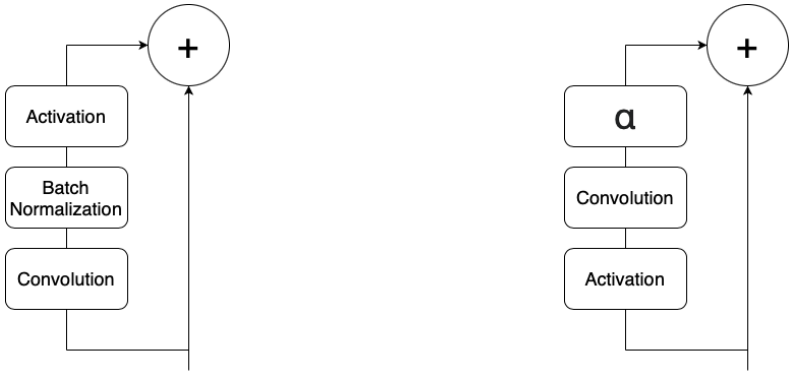
4
Results and Analysis

Contents

Below we present the detailed setup of each group of experiments, as well as results. We analyse the results with the central question of whether the condition presented allows for one of the hypotheses to hold.

4.1 The Control Group

Here we specify the experimental setup for our control group. We use the architecture with the SkipInit initialisation as described by De and Smith [2020]. We refer to this architecture as the "base architecture" since it is used in the control group experiments. The base architecture is similar to Residual Networks [He et al., 2016], except that it doesn't use batch normalization [Ioffe and Szegedy, 2015].



(a) The Residual block **(b)** The SkipInit block

Figure 4.1: Comparison between the Residual block and the SkipInit block. Notice that in the SkipInit block, the position of convolution and activation is also swapped, as specified in the paper of De and Smith [2020].

The SkipInit block and architecture In Figure 4.1 we illustrate the Residual block and a SkipInit block. The main difference is that in the SkipInit block there is no Batch Normalization layer. Instead, it uses a learnable multiplicative parameter α which is initialised to be zero. This is because we wish to study the effects of Batch Normalization explicitly, which changes the training dynamics significantly, in one of our experimental groups. Hence in the control group, we deliberately use a Batch-Normalization-free architecture.

De and Smith [2020] found that the base architecture using SkipInit can achieve comparable performances with Residual Networks, under some assumptions on the training regime. Since we wish to study the effect of batch normalizaiton explicitly, we need to reserve architectures with batch normalization for an experimental group and use the base architecture for our control group.

In the control group of experiments, we conduct a grid search over the following hyperparameters: learning rates, network depths, and network widths. We use 5 different learning rates [0.001, 0.0016, 0.003, 0.006, 0.01], 3 network widths [8, 12, 16], and 3 network depths [2, 3, 4]. Each network has 3 stages. For a network with width w and depth d, each stage consists of d SkipInit blocks. The three stages have $2 * w$, $4 * w$, $4 * w$ channels respectively.

We use the SGD optimiser to train our networks on the CIFAR10 [Krizhevsky et al., 2009] dataset. The loss criterion we use is cross entropy.

We present some useful entities for the control group in Table 4.1.

Control group	Average test accuracy: 71.1 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\,(U;\mu,g)$	$\tau\,(U;\mu,\text{TSE-EMA})$
SOTL	**0.130**	**0.325**	**0.640**	**0.949**
SOTL-E_50	0.015	0.050	-0.228	-0.125
TSE-EMA	0.123	0.310	0.610	1.00
PAC-Bayes sharpness	0.071	0.266	0.392	0.642
Value sensitivity	0.037	0.262	0.400	0.644
Initial gradient variance	0.063	0.181	-0.436	-0.737
Final gradient variance	0.010	0.030	0.166	0.071

Table 4.1: The important analytical quantities mentioned in Section 3.2.3 of the complexity measures considered in the control group. Bold values indicate the largest value out of the complexity measures by magnitude. U is the set of all experiments in the group and g is the generalisation error.

Training speed estimators predict generalisation well We notice that the SOTL measure has the highest values for $\mathcal{K}$ (0.130) and $\bar{\mathcal{K}}$ (0.325). It is also the most correlated measure with the generalisation error ($\tau = 0.640$) out of the measures we consider. The measure TSE-EMA closely follows in the same three entities, taking the values of 0.123, 0.310, and 0.610 respectively. SOTL and TSE-EMA are also highly correlated themselves-the τ value between them is 0.949. This tells us that the early training speed is highly correlated with the training speed of the entire training process.

The SOTL-E_50 measure is negatively correlated with the generalisation error. This means that when the sum of the training losses over the last 50 epochs is larger, the generalisation error is likely to be smaller. This also supports the theory that the faster training speed is an indicator of good generalisation: we terminate training when the cross entropy reaches 0.01, therefore a larger SOTL-E_50 indicates a faster-declining loss in the last 50 epochs and hence faster training speed.

The flatness-based measures (PAC-Bayes sharpness and value sensitivity) behave quite similarly. They both have moderate values of $\mathcal{K}$ and $\bar{\mathcal{K}}$: 0.071 and 0.266

for PAC-Bayes sharpness, 0.037 and 0.262 for value sensitivity. They are also positively correlated with the generalisation error and TSE-EMA, having very similar coefficients in both correlations.

Initial gradient variance behaves differently when different learning rates are used The initial gradient variance is the most negatively correlated measure with the generalisation error, having a correlation coefficient of -0.436, out of the seven considered. This suggests that a larger gradient variance in the early training is likely to indicate a smaller generalisation error of the final network. It is also the most negatively correlated measure with TSE-EMA, the value of τ being -0.737. This tells us that faster training speed in the earlier stage is correlated with big gradient variance. Since the training speed is very closely correlated with the learning rate, we want to investigate whether this dependence still exists if conditioned on the learning rate hyperparameter.

To this end, we first calculated the correlation coefficient between the initial gradient variance and the learning rate in the group, which is 0.604. This suggests that the larger the learning rate, the larger the initial gradient variance. Then, we calculated the correlation coefficients between the generalisation error and the initial gradient variance, as well as between TSE-EMA and the initial gradient variance. We present the coefficients with different learning rate groups in Table 4.2.

Control group	μ = initial gradient variance	
Learning rate	$\tau(U;\mu,g)$	$\tau(U;\mu,\text{TSE-EMA})$
0.001	0.167	-0.333
0.0016	0.111	-0.388
0.003	0.111	-0.388
0.006	-0.388	-0.611
0.01	-0.611	-0.833

Table 4.2: The correlation coefficients between the initial gradient variance and the generalisation error, as well as between the initial gradient variance and TSE-EMA in the control group.

It is easy to see that there is the trend that when the learning rate increases, the correlation between initial gradient variance becomes more negative, as does the correlation between initial gradient variance and TSE-EMA. The difference for different learning rates is large: the coefficient between initial gradient variance and generalisation error changes from 0.167 for the learning rate 0.001 to -0.611 for the learning rate 0.01. The coefficient between initial gradient variance and TSE-EMA changes from -0.333 to -0.833. This suggests that for all learning rates considered, greater initial gradient variance is correlated with faster training speed. For smaller learning rates, greater initial gradient variance is correlated with worse generalisation, while for larger ones, it is correlated with better generalisation.
The final gradient variance as a measure has small and even negligible values in the quantities present in the table.

This section presented the results from the control group. Key takeaways are:

- Training speed estimators can predict generalisation well.
- Flatness and gradient transfer measures are correlated with generalisation as well as TSE-EMA.
- Initial gradient variance has more negative correlations with generalisation and TSE-EMA as the learning rate increases.

4.2 The Experimental Groups

We now study the relationship between complexity measures and generalisation when we make different design choices described in the control group.

4.2.1 Data transformation

Data augmentation is one of the most popular ways to increase the performance of neural networks for visual tasks. We want to investigate its effects on generalisation. Here we explain in detail the data augmentation scheme used and how logit averaging is applied in our experimental groups.

Data augmentation by invariant transformation In the data augmentation experimental group, for each input image, we perform a random crop of size 32×32 pixels, with a padding of 4 pixels. Then with a probability of 0.5, the cropped image is horizontally flipped. The basis for this transformation is that visual classification should be invariant to these transformations, i.e. an image slightly cropped or flipped horizontally should have the same label as the original image. The resulting image will then be used for network training.

From a Bayesian perspective, however, this procedure is likely to result in an overconfident network. This is because we treat each of the transformed images as an independent image, while they are not since the ground truth of each transformed image is assumed to be identical to the original one. Because we assumed more training data than we really have with this augmentation, the posterior distribution will be influenced by the likelihood more than it should. This is particularly troublesome from the Bayesian perspective: we assume that the data is fixed, while transformation can create infinitely many data points potentially. In the infinite data-point case, the model will ignore the prior completely.

Logit averaging Nabarro et al. [2021] pointed out that one way to circumvent this is to regard the transformed images as drawn out of a distribution parameterised by the original image and the loss as the expectation over the distribution. Concretely, let x be the original image, y be the true target, θ be the network parameters, f_θ be the function that the network implements, R be the original loss, and T_x be the distribution of transformed images. Let the output of the network be unnormalised, i.e. passing the network output through a softmax function gives the output probabilities. These outputs are also called logits. We can write the modified loss as:

$$R_{modified}\left(x, y; \theta\right) = R\left(\mathbb{E}_{x' \sim T_x(\cdot)}[f_\theta(x')], y\right). \tag{4.2.1.1}$$

Then, we can think of data augmentation by transformation as a new way of defining the likelihood function, by taking the expectation over a fixed distribution.

In practice, this modified loss can be estimated by logit averaging. The idea of logit averaging [Nabarro et al., 2021, van der Wilk et al., 2018, Dao et al., 2019] is

that, for each data point, we perform multiple transformations. Then the multiple transformed images are fed into the neural network, resulting in multiple logit vectors. The logit vectors are then averaged over each data point and loss is calculated with the averaged logit vector and the target.

A Monte Carlo estimator is the result of logit averaging:

$$\hat{R}_{modified}(x, y; \theta) = -\log \text{softmax}_y[\sum_{i=1}^{K} f_\theta(x_i)], \tag{4.2.1.2}$$

where $x_1, x_2, \ldots, x_K \sim T_x(\cdot)$. To put the idea into a computational context, we present the pytorch-like pseudocode in Algorithm 1.

Algorithm 1: Logit averaging algorithm

Input: Dataloader D, neural network N, duplication factor K
for *data d, target t in D* **do**
  // Tensor d has shape $batch\_size \times 3 \times 32 \times 32$
  Transform the data
  $d' \leftarrow duplicate\_and\_transform(d, duplicates = K)$;
  // Tensor d' has shape $(K * batch\_size) \times 3 \times 32 \times 32$
  Logits $l \leftarrow N$.forward(d');
  // Tensor l has shape $(K * batch\_size) \times 10$
  Averaged logits $l' \leftarrow$ torch.mean(l.view(K, $batch\_size$, 10),dim=1);
  Loss $L \leftarrow$ criterion(l', t);
  L.backward();
end

In our experiments, we used the same transformations as the data augmentation experiments, and a duplication factor of $K = 4$. At both training and test time, losses were calculated by doing transformation and logit averaging (except no backpropagation at test time). The generalisation gap and related complexity measures were also calculated based on this loss.

To train a model for B epochs with logit averaging, the computational budget is K times that of training the same model without logit averaging. Because the training loss in the logit averaging experiments decreases much more slowly, we change its maximum number of epochs to 250, which uses the same compute as 1000 epochs without logit averaging (compared to 400). Also, the cross entropy milestone is

changed to 0.25 (from 0.01). This change can be justified as it has similar effects on the test accuracies as the straightforward data augmentation experiments.
We compare and contrast the results from these two experimental groups, as well as those from the control group, in Table 4.3.

Data augmentation improves flatness and generalisation Comparing the data augmentation group with the control group, we found that the average test accuracy increases by 13.9%, from 71.1% to 85.0%. Additionally, the correlation coefficients of complexity measures with the generalisation error are much smaller. Notably, the SOTL measure, having a correlation coefficient of 0.640 in the control group, becomes negatively correlated with the generalisation error in the data augmentation group. The TSE-EMA measure is strongly correlated with the generalisation error in the control group, with a coefficient of 0.610, but the coefficient becomes 0.046 in the data augmentation group. The only complexity measure that increases in the correlation coefficient is the initial gradient variance measure, from -0.436 to 0.073.
In terms of the correlation with TSE-EMA, a noticeable change happens for the final gradient variance measure. Its correlation coefficient increases from 0.071 to 0.768, i.e. the final gradient variance changes from nearly uncorrelated to strongly positively correlated with TSE-EMA.
To find a potential reason for the improvement in performance, we compare the flatness and the generalisation error of pairs of experiments in the two groups whose only hyperparameter difference is whether data augmentation is applied. The results of experiments in the data augmentation group compared to the control group are presented in Table 4.4.
We can see clearly that in all experiments, employing data augmentation improves both the flatness of the minima found and the generalisation performance. This tells us that it's possible that the improvement in the performance of the data augmentation group can be explained by flatness changes.

Control group	Average test accuracy: 71.1 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau(U;\mu,g)$	$\tau(U;\mu,\text{TSE-EMA})$
SOTL	**0.130**	**0.325**	**0.640**	**0.949**
SOTL-E_50	0.015	0.050	-0.228	-0.125
TSE-EMA	0.123	0.310	0.610	1.00
PAC-Bayes sharpness	0.071	0.266	0.392	0.642
Value sensitivity	0.037	0.262	0.400	0.644
Initial gradient variance	0.063	0.181	-0.436	-0.737
Final gradient variance	0.010	0.030	0.166	0.071
Data augmentation group	Average test accuracy: 85.0 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau(U;\mu,g)$	$\tau(U;\mu,\text{TSE-EMA})$
SOTL	0.116	0.222	-0.097	**0.820**
SOTL-E_50	0.032	0.140	**-0.327**	0.549
TSE-EMA	0.134	**0.182**	0.046	1.00
PAC-Bayes sharpness	0.051	0.115	0.172	0.717
Value sensitivity	**0.146**	0.173	0.202	0.756
Initial gradient variance	0.097	0.140	0.073	-0.687
Final gradient variance	0.092	0.125	0.032	0.768
Logit averaging group	Average test accuracy: 82.6 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau(U;\mu,g)$	$\tau(U;\mu,\text{TSE-EMA})$
SOTL	0.035	0.154	0.414	**0.907**
SOTL-E_50	**0.047**	0.138	-0.354	-0.626
TSE-EMA	0.029	0.150	0.428	1.00
PAC-Bayes sharpness	0.038	**0.156**	**0.473**	0.651
Value sensitivity	0.031	0.138	0.469	0.644
Initial gradient variance	0.023	0.112	-0.339	-0.685
Final gradient variance	0.027	0.152	0.469	0.721

Table 4.3: The important analytical quantities mentioned in Section 3.2.3 of the complexity measures considered in the three groups of experiments. Bold values indicate the largest value out of the complexity measures in the group by magnitude.

Training speed estimators need rethinking for data transformation Ideally, we should wish that the complexity measures be positively correlated with

	Data augmentation has better Generalisation	Data augmentation has worse generalisation
Higher sharpness	0	0
Lower sharpness	45	0

Table 4.4: The sharpness and generalisation compared for the data augmentation group and the control group. Entries are the number of experiment pairs fitting the description.

the generalisation error, and for a mechanism to uphold, we should wish that a representative measure of the mechanism to correlate with TSE-EMA. These training speed estimators are either barely positively correlated, or negatively correlated with the generalisation error. So, not using these data transformations is a necessary condition for analysing our hypotheses. However, this is not necessarily the fault of data transformation itself, it could be that the way we calculate the training loss becomes unsuitable when data transformation is present: the training losses are calculated with the transformed images while the generalisation error is calculated with original images. The distribution of the transformed images is shifted from the distribution of the original CIFAR10 images while we use the same loss to indicate the training speed. This discrepancy could make the training speed estimators unsuitable as generalisation measures. To verify that the training speed estimators can work when data transformation is present, we look at the logit averaging group.

Logit averaging and modified loss restores training speed estimators as generalisation measures The average test accuracy increases by 11.5%, from 71.1% in the control group, to 82.6% in the logit averaging group. This is a large increase and makes its effect similar to that brought by the data transformations in the previous group. This validates our change of training termination criteria: the logit averaging applied can represent a computationally feasible practice with good performance.

We find that doing logit averaging and calculating the loss in this way makes the training speed estimators behave as we expect: SOTL has a correlation coefficient of 0.414 with the generalisation error, SOTL-E_50 -0.354, and TSE-EMA 0.428.

This trend is consistent with that found in the control group experiment (with no data transformation).

4.2.2 Variance of the stochastic gradients

As we introduced in Subsection 2.3.1, the SGD optimiser update follows the stochastic gradient. The network weights are updated according to Equation 2.3.1.2. The variance in the gradient purely comes from the stochasticity of the mini-batches. We can investigate the effect of the gradient variance by explicitly adding Gaussian noise to the gradient estimator.

Stochastic Gradient Langevin Dynamics Stochastic Gradient Langevin Dynamics [Welling and Teh, 2011] is a process during which, noise is added to the gradient updates. Specifically, we can write the change in the network parameters for one step of SGD is:

$$\Delta\theta_{SGD} = \eta \cdot \nabla_{\boldsymbol{\theta}} R_{emp}\left(\boldsymbol{\theta}, \{\mathbf{x_j}, y_j\}_{j=i}^{i+n-1}\right). \tag{4.2.2.1}$$

But for one step of SGLD, the change in parameters is:

$$\Delta\theta_{SGLD} = \eta \cdot \nabla_{\boldsymbol{\theta}} R_{emp}\left(\boldsymbol{\theta}, \{\mathbf{x_j}, y_j\}_{j=i}^{i+n-1}\right) + \epsilon, \tag{4.2.2.2}$$

where $\epsilon \sim \mathcal{N}(0, 2\eta)$. Asymptotically, SGLD allows sampling from the posterior distribution of the network weights. Compared with the SGD update rule, we can easily see that because ϵ is sampled independently, we have

$$\mathbb{V}\left[\Delta\theta_{SGLD}\right] = \mathbb{V}\left[\Delta\theta_{SGD}\right] + \mathbb{V}\left[\epsilon\right] \tag{4.2.2.3}$$

$$= \mathbb{V}\left[\Delta\theta_{SGD}\right] + 2\eta. \tag{4.2.2.4}$$

This is to say, the variance in the SGLD gradient update is the sum of the variance in the SGD update and the variance of the Gaussian noise, which is proportional to the learning rate.

To investigate the effects of SGLD, we conduct experiments identical to those in the control group, except with SGLD optimisers instead of SGD optimisers. We

used the PyTorch [Paszke et al., 2019] implementation of the SGLD optimisers with default parameters except for learning rates. In practice, there are issues with preconditioning, pseudo batching, burning-in, and diagonal bias for the stability and the sampling quality of the optimiser, which we do not go into detail about. In our experiments, we used the following parameters: the dataset size is 50000, the precondition decay rate is 0.95, the number of pseudo batches is 1, the number of burn-in steps is 3000, and the diagonal bias is 1×10^{-8}.

We present the experimental results for the SGLD group, as well as the control group, in Table 4.5.

Control group	Average test accuracy: 71.1 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\left(U;\mu,g\right)$	$\tau\left(U;\mu,\text{TSE-EMA}\right)$
SOTL	**0.130**	**0.325**	**0.640**	**0.949**
SOTL-E_50	0.015	0.050	-0.228	-0.125
TSE-EMA	0.123	0.310	0.610	1.00
PAC-Bayes sharpness	0.071	0.266	0.392	0.642
Value sensitivity	0.037	0.262	0.400	0.644
Initial gradient variance	0.063	0.181	-0.436	-0.737
Final gradient variance	0.010	0.030	0.166	0.071
SGLD group	Average test accuracy: 78.4 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\left(U;\mu,g\right)$	$\tau\left(U;\mu,\text{TSE-EMA}\right)$
SOTL	0.044	0.140	0.236	**0.669**
SOTL-E_50	0.041	0.116	-0.040	0.246
TSE-EMA	**0.171**	**0.251**	**0.475**	1.00
PAC-Bayes sharpness	0.015	0.048	-0.253	-0.022
Value sensitivity	0.041	0.060	-0.154	-0.085
Initial gradient variance	0.057	0.088	-0.048	0.251
Final gradient variance	0.026	0.032	-0.176	0.192

Table 4.5: The important analytical quantities mentioned in Section 3.2.3 of the complexity measures considered in the three groups of experiments. Bold values indicate the largest value out of the complexity measures in the group by magnitude.

The average test accuracy increases by 7.3%, from 71.1% in the control group to

78.4% in the SGLD group. This suggests that in practice, adding Gaussian noise to the stochastic gradient update helps regularise the neural network effectively, in the absence of data augmentation. From the Bayesian perspective, approximately sampling from the posterior of the neural network weights (exact sampling requires infinite training iterations) increases the performance of the examined neural models.

TSE-EMA retains predictive power while other training speed estimators fail We can observe that for the SOTL measure, the NCMI values and the correlation coefficient with the generalisation error all decrease in the SGLD group. Its minimal NCMI plunges from 0.130 to 0.044, average NCMI drops from 0.325 to 0.116, and the correlation coefficient changes from 0.640 to 0.236. For the SOTL-E_50 measure, the correlation with generalisation error also becomes weaker for the SGLD group: the correlation coefficient magnitude changes from 0.228 to 0.004, i.e. the correlation is negligible. However, the TSE-EMA measure, out of all three training speed estimators, increased in minimal NCMI from 0.123 to 0.171.

Sharpness-based measures fail when SGLD is applied Noticeably, the sharpness-based measures (PAC-Bayes sharpness and value sensitivity) become negatively correlated with the generalisation error. The correlation coefficient of PAC-Bayes sharpness drops from 0.392 to -0.253, and that of value sensitivity drops from 0.400 to -0.154. This means that sharpness measures fail when the variance of stochastic gradient updates is increased by applying SGLD. Their correlations with TSE-EMA, the effective training speed estimator, are also very weak (correlation coefficients being -0.022 and -0.085).

To confirm that the intervention applied to the variance of the stochastic gradient updates makes the sharpness measures useless, we present Table 4.6. For each of the 45 experiments in the SGLD group, we find the experiment in the control group with the same hyperparameters, apart from the optimiser used, and compare their generalisation and flatness.

This table shows that for all 45 pairs of experiments, the one trained with SGLD has better generalisation performance. However, the change in the sharpness of the

	Better generalisation	Worse generalisation
Higher sharpness	24	0
Lower sharpness	21	0

Table 4.6: The sharpness and generalisation compared for the SGLD group and the control group.

final network is largely random. Therefore, we can conclude that not only does this intervention remove the connection between network flatness (in the PAC-Bayes and value sensitivity sense) and generalisation error, but also the intervention does not bias the flatness in one way or another.

It is clear that training speed's predictive power of generalisation is not only due to its connection with flatness, otherwise it won't exhibit such a good predictive performance in the SGLD group.

4.2.3 Ablation of convolutional layers

Convolutional networks [LeCun et al., 1995] have become a necessary component in the models applied to the computer vision domain. The weight-sharing convolutional layers that embody the translational equivariance property in many visual problems are a very effective way of presenting inductive bias in neural networks. As part of our neural architecture investigation, we look at the networks without convolutional layers, namely, multi-layer perceptrons.

The multi-layer perceptron architecture Multi-Layer Perceptrons (MLPs) are one of the earliest neural network architectures. Here we briefly introduce what functions an MLP can implement. Let there be an L-layer MLP used for classification with its weights and bias at layer l being θ_l and b_l, the activation function used be σ, and the input be x. We can inductively define the function

that the network implements:

$$f_{\theta_1}(x) = \sigma(\theta_1 x + b_1); \tag{4.2.3.1}$$

$$f_{\theta_l}(x) = \sigma(\theta_l f_{\theta_{l-1}}(x)) + b_l), \text{for } l \geq 1; \tag{4.2.3.2}$$

$$f_\theta(x) = f_{\theta_L}(x), \tag{4.2.3.3}$$

where $f_\theta(x)$ is the output logits of the network. Passing the logits through a softmax layer and we get the output probabilities for input x. The biggest difference between MLPs and the base architecture we used in the control group is that it doesn't have convolutional layers [LeCun et al., 1995].

To investigate the effects of not having convolutional layers in the network architecture, we repeat the experiments of the control group, except with MLPs as the network architecture. We present the important quantities of the MLP group and the control group in Table 4.7.

The lack of inductive bias makes generalisation very poor in MLPs The table suggests that the MLP group only achieves 47.4% test accuracy on average. This is not surprising for a neural architecture without a convolutional structure. Although there are creative ways to combine MLPs to achieve better results for CIFAR10 [Tolstikhin et al., 2021], we believe these experiments can represent the typical performance of MLPs on visual tasks. There could be two possible reasons for this: (1) MLPs are very inexpressive architecture which cannot effectively solve the visual classification problem (2) MLPs do not have the implicit translational equivariance inductive bias and generalise poorly. One observable consequence for these two reasons is that if (1) were true, the training accuracy would also be very low for MLPs, which would not be the case if (2) were true. Observing the experimental data, we found that in the MLP group, the worst training accuracy at the end of training is $> 90\%$. Therefore we conclude that MLPs are expressive enough to overfit visual classification datasets like CIFAR10 but generalise poorly.

Control group	Average test accuracy: 71.1 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\left(U;\mu,g\right)$	$\tau\left(U;\mu,\text{TSE-EMA}\right)$
SOTL	**0.130**	**0.325**	**0.640**	**0.949**
SOTL-E_50	0.015	0.050	-0.228	-0.125
TSE-EMA	0.123	0.310	0.610	1.00
PAC-Bayes sharpness	0.071	0.266	0.392	0.642
Value sensitivity	0.037	0.262	0.400	0.644
Initial gradient variance	0.063	0.181	-0.436	-0.737
Final gradient variance	0.010	0.030	0.166	0.071
MLP group	Average test accuracy: 47.4 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\left(U;\mu,g\right)$	$\tau\left(U;\mu,\text{TSE-EMA}\right)$
SOTL	0.012	0.155	0.152	0.651
SOTL-E_50	**0.123**	0.153	**-0.343**	0.063
TSE-EMA	0.034	0.133	0.210	1.00
PAC-Bayes sharpness	0.024	0.163	0.186	0.677
Value sensitivity	0.042	**0.189**	0.220	**0.715**
Initial gradient variance	0.013	0.082	-0.111	-0.663
Final gradient variance	0.091	0.113	-0.149	0.305

Table 4.7: The important analytical quantities mentioned in Section 3.2.3 of the complexity measures considered in the three groups of experiments. Bold values indicate the largest value out of the complexity measures in the group by magnitude.

Most complexity measures are bad generalisation measures with MLPs

We can observe from the table that most complexity measures monitored have low minimal NCMI (< 0.1) with the generalisation error, apart from SOTL-E_50, which has a $\mathcal{K}$ value of 0.123. This suggests that the probability that a complexity measure causes generalisation is low for most. The relatively higher NCMI from SOTL-E_50 suggests that the training speed near the end of the training is indicative of the generalisation error.

For memorisation-prone architectures like MLPs that overfit severely, training speed loses its predictive power of generalisation. The connection between training speed and generalisation benefits from strong inductive biases in the neural architecture.

4.2.4 Batch normalization

Batch Normalization [Ioffe and Szegedy, 2015] is a technique frequently used in training deep neural networks. By reducing internal covariate shift, it accelerates training deep neural networks. As a result, it allows for larger learning rates and more network layers to be used. As we have shown in Figure 4.1, the difference between the Residual block and the SkipInit block is mainly the use of batch normalization. To investigate the effects of batch normalization, we employ networks using the Residual blocks, namely, ResNets [He et al., 2016].

The ResNet neural architecture As we briefly introduced in Figure 4.1, the residual block in a network is a structure that adds the original input and the input after processing together, instead of simply processing it. As a result, this block allows layers to be bypassed if they are not beneficial. It also avoids the vanishing gradient problem by this addition.
To investigate the effects of batch normalization, we conducted experiments where the neural architecture is the same as the base architecture, but with SkipInit blocks substituted for residual blocks. We present the important quantities of this ResNet group together with the control group in Table 4.8.
We find that the ResNet group has the same average test accuracy: 71.1%. This validates that our choice of the base architecture does indeed match the performance of ResNets used in practice.

Training speed estimators perform well in predicting generalisation for ResNets The training speed estimators SOTL and TSE-EMA have better abilities to predict generalisation in the ResNet group than in the control group: The value of $\mathcal{K}$ increases from 0.130 to 0.141 for SOTL, and from 0.123 to 0.141 for TSE-EMA; the correlation coefficient with the generalisation error increases from 0.640 to 0.696 for SOTL, and from 0.610 to 0.711 for TSE-EMA. We can see that when a popular neural architecture is used for a visual classification task, the training speed estimators have a consistently strong performance in predicting generalisation.

Control group	Average test accuracy: 71.1 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\left(U;\mu,g\right)$	$\tau\left(U;\mu,\text{TSE-EMA}\right)$
SOTL	**0.130**	**0.325**	**0.640**	**0.949**
SOTL-E_50	0.015	0.050	-0.228	-0.125
TSE-EMA	0.123	0.310	0.610	1.00
PAC-Bayes sharpness	0.071	0.266	0.392	0.642
Value sensitivity	0.037	0.262	0.400	0.644
Initial gradient variance	0.063	0.181	-0.436	-0.737
Final gradient variance	0.010	0.030	0.166	0.071
ResNet group	Average test accuracy: 71.1 %			
μ	$\mathcal{K}(\mu)$	$\bar{\mathcal{K}}(\mu)$	$\tau\left(U;\mu,g\right)$	$\tau\left(U;\mu,\text{TSE-EMA}\right)$
SOTL	**0.141**	0.381	0.696	**0.985**
SOTL-E_50	0.021	0.077	-0.101	-0.240
TSE-EMA	**0.141**	**0.392**	**0.711**	1.00
PAC-Bayes sharpness	0.032	0.333	0.596	0.358
Value sensitivity	0.043	0.266	0.368	0.111
Initial gradient variance	0.013	0.019	-0.101	-0.362
Final gradient variance	0.002	0.069	0.085	-0.111

Table 4.8: The important analytical quantities mentioned in Section 3.2.3 of the complexity measures considered in the three groups of experiments. Bold values indicate the largest value out of the complexity measures in the group by magnitude.

Moreover, when the standard ResNet structure is used, their predictive power is even stronger. Because the standard ResNet architecture uses Batch Normalization, we conclude that the training speed estimators are robust against the presence of Batch Normalization layers in networks.

Sharpness-based measures have weaker correlations with training speed
In the control group, the two sharpness-based measures, PAC-Bayes sharpness and value sensitivity, have relatively high correlation coefficients with the training speed estimator TSE-EMA (0.642 and 0.644 respectively). However, in the ResNet group, they decrease to 0.358 and 0.111 respectively. Therefore we may think that although the predictive power of training speed estimators is stronger in the ResNet group,

it is likely not entirely due to the flatness mechanism.

This section presented the results from the experimental groups. Key takeaways are:

- Training speed estimators, especially TSE-EMA, can consistently predict generalisation well.
- Flatness measures positively correlate with TSE-EMA when logit averaging is applied and when Batch Normalization is present. Its correlation with TSE-EMA is negative but negligible.
- Gradient transfer measures correlate with TSE-EMA when logit averaging is applied, and when Batch Normalization is used.
- The flatness of network weights and generalisation performance both increase when data augmentation is applied. When SGLD is used, the flatness changes up or down randomly, while generalisation performance improves consistently.

4.3 Cross-group analysis

In this section, we perform analysis based on the important quantities observed to discover trends across different groups of experiments.

4.3.1 Normalised Conditional Mutual Information (NCMI)

We present the minimum values of NCMI ($\mathcal{K}$) for each of the six group of experiments in Table 4.9.

From Table 4.9, we can tell that the training-loss-based complexity measures (SOTL, SOTL-E_50, TSE-EMA) have the highest values of $\mathcal{K}(\mu)$ in all groups of experiments except for the data augmentation one. This means that they are mostly likely to have a causal connection with the generalisation error in those groups.

In the MLP group as well as the logit averaging group, the majority of complexity measures have very low values of $\mathcal{K}$ with the generalisation error. The largest $\mathcal{K}$ values in these two groups are 0.123 and 0.047 respectively, both achieved when μ = SOTL-E_50. In these two groups it is more likely that there are some

$\mathcal{K}(\mu)$	Control	MLP	ResNet	SGLD	Data augmentation	Logit averaging
SOTL	**0.130**	0.012	**0.141**	0.044	0.116	0.035
SOTL-E_50	0.015	**0.123**	0.021	0.041	0.032	**0.047**
TSE-EMA	0.123	0.034	**0.141**	**0.171**	0.134	0.029
PACBayes sharpness	0.071	0.024	0.032	0.015	0.051	0.038
Value sensitivity	0.037	0.042	0.043	0.041	**0.146**	0.031
Initial gradient variance	0.063	0.013	0.013	0.057	0.097	0.023
Final gradient variance	0.010	0.091	0.002	0.026	0.092	0.027

Table 4.9: Minimum value of NCMI ($\mathcal{K}(\mu)$) between complexity measures and the generalisation error for various groups of experiments. Best $\mathcal{K}(\mu)$ for each group is in bold.

other factors influencing the generalisation of neural networks, not captured by the selected complexity measures.

When μ = value sensitivity, $\mathcal{K}$ has relatively small values in all groups but the data augmentation group, on which it has a value of 0.146. Training neural networks with data augmentation by transformation makes stronger the connection between generalisation and the sensitivity to input values.

4.3.2 Kendall's rank-correlation coefficient

We present Kendall's rank coefficients between different complexity measures and the generalisation error, for each group of experiments, in Table 4.11.

We can see from the table that the SOTL complexity measure is positively correlated with the generalisation error in all groups of experiments except for the data augmentation one, where the correlation coefficient (-0.097) is very close to zero. The other two optimisation-based complexity measures can be seen as variations of the SOTL, where the SOTL-E_50 focuses more on the later stage of training while TSE-EMA the earlier. Across all groups, the SOTL-E_50 measure is always negatively correlated with generation error and the TSE-EMA is always positively

$\bar{\mathcal{K}}(\mu)$	Control	MLP	ResNet	SGLD	Data augmentation	Logit averaging
SOTL	**0.325**	0.155	0.381	0.140	**0.222**	0.154
SOTL-E_50	0.050	0.153	0.077	0.116	0.140	0.138
TSE-EMA	0.310	0.133	**0.392**	**0.251**	0.182	0.150
PACBayes sharpness	0.266	0.163	0.333	0.048	0.115	**0.156**
Value sensitivity	0.262	**0.189**	0.266	0.060	0.173	0.138
Initial gradient variance	0.181	0.082	0.019	0.088	0.140	0.112
Final gradient variance	0.030	0.113	0.069	0.032	0.125	0.152

Table 4.10: Average value of NCMI ($\bar{\mathcal{K}}(\mu)$) between complexity measures and the generalisation error for various groups of experiments. Best $\bar{\mathcal{K}}(\mu)$ for each group is in bold.

correlated. In fact, they are the only two measures out of the seven monitored to have a consistent correlation with the generalisation error (all negative or all positive). The PACBayes sharpness and the value sensitivity measures are similar in terms of their correlation with generalisation error. When one is positively correlated with generalisation error, so is the other one. Vice versa, their correlation coefficients are also quite similar, across all groups of experiments.

The initial and the final gradient variances have quite different correlations with the generalisation error. The initial gradient variance is negatively correlated with the generalisation error except in the data augmentation group. In that group, the value of $\tau(U; \text{initial gradient variance}, g)$ is 0.073, which is very small and even negligible. The final gradient variance's correlation with the generalisation error is more diverse. The correlation is not strong (the absolute value for the correlation coefficient is smaller than 0.2), apart from the logit averaging group, where the coefficient is 0.469. This is close to the strongest correlation coefficient 0.473 for that group.

For the control group, the proper complexity measures exclude SOTL-E_50 and initial gradient variance, whose correlation coefficients with the generalisation error are -0.228 and -0.436, respectively. The correlation coefficient of SOTL

$\tau(U;\mu,g)$	Control	MLP	ResNet	SGLD	Data augmentation	Logit averaging
SOTL	**0.640**	0.152	0.697	0.236	-0.097	0.414
SOTL-E_50	-0.228	**-0.343**	-0.101	-0.040	**-0.327**	-0.354
TSE-EMA	0.610	0.210	**0.711**	**0.475**	0.046	0.428
PACBayes sharpness	0.392	0.186	0.596	-0.253	0.172	**0.473**
Value sensitivity	0.400	0.220	0.368	-0.154	0.202	0.469
Initial gradient variance	-0.436	-0.111	-0.101	-0.048	0.073	-0.339
Final gradient variance	0.166	-0.149	0.085	-0.176	0.032	0.469

Table 4.11: Kendall's rank correlation coefficient between complexity measures and the generalisation error for various groups of experiments. Largest $\tau(U;\mu,g)$ for each group (by magnitude) is in bold.

(0.640) is the largest in absolute value out of all the measures, while that of TSE-EMA (0.610) is a close second.

For the SGLD group, only SOTL and TSE-EMA remain proper complexity measures and have non-negative correlation coefficients with the generalisation error (0.236 and 0.475 respectively).

For the MLP group and the data augmentation group, the strongest positive correlation coefficients are 0.220 and 0.202, respectively, which are relatively small. For other groups, the strongest positive correlation coefficients are all larger than 0.45.

Initial gradient variance behaves surprisingly We see that in most cases, the initial gradient variance is negatively correlated with the generalisation error, which suggests that the more varied the initial gradients are, the better the generalisation. This is surprising, as we associate a small gradient variance with gradient vectors with aligned directions, and thus the gradient update on one mini-batch to transfer better to other mini-batches. The good gradient transfer means that the training loss is decreasing faster, and faster training speed. Since we discovered that the

training speed is a good predictor of generalisation error, this association between small gradient variance and fast training speed seems to contradict the consistent negative correlation between gradient variance and generalisation error.
A possible reason for this phenomenon is that different learning rates might cause the connection between initial gradient variance and generalisation to be different, as we explained in Section 4.1.

SGLD improves generalisation without altering flatness When the SGLD optimiser is used instead of the SGD optimiser, it artificially increases the variance in the stochastic gradient estimator. When we compare pairs of experiments with the same hyperparameters except for the optimiser used, we found that in the 45 pairs of experiments, 21 experiments have flatter minima with SGLD, and 24 have sharper minima with SGLD. All experiments have better generalisation when SGLD is applied. This means that applying an intervention in the form of using SGLD does not alter the flatness of the optima in a statistically significant way. The improvement in generalisation brought by this intervention cannot be explained by the flatness hypothesis. Moreover, in experiments where SGLD is used, PACBayes flatness gives very low CMI with generalisation error and the correlation with generalisation error is negligible. This means that not using SGLD is a necessary condition for the possible mechanism of flatness inducing generalisation.

5 Conclusion and Discussion

Contents

5.1 Conclusion

In this dissertation, we set out to investigate the mechanism through which training speed can be connected with generalisation. We started by hypothesising the quantities in classical theories that are potentially connected to training speed and can explain generalisation. Next, we conducted experiments. First to confirm the connection between training speed and generalisation, and second to find out the necessary conditions under which the hypotheses can hold.

More concretely, we hypothesised that the connection between training speed and generalisation can be explained via one of these two quantities:

- Network flatness;
- Gradient transfer.

To validate the hypotheses, we conducted experiments for a large family of neural networks. The central question our experiments try to answer is: under what conditions might the two hypotheses be causing the connection between training speed and generalisation. This question is decomposed into two sub-questions: (1) under what conditions does the connection exist? (2) assuming that the connection exists, which of the two hypotheses can potentially cause the connection?
In our experiments, we chose two quantities (PAC-Bayes sharpness and value sensitivity) to reflect the network flatness and two quantities (initial and final gradient variances) to reflect the gradient transfer quantitatively. For training speed, we considered three training-loss-based entities (SOTL, SOTL-E_50, and TSE-EMA) to reflect the training speed in general, as well as the training speed in the late and early stages, respectively. Since it is difficult to apply intervention to the flatness-related or the gradient-transfer-related quantities directly and therefore to reason about the causal connections that might exist, we instead varied the hyperparameters of the networks in order to vary the quantities. We experimented with architectures such as SkipInit (section 4.1), ResNet (subsection 4.2.4), and MLP (subsection 4.2.3); optimisers such as SGD (section 4.1) and SGLD (subsection 4.2.2); data augmentation techniques such as straighforward transformation and logit averaging (subsection 4.2.1). By observing the correlation between training speed and generalisation, between quantities indicative of the validity of hypotheses and training speed, we reached the following conclusions in Chapter 4:

- In line with the work of Lyle et al. [2020] and Ru et al. [2020], we showed that training speed exhibits a connection with the generalisation performance of deep neural networks. We extend the work by showing that the connection exists when the neural architecture is SkipInit or ResNet, the optimiser is SGD or SGLD, and logit averaging is used for data augmentation or no data augmentation at all. It does not exist when the MLP architecture is used for the CIFAR10 dataset.

- Network flatness might explain this connection when the optimiser used is SGD, the neural architecture is SkipInit or ResNet, and logit averaging is used for data augmentation or no data augmentation at all. It cannot explain the connection when SGLD is used;
- Gradient transfer might explain this connection when the optimiser used is SGD, the neural architecture is SkipInit, and logit averaging is used for data augmentation or no data augmentation at all. It is less likely that it will be able to explain the connection when SGLD is used as the optimiser or ResNet as the architecture.

Hence we gained the confirmation of the existence of a connection between training speed and generalisation in deep neural networks, as well as the conditions under which the connection might be explainable by at least one of our two hypotheses.

5.2 Discussion and Future Work

The conclusions reached advance our understanding of the mechanism behind the training-speed-generalisation in two ways:

- Firstly, we know in which cases the connection does not exist or the connection cannot be explained by one of the two hypotheses. The neural architecture being suitable for the tasks considered is necessary for the connection to exist, as demonstrated in the MLP group. The application of SGLD renders the two hypotheses invalid, and not using Batch Normalization (as in the ResNet group) is crucial for the gradient transfer hypothesis to remain plausible. Figuring out why these interventions have the effects could be a direction for future work.
- Secondly, we gathered some conditions under which the connection exists and at least one of the hypotheses can potentially explain it. Experimenting further under these conditions is a direction for future work in order to understand the true mechanism of the connection.

Additionally, while doing a large family of experiments, we found some interesting insights which raise questions for potential future research projects.

- In Section 4.1, we found that the initial gradient variance measure behaves differently when different learning rates are applied. Namely, its correlation coefficient with generalisation error monotonically decreases from 0.167 to -0.611 when the learning rate increases from 0.001 to 0.01. Its correlation coefficient with TSE-EMA also monotonically decreases from -0.333 to -0.833 when the learning rate is increased from 0.001 to 0.01. Why does the intervention of increasing the learning rate change the correlation between initial gradient variance and the two entities? How do the training regimes, determined by learning rates, affect how gradient transfer is related to generalisation?

- We found in Subsection 4.2.1 that data augmentation improves the flatness as well as generalisation of all networks considered. Is this correlation actually causation, i.e. the intervention of data augmentation improves network flatness, and hence better flatness causes better generalisation? Or is this correlation spurious, and there are other factors causing the better generalisation performance?

- We found in Subsection 4.2.2 that the intervention of increased gradient variance causes better generalisation. However, it changes the sharpness of networks to higher or lower almost randomly. It also makes the sharpness-based measures much less correlated with training speed, and even negatively correlated with generalisation error. How does the change in gradient variance cause such changes? How does reducing gradient variance (e.g. by using larger batch sizes) change the connection between sharpness and generalisation and training speed?

Bibliography

Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *J. Mach. Learn. Res.*, 3:463–482, 2002. URL http://jmlr.org/papers/v3/bartlett02a.html.

Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6240–6249, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/b22b257ad0519d4500539da3c8bcf4dd-Abstract.html.

Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, 2002. URL http://jmlr.org/papers/v2/bousquet02a.html.

Wray L. Buntine and Andreas S. Weigend. Bayesian back-propagation. *Complex Syst.*, 5 (6), 1991. URL http://www.complex-systems.com/abstracts/v05_i06_a04.html.

Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=HyWrIgW0W.

Tri Dao, Albert Gu, Alexander Ratner, Virginia Smith, Chris De Sa, and Christopher Ré. A kernel theory of modern data augmentation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 1528–1537. PMLR, 2019. URL http://proceedings.mlr.press/v97/dao19b.html.

Soham De and Samuel L. Smith. Batch normalization biases residual blocks towards the identity function in deep networks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6b738eca0e6792ba8a9cbcba6c1881d-Abstract.html.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW,*

Australia, 6-11 August 2017, volume 70 of *Proceedings of Machine Learning Research*, pages 1019–1028. PMLR, 2017. URL http://proceedings.mlr.press/v70/dinh17b.html.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In Gal Elidan, Kristian Kersting, and Alexander T. Ihler, editors, *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*. AUAI Press, 2017. URL http://auai.org/uai2017/proceedings/papers/173.pdf.

Gintare Karolina Dziugaite, Alexandre Drouin, Brady Neal, Nitarshan Rajkumar, Ethan Caballero, Linbo Wang, Ioannis Mitliagkas, and Daniel M. Roy. In search of robust measures of generalization. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/86d7c8a08b4aaa1bc7c599473f5dddda-Abstract.html.

Stanislav Fort, Pawel Krzysztof Nowak, and Srini Narayanan. Stiffness: A new perspective on generalization in neural networks. *CoRR*, abs/1901.09491, 2019. URL http://arxiv.org/abs/1901.09491.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

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

Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *CoRR*, abs/1902.09574, 2019. URL http://arxiv.org/abs/1902.09574.

Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural network. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1135–1143, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/ae0eb3eed39d2bcef4622b2499a05fe6-Abstract.html.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and*

Conference Proceedings, pages 1225–1234. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/hardt16.html.

Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In Stephen Jose Hanson, Jack D. Cowan, and C. Lee Giles, editors, *Advances in Neural Information Processing Systems 5, [NIPS Conference, Denver, Colorado, USA, November 30 - December 3, 1992]*, pages 164–171. Morgan Kaufmann, 1992. URL http://papers.nips.cc/paper/647-second-order-derivatives-for-network-pruning-optimal-brain-surgeon.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Comput.*, 9(1):1–42, 1997. doi: 10.1162/neco.1997.9.1.1. URL https://doi.org/10.1162/neco.1997.9.1.1.

W. Ronny Huang, Zeyad Emam, Micah Goldblum, Liam Fowl, Justin K. Terry, Furong Huang, and Tom Goldstein. Understanding generalization through visualizations. *CoRR*, abs/1906.03291, 2019. URL http://arxiv.org/abs/1906.03291.

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

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8580–8589, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/5a4be1fa34e62bb8a6ec6b91d2462f5a-Abstract.html.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SJgIPJBFvH.

M. Kendall. A new measure of rank correlation. *Biometrika*, 30:81–93, 1938.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In David S. Touretzky, editor, *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]*, pages 598–605. Morgan Kaufmann, 1989. URL http://papers.nips.cc/paper/250-optimal-brain-damage.

Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade - Second Edition*, volume 7700 of *Lecture Notes in Computer Science*, pages 9–48. Springer, 2012. doi: 10.1007/978-3-642-35289-8_3. URL https://doi.org/10.1007/978-3-642-35289-8_3.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rJqFGTslg.

Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. ISBN 978-3-030-11297-4. doi: 10.1007/978-3-030-11298-1. URL https://doi.org/10.1007/978-3-030-11298-1.

Tengyuan Liang, Tomaso A. Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pages 888–896. PMLR, 2019. URL http://proceedings.mlr.press/v89/liang19a.html.

Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJlnB3C5Ym.

Clare Lyle, Lisa Schut, Robin Ru, Yarin Gal, and Mark van der Wilk. A bayesian perspective on training speed and model selection. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/75a7c30fc0063c4952d7eb044a3c0897-Abstract.html.

David A. McAllester. Pac-bayesian model averaging. In Shai Ben-David and Philip M. Long, editors, *Proceedings of the Twelfth Annual Conference on Computational*

Learning Theory, COLT 1999, Santa Cruz, CA, USA, July 7-9, 1999, pages 164–170. ACM, 1999. doi: 10.1145/307400.307435. URL https://doi.org/10.1145/307400.307435.

Sayan Mukherjee, Partha Niyogi, Tomaso A. Poggio, and Ryan M. Rifkin. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Adv. Comput. Math.*, 25(1-3):161–193, 2006. doi: 10.1007/s10444-004-7634-z. URL https://doi.org/10.1007/s10444-004-7634-z.

Seth Nabarro, Stoil Ganev, Adrià Garriga-Alonso, Vincent Fortuin, Mark van der Wilk, and Laurence Aitchison. Data augmentation in bayesian neural networks and the cold posterior effect. *CoRR*, abs/2106.05586, 2021. URL https://arxiv.org/abs/2106.05586.

Vaishnavh Nagarajan and J. Zico Kolter. Generalization in deep networks: The role of distance from initialization. *CoRR*, abs/1901.01672, 2019. URL http://arxiv.org/abs/1901.01672.

Behnam Neyshabur, Ruslan Salakhutdinov, and Nathan Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2422–2430, 2015a. URL https://proceedings.neurips.cc/paper/2015/hash/eaa32c96f620053cf442ad32258076b9-Abstract.html.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In Peter Grünwald, Elad Hazan, and Satyen Kale, editors, *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, volume 40 of *JMLR Workshop and Conference Proceedings*, pages 1376–1401. JMLR.org, 2015b. URL http://proceedings.mlr.press/v40/Neyshabur15.html.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5947–5956, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/10ce03a1ed01077e3e289f3e53c72813-Abstract.html.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*

2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 8024–8035, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Tomaso Poggio, Ryan Rifkin, Sayan Mukherjee, and Partha Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419–422, 2004.

Carl Edward Rasmussen and Zoubin Ghahramani. Occam's razor. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 294–300. MIT Press, 2000. URL https://proceedings.neurips.cc/paper/2000/hash/0950ca92a4dcf426067cfd2246bb5ff3-Abstract.html.

Jorma Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of statistics*, pages 416–431, 1983.

Binxin Ru, Clare Lyle, Lisa Schut, Mark van der Wilk, and Yarin Gal. Revisiting the train loss: an efficient performance estimator for neural architecture search. *CoRR*, abs/2006.04492, 2020. URL https://arxiv.org/abs/2006.04492.

Karthik Abinav Sankararaman, Soham De, Zheng Xu, W. Ronny Huang, and Tom Goldstein. The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 8469–8479. PMLR, 2020. URL http://proceedings.mlr.press/v119/sankararaman20a.html.

Samuel L. Smith and Quoc V. Le. A bayesian perspective on generalization and stochastic gradient descent. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=BJij4yg0Z.

Samuel L. Smith, Benoit Dherin, David G. T. Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. *CoRR*, abs/2101.12176, 2021. URL https://arxiv.org/abs/2101.12176.

Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. *CoRR*, abs/2105.01601, 2021. URL https://arxiv.org/abs/2105.01601.

Mark van der Wilk, Matthias Bauer, S. T. John, and James Hensman. Learning invariances using the marginal likelihood. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9960–9970, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/d465f14a648b3d0a1faa6f447e526c60-Abstract.html.

Vladimir Vapnik. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015.

Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In Piero P. Bonissone, Max Henrion, Laveen N. Kanal, and John F. Lemmer, editors, *UAI '90: Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence, MIT, Cambridge, MA, USA, July 27-29, 1990*, pages 255–270. Elsevier, 1990. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1918&proceeding_id=1006.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 681–688. Omnipress, 2011. URL https://icml.cc/2011/papers/398_icmlpaper.pdf.

Ashia C. Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4148–4158, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/81b3833e2504647f9d794f7d7b9bf341-Abstract.html.

Sho Yaida. Fluctuation-dissipation relations for stochastic gradient descent. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=SkNksoRctQ.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Appendices

A

All Experimental Results

In this appendix we present the raw data for all of our experiments in the six different groups.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
1	SkipInit	SGD	False	4	16	0.01	75.66	0.242	0.019	0.019	0.64	0.038	0.001	1316.887	161.272
2	SkipInit	SGD	False	4	16	0.006	73.82	0.26	0.021	0.021	0.741	0.044	0.003	1794.789	835.613
3	SkipInit	SGD	False	4	16	0.003	72.46	0.275	0.025	0.025	0.933	0.05	0.004	689.783	38.25
4	SkipInit	SGD	False	4	16	0.0016	71.61	0.284	0.03	0.028	1.152	0.059	0.006	212.564	636.701
5	SkipInit	SGD	False	4	16	0.001	70.35	0.296	0.035	0.024	1.335	0.064	0.007	118.158	563.588
6	SkipInit	SGD	False	4	12	0.01	75.02	0.248	0.021	0.021	0.718	0.036	0.001	951.004	745.224
7	SkipInit	SGD	False	4	12	0.006	73.95	0.259	0.023	0.023	0.837	0.041	0.002	888.288	593.411
8	SkipInit	SGD	False	4	12	0.003	73.55	0.264	0.027	0.025	1.046	0.048	0.003	216.551	206.003
9	SkipInit	SGD	False	4	12	0.0016	71.55	0.284	0.033	0.023	1.272	0.057	0.005	86.209	226.169
10	SkipInit	SGD	False	4	12	0.001	70.93	0.29	0.039	0.021	1.455	0.062	0.006	64.991	1016.057
11	SkipInit	SGD	False	4	8	0.01	73.55	0.263	0.023	0.017	0.783	0.034	0.001	893.753	379.854
12	SkipInit	SGD	False	4	8	0.006	72.1	0.277	0.025	0.016	0.905	0.039	0.002	769.444	872.506
13	SkipInit	SGD	False	4	8	0.003	70.62	0.293	0.03	0.02	1.117	0.048	0.003	140.515	1214.869
14	SkipInit	SGD	False	4	8	0.0016	70.03	0.299	0.036	0.02	1.345	0.055	0.005	52.602	589.72
15	SkipInit	SGD	False	4	8	0.001	69.59	0.304	0.043	0.018	1.521	0.061	0.006	40.827	648.236
16	SkipInit	SGD	False	3	16	0.01	74.11	0.257	0.019	0.019	0.669	0.037	0.001	1177.997	603.683
17	SkipInit	SGD	False	3	16	0.006	73.04	0.269	0.021	0.021	0.785	0.042	0.002	985.091	800.608
18	SkipInit	SGD	False	3	16	0.003	72.18	0.278	0.026	0.026	0.995	0.05	0.004	235.234	94.948
19	SkipInit	SGD	False	3	16	0.0016	71.86	0.281	0.032	0.025	1.23	0.057	0.005	73.312	296.852
20	SkipInit	SGD	False	3	16	0.001	71.06	0.289	0.037	0.022	1.419	0.064	0.006	53.793	532.611
21	SkipInit	SGD	False	3	12	0.01	72.84	0.27	0.022	0.022	0.778	0.036	0.001	927.47	574.569
22	SkipInit	SGD	False	3	12	0.006	72.53	0.274	0.024	0.024	0.898	0.041	0.002	1269.525	252.148
23	SkipInit	SGD	False	3	12	0.003	70.3	0.297	0.029	0.025	1.103	0.048	0.003	964.123	269.917
24	SkipInit	SGD	False	3	12	0.0016	69.49	0.305	0.035	0.022	1.319	0.057	0.005	278.932	667.829
25	SkipInit	SGD	False	3	12	0.001	68.85	0.311	0.042	0.019	1.481	0.062	0.006	145.337	617.933

Table A.1: Results from the control group, experiments 1-25.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E__50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
26	SkipInit	SGD	False	3	8	0.01	70.5	0.293	0.024	0.015	0.828	0.035	0.001	888.64	672.62
27	SkipInit	SGD	False	3	8	0.006	69.17	0.306	0.026	0.016	0.958	0.039	0.002	810.265	1087.376
28	SkipInit	SGD	False	3	8	0.003	68.84	0.31	0.031	0.02	1.164	0.048	0.003	336.903	2170.696
29	SkipInit	SGD	False	3	8	0.0016	68.65	0.313	0.038	0.018	1.374	0.055	0.005	160.051	337.458
30	SkipInit	SGD	False	3	8	0.001	67.85	0.321	0.045	0.016	1.534	0.06	0.006	114.792	735.146
31	SkipInit	SGD	False	2	16	0.01	72.13	0.277	0.021	0.021	0.736	0.036	0.001	995.129	592.616
32	SkipInit	SGD	False	2	16	0.006	71.71	0.282	0.024	0.024	0.871	0.041	0.002	1154.385	294.58
33	SkipInit	SGD	False	2	16	0.003	70.57	0.294	0.029	0.024	1.1	0.048	0.003	634.374	380.623
34	SkipInit	SGD	False	2	16	0.0016	69.76	0.302	0.037	0.018	1.329	0.055	0.004	201.846	270.576
35	SkipInit	SGD	False	2	16	0.001	69.72	0.303	0.044	0.016	1.493	0.06	0.006	117.209	446.407
36	SkipInit	SGD	False	2	12	0.01	70.83	0.289	0.022	0.022	0.807	0.036	0.001	805.223	464.932
37	SkipInit	SGD	False	2	12	0.006	70.12	0.297	0.025	0.023	0.936	0.041	0.002	736.384	570.572
38	SkipInit	SGD	False	2	12	0.003	69.94	0.3	0.03	0.022	1.161	0.048	0.003	150.749	208.885
39	SkipInit	SGD	False	2	12	0.0016	69.08	0.309	0.037	0.019	1.39	0.057	0.004	82.52	627.778
40	SkipInit	SGD	False	2	12	0.001	69.0	0.31	0.044	0.017	1.56	0.062	0.006	73.37	914.166
41	SkipInit	SGD	False	2	8	0.001	69.22	0.308	0.048	0.013	1.578	0.059	0.005	37.294	356.894
42	SkipInit	SGD	False	2	8	0.01	71.33	0.284	0.025	0.013	0.846	0.034	0.001	846.562	269.862
43	SkipInit	SGD	False	2	8	0.0016	69.98	0.3	0.04	0.015	1.405	0.052	0.004	40.619	463.527
44	SkipInit	SGD	False	2	8	0.006	71.02	0.289	0.027	0.016	0.97	0.038	0.002	487.749	227.9
45	SkipInit	SGD	False	2	8	0.003	70.74	0.291	0.033	0.016	1.186	0.046	0.003	71.103	1075.412

Table A.2: Results from the control group, experiments 26-45.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
1	SkipInit	SGD	False	4	16	0.01	87.53	0.121	0.029	0.005	0.886	0.029	0.001	1004.449	562.772
2	SkipInit	SGD	False	4	16	0.006	86.65	0.13	0.032	0.005	0.998	0.033	0.001	1050.062	585.328
3	SkipInit	SGD	False	4	16	0.003	85.51	0.142	0.038	0.005	1.185	0.038	0.002	566.069	3717.706
4	SkipInit	SGD	False	4	16	0.0016	84.91	0.147	0.047	0.005	1.377	0.042	0.003	198.881	3930.668
5	SkipInit	SGD	False	4	16	0.001	83.68	0.157	0.055	0.006	1.524	0.048	0.004	111.244	12824.123
6	SkipInit	SGD	False	4	12	0.01	88.06	0.116	0.032	0.005	0.931	0.028	0.001	765.752	548.83
7	SkipInit	SGD	False	4	12	0.006	87.71	0.12	0.036	0.005	1.056	0.032	0.001	444.279	1020.348
8	SkipInit	SGD	False	4	12	0.003	86.93	0.128	0.043	0.005	1.252	0.036	0.002	175.695	3178.628
9	SkipInit	SGD	False	4	12	0.0016	85.07	0.143	0.051	0.006	1.448	0.042	0.002	79.185	8252.413
10	SkipInit	SGD	False	4	12	0.001	84.57	0.134	0.059	0.009	1.602	0.048	0.003	62.285	15913.664
11	SkipInit	SGD	False	4	8	0.01	86.86	0.119	0.04	0.006	1.008	0.027	0.001	623.421	3075.091
12	SkipInit	SGD	False	4	8	0.006	86.18	0.127	0.044	0.007	1.13	0.03	0.001	441.306	2853.679
13	SkipInit	SGD	False	4	8	0.003	85.78	0.124	0.051	0.008	1.324	0.037	0.002	96.507	7464.115
14	SkipInit	SGD	False	4	8	0.0016	83.65	0.124	0.06	0.011	1.52	0.041	0.003	48.161	14526.586
15	SkipInit	SGD	False	4	8	0.001	83.02	0.1	0.068	0.015	1.669	0.046	0.003	40.14	19010.252
16	SkipInit	SGD	False	3	16	0.01	87.04	0.126	0.03	0.005	0.896	0.029	0.001	898.607	636.888
17	SkipInit	SGD	False	3	16	0.006	86.6	0.131	0.033	0.005	1.017	0.032	0.001	623.678	1027.656
18	SkipInit	SGD	False	3	16	0.003	85.94	0.137	0.04	0.005	1.21	0.037	0.002	175.735	3525.198
19	SkipInit	SGD	False	3	16	0.0016	84.68	0.148	0.048	0.005	1.405	0.041	0.003	67.86	3764.551
20	SkipInit	SGD	False	3	16	0.001	81.82	0.151	0.056	0.007	1.561	0.047	0.004	53.439	37477.188
21	SkipInit	SGD	False	3	12	0.01	87.46	0.123	0.035	0.005	0.992	0.027	0.001	796.366	337.804
22	SkipInit	SGD	False	3	12	0.006	86.79	0.129	0.039	0.004	1.116	0.03	0.001	918.952	1359.419
23	SkipInit	SGD	False	3	12	0.003	85.69	0.134	0.046	0.006	1.303	0.036	0.002	752.197	5096.753
24	SkipInit	SGD	False	3	12	0.0016	83.79	0.142	0.056	0.008	1.486	0.042	0.003	230.436	14731.365
25	SkipInit	SGD	False	3	12	0.001	83.04	0.134	0.064	0.011	1.622	0.048	0.004	139.438	23582.111

Table A.3: Results from the data augmentation group, experiments 1-25.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E__50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
26	SkipInit	SGD	False	3	8	0.01	85.12	0.133	0.043	0.008	1.065	0.027	0.001	666.324	2192.937
27	SkipInit	SGD	False	3	8	0.006	83.01	0.124	0.047	0.008	1.18	0.032	0.001	546.28	11215.236
28	SkipInit	SGD	False	3	8	0.003	83.72	0.132	0.054	0.01	1.355	0.038	0.002	303.517	6423.059
29	SkipInit	SGD	False	3	8	0.0016	83.6	0.117	0.063	0.013	1.533	0.041	0.002	157.767	10325.62
30	SkipInit	SGD	False	3	8	0.001	82.65	0.096	0.07	0.016	1.668	0.046	0.003	118.848	14417.588
31	SkipInit	SGD	False	2	16	0.01	85.99	0.137	0.037	0.005	1.003	0.026	0.001	773.069	167.941
32	SkipInit	SGD	False	2	16	0.006	85.4	0.137	0.041	0.005	1.126	0.03	0.001	896.996	3578.069
33	SkipInit	SGD	False	2	16	0.003	84.66	0.143	0.048	0.006	1.306	0.036	0.002	467.966	6111.32
34	SkipInit	SGD	False	2	16	0.0016	83.99	0.145	0.057	0.009	1.487	0.042	0.003	176.733	8032.877
35	SkipInit	SGD	False	2	16	0.001	83.06	0.131	0.066	0.012	1.623	0.048	0.004	113.223	17420.064
36	SkipInit	SGD	False	2	12	0.01	86.13	0.122	0.039	0.006	1.02	0.027	0.001	644.35	3333.496
37	SkipInit	SGD	False	2	12	0.006	86.33	0.126	0.042	0.006	1.146	0.03	0.001	383.829	2900.578
38	SkipInit	SGD	False	2	12	0.003	85.04	0.133	0.05	0.008	1.344	0.036	0.002	113.418	5584.465
39	SkipInit	SGD	False	2	12	0.0016	84.97	0.116	0.059	0.011	1.538	0.041	0.002	74.543	12081.891
40	SkipInit	SGD	False	2	12	0.001	82.64	0.112	0.067	0.014	1.685	0.046	0.003	68.323	17036.516
41	SkipInit	SGD	False	2	8	0.0016	83.65	0.08	0.066	0.016	1.533	0.041	0.002	39.616	10116.684
42	SkipInit	SGD	False	2	8	0.003	83.76	0.1	0.058	0.013	1.34	0.036	0.002	66.878	7099.093
43	SkipInit	SGD	False	2	8	0.006	84.1	0.108	0.051	0.011	1.159	0.03	0.001	343.823	4612.646
44	SkipInit	SGD	False	2	8	0.01	85.57	0.111	0.048	0.011	1.045	0.027	0.001	537.298	1837.349
45	SkipInit	SGD	False	2	8	0.001	82.34	0.066	0.072	0.018	1.685	0.044	0.003	37.377	13370.808

Table A.4: Results from the data augmentation group, experiments 26-45.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
1	SkipInit	SGD	False	4	16	0.01	84.23	0.089	0.022	0.022	0.814	0.034	0.001	1085.84	13412.898
2	SkipInit	SGD	False	4	16	0.006	83.21	0.079	0.025	0.025	0.928	0.038	0.002	1199.109	16867.408
3	SkipInit	SGD	False	4	16	0.003	83.44	0.077	0.03	0.026	1.11	0.041	0.002	608.642	24428.49
4	SkipInit	SGD	False	4	16	0.0016	82.5	0.091	0.037	0.022	1.305	0.048	0.003	206.789	43121.77
5	SkipInit	SGD	False	4	16	0.001	81.77	0.094	0.043	0.02	1.461	0.052	0.004	113.505	62817.395
6	SkipInit	SGD	False	4	12	0.01	84.36	0.068	0.023	0.023	0.862	0.034	0.001	801.651	9166.487
7	SkipInit	SGD	False	4	12	0.006	83.88	0.091	0.027	0.026	0.976	0.038	0.001	544.969	14037.579
8	SkipInit	SGD	False	4	12	0.003	83.3	0.083	0.032	0.022	1.168	0.041	0.002	197.858	20755.742
9	SkipInit	SGD	False	4	12	0.0016	82.68	0.096	0.04	0.02	1.37	0.046	0.003	82.122	38683.246
10	SkipInit	SGD	False	4	12	0.001	82.0	0.1	0.046	0.019	1.532	0.052	0.003	62.793	49593.578
11	SkipInit	SGD	False	4	8	0.01	83.22	0.083	0.026	0.026	0.916	0.032	0.001	681.233	10167.109
12	SkipInit	SGD	False	4	8	0.006	82.87	0.087	0.029	0.022	1.034	0.035	0.001	516.675	16815.535
13	SkipInit	SGD	False	4	8	0.003	82.93	0.094	0.036	0.019	1.235	0.039	0.002	112.028	26054.545
14	SkipInit	SGD	False	4	8	0.0016	81.77	0.099	0.045	0.019	1.442	0.046	0.003	50.429	34929.027
15	SkipInit	SGD	False	4	8	0.001	81.77	0.098	0.053	0.018	1.603	0.05	0.003	40.946	55369.398
16	SkipInit	SGD	False	3	16	0.01	84.19	0.081	0.022	0.022	0.833	0.034	0.001	993.831	9729.659
17	SkipInit	SGD	False	3	16	0.006	82.88	0.085	0.025	0.025	0.946	0.038	0.002	716.718	12838.3
18	SkipInit	SGD	False	3	16	0.003	82.93	0.091	0.031	0.024	1.137	0.041	0.002	192.562	21691.758
19	SkipInit	SGD	False	3	16	0.0016	81.77	0.103	0.038	0.02	1.339	0.047	0.003	69.826	37539.246
20	SkipInit	SGD	False	3	16	0.001	82.04	0.094	0.044	0.02	1.502	0.052	0.004	54.306	41634.863
21	SkipInit	SGD	False	3	12	0.01	83.79	0.075	0.025	0.025	0.915	0.033	0.001	866.297	10040.325
22	SkipInit	SGD	False	3	12	0.006	83.3	0.088	0.028	0.025	1.032	0.037	0.001	1045.852	15342.189
23	SkipInit	SGD	False	3	12	0.003	83.27	0.086	0.035	0.022	1.223	0.041	0.002	831.276	20266.102
24	SkipInit	SGD	False	3	12	0.0016	81.75	0.097	0.043	0.019	1.411	0.044	0.003	245.674	50841.973
25	SkipInit	SGD	False	3	12	0.001	80.99	0.103	0.051	0.019	1.557	0.05	0.003	145.32	56331.949

Table A.5: Results from the logit averaging group, experiments 1-25.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E__50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
26	SkipInit	SGD	False	3	8	0.01	82.12	0.093	0.028	0.023	0.97	0.031	0.001	728.705	9254.092
27	SkipInit	SGD	False	3	8	0.006	82.37	0.088	0.031	0.021	1.084	0.036	0.001	608.63	14533.864
28	SkipInit	SGD	False	3	8	0.003	82.83	0.09	0.038	0.019	1.268	0.039	0.002	328.368	24039.861
29	SkipInit	SGD	False	3	8	0.0016	81.54	0.099	0.047	0.018	1.457	0.044	0.003	161.924	33438.547
30	SkipInit	SGD	False	3	8	0.001	79.51	0.082	0.055	0.019	1.603	0.047	0.003	119.091	49526.477
31	SkipInit	SGD	False	2	16	0.01	82.41	0.09	0.026	0.026	0.921	0.033	0.001	883.262	10012.531
32	SkipInit	SGD	False	2	16	0.006	82.1	0.095	0.03	0.023	1.043	0.036	0.002	1068.329	15121.888
33	SkipInit	SGD	False	2	16	0.003	82.07	0.093	0.036	0.022	1.233	0.041	0.002	524.701	18590.494
34	SkipInit	SGD	False	2	16	0.0016	82.03	0.093	0.044	0.02	1.421	0.044	0.003	188.521	29094.523
35	SkipInit	SGD	False	2	16	0.001	81.21	0.105	0.052	0.018	1.564	0.048	0.003	117.861	47360.832
36	SkipInit	SGD	False	2	12	0.01	83.68	0.077	0.026	0.026	0.928	0.032	0.001	760.272	8901.662
37	SkipInit	SGD	False	2	12	0.006	83.07	0.081	0.029	0.023	1.054	0.035	0.002	439.384	16090.36
38	SkipInit	SGD	False	2	12	0.0016	82.24	0.091	0.044	0.019	1.467	0.044	0.002	77.082	35222.461
39	SkipInit	SGD	False	2	12	0.001	81.99	0.093	0.052	0.018	1.625	0.048	0.003	69.897	47277.969
40	SkipInit	SGD	False	2	12	0.003	82.58	0.09	0.036	0.021	1.26	0.039	0.002	122.546	21196.9
41	SkipInit	SGD	False	2	8	0.0016	82.28	0.089	0.05	0.017	1.461	0.041	0.002	41.048	28443.092
42	SkipInit	SGD	False	2	8	0.01	84.07	0.073	0.028	0.021	0.951	0.03	0.001	607.751	8427.526
43	SkipInit	SGD	False	2	8	0.006	83.11	0.08	0.032	0.02	1.068	0.033	0.001	425.019	10668.374
44	SkipInit	SGD	False	2	8	0.003	83.61	0.083	0.04	0.018	1.259	0.038	0.002	76.49	21302.078
45	SkipInit	SGD	False	2	8	0.001	80.84	0.059	0.055	0.02	1.624	0.045	0.003	38.259	32382.34

Table A.6: Results from the logit averaging group, experiments 26-45.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
1	SkipInit	SGD	False	4	16	0.01	73.53	0.192	0.061	0.022	0.515	0.066	0.013	2879.992	50610.129
2	SkipInit	SGD	False	4	16	0.006	83.14	0.165	0.017	0.01	0.358	0.064	0.009	1124.76	10285.813
3	SkipInit	SGD	False	4	16	0.003	82.52	0.172	0.013	0.013	0.322	0.057	0.011	627.478	3752.083
4	SkipInit	SGD	False	4	16	0.0016	81.21	0.185	0.013	0.013	0.348	0.057	0.007	503.472	653.013
5	SkipInit	SGD	False	4	16	0.001	80.52	0.192	0.014	0.014	0.389	0.034	0.001	455.562	337.467
6	SkipInit	SGD	False	4	12	0.01	68.35	0.103	0.054	0.026	0.575	0.053	0.003	410.684	48147.328
7	SkipInit	SGD	False	4	12	0.006	81.73	0.18	0.019	0.01	0.416	0.067	0.006	432.272	4055.483
8	SkipInit	SGD	False	4	12	0.003	83.29	0.164	0.014	0.014	0.374	0.057	0.009	381.169	2784.668
9	SkipInit	SGD	False	4	12	0.0016	81.92	0.177	0.014	0.014	0.393	0.044	0.002	388.781	2869.036
10	SkipInit	SGD	False	4	12	0.001	80.97	0.188	0.015	0.015	0.437	0.029	0.001	428.164	161.07
11	SkipInit	SGD	False	4	8	0.01	73.26	0.159	0.048	0.016	0.565	0.071	0.006	620.816	144130.266
12	SkipInit	SGD	False	4	8	0.006	81.28	0.184	0.022	0.01	0.419	0.061	0.008	301.724	6698.788
13	SkipInit	SGD	False	4	8	0.003	80.89	0.188	0.016	0.008	0.396	0.057	0.014	356.538	4122.349
14	SkipInit	SGD	False	4	8	0.0016	78.81	0.21	0.016	0.009	0.457	0.057	0.014	424.527	3670.078
15	SkipInit	SGD	False	4	8	0.001	78.4	0.213	0.018	0.011	0.526	0.041	0.015	497.13	2737.706
16	SkipInit	SGD	False	3	16	0.01	73.95	0.204	0.053	0.019	0.558	0.057	0.006	1079.789	147831.016
17	SkipInit	SGD	False	3	16	0.006	82.78	0.169	0.017	0.01	0.381	0.062	0.012	606.764	7736.417
18	SkipInit	SGD	False	3	16	0.003	82.89	0.168	0.013	0.013	0.32	0.057	0.006	311.296	5036.242
19	SkipInit	SGD	False	3	16	0.0016	82.07	0.177	0.013	0.013	0.346	0.041	0.001	288.679	347.001
20	SkipInit	SGD	False	3	16	0.001	80.52	0.192	0.014	0.014	0.39	0.036	0.0	333.385	184.505
21	SkipInit	SGD	False	3	12	0.01	70.99	0.217	0.05	0.018	0.589	0.052	0.004	1570.756	186125.891
22	SkipInit	SGD	False	3	12	0.006	80.77	0.189	0.021	0.01	0.452	0.061	0.01	479.824	6199.931
23	SkipInit	SGD	False	3	12	0.003	81.46	0.182	0.015	0.01	0.385	0.051	0.011	305.281	4534.629
24	SkipInit	SGD	False	3	12	0.0016	80.75	0.19	0.015	0.015	0.405	0.054	0.015	389.804	1939.813
25	SkipInit	SGD	False	3	12	0.001	79.52	0.202	0.016	0.016	0.462	0.03	0.002	442.984	584.038

Table A.7: Results from the SGLD group, experiments 1-25.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
26	SkipInit	SGD	False	3	8	0.01	73.67	0.231	0.048	0.017	0.585	0.059	0.005	531.033	41450.605
27	SkipInit	SGD	False	3	8	0.006	79.24	0.205	0.021	0.01	0.455	0.063	0.008	246.881	1566.477
28	SkipInit	SGD	False	3	8	0.003	78.66	0.211	0.017	0.007	0.437	0.048	0.006	298.599	8626.144
29	SkipInit	SGD	False	3	8	0.0016	77.45	0.223	0.017	0.009	0.49	0.035	0.001	399.17	836.139
30	SkipInit	SGD	False	3	8	0.001	76.42	0.234	0.018	0.012	0.569	0.027	0.0	399.397	92.225
31	SkipInit	SGD	False	2	16	0.01	71.42	0.201	0.048	0.016	0.573	0.064	0.006	2705.638	84185.773
32	SkipInit	SGD	False	2	16	0.006	81.4	0.183	0.016	0.011	0.403	0.064	0.011	972.945	2224.683
33	SkipInit	SGD	False	2	16	0.003	80.97	0.188	0.013	0.013	0.346	0.041	0.002	345.2	1472.917
34	SkipInit	SGD	False	2	16	0.0016	79.98	0.197	0.014	0.014	0.372	0.026	0.001	345.552	201.181
35	SkipInit	SGD	False	2	16	0.001	79.48	0.202	0.015	0.015	0.432	0.022	0.0	408.572	158.089
36	SkipInit	SGD	False	2	12	0.01	63.6	0.04	0.049	0.018	0.535	0.036	0.003	926.639	101084.609
37	SkipInit	SGD	False	2	12	0.006	80.03	0.197	0.019	0.01	0.415	0.057	0.007	388.249	1434.108
38	SkipInit	SGD	False	2	12	0.003	79.93	0.197	0.014	0.014	0.375	0.036	0.005	250.52	718.963
39	SkipInit	SGD	False	2	12	0.0016	80.06	0.196	0.015	0.015	0.419	0.024	0.001	269.671	296.301
40	SkipInit	SGD	False	2	12	0.001	78.9	0.209	0.016	0.016	0.498	0.024	0.0	332.843	255.321
41	SkipInit	SGD	False	2	8	0.006	78.55	0.211	0.024	0.01	0.473	0.048	0.012	330.688	9743.203
42	SkipInit	SGD	False	2	8	0.003	78.14	0.216	0.017	0.008	0.465	0.044	0.005	388.115	1007.945
43	SkipInit	SGD	False	2	8	0.0016	76.7	0.23	0.018	0.009	0.533	0.028	0.001	379.875	962.407
44	SkipInit	SGD	False	2	8	0.001	76.34	0.234	0.02	0.013	0.62	0.026	0.001	425.748	457.735
45	SkipInit	SGD	False	2	8	0.01	73.21	0.231	0.053	0.017	0.591	0.064	0.007	587.254	29264.527

Table A.8: Results from the SGLD group, experiments 26-45.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
1	SkipInit	SGD	False	4	16	0.01	50.55	0.492	0.034	0.007	1.06	0.032	0.001	135.995	560.157
2	SkipInit	SGD	False	4	16	0.006	49.83	0.5	0.037	0.008	1.182	0.037	0.001	15.98	586.199
3	SkipInit	SGD	False	4	16	0.003	49.77	0.501	0.042	0.011	1.399	0.044	0.003	3.874	267.832
4	SkipInit	SGD	False	4	16	0.0016	48.77	0.511	0.05	0.01	1.636	0.052	0.004	3.603	436.541
5	SkipInit	SGD	False	4	16	0.001	47.39	0.525	0.059	0.009	1.821	0.059	0.006	3.563	435.227
6	SkipInit	SGD	False	4	12	0.01	49.24	0.505	0.04	0.008	1.105	0.03	0.001	300.26	950.603
7	SkipInit	SGD	False	4	12	0.006	48.42	0.514	0.042	0.009	1.206	0.036	0.001	57.451	478.032
8	SkipInit	SGD	False	4	12	0.003	48.52	0.514	0.047	0.011	1.387	0.044	0.003	5.865	338.035
9	SkipInit	SGD	False	4	12	0.0016	47.22	0.527	0.056	0.01	1.592	0.052	0.004	4.398	554.667
10	SkipInit	SGD	False	4	12	0.001	46.33	0.536	0.065	0.009	1.758	0.057	0.005	4.197	416.829
11	SkipInit	SGD	False	4	8	0.01	45.61	0.5	0.056	0.012	1.228	0.032	0.001	123.802	4771.721
12	SkipInit	SGD	False	4	8	0.006	45.19	0.497	0.057	0.011	1.327	0.036	0.002	20.667	13626.974
13	SkipInit	SGD	False	4	8	0.003	44.4	0.516	0.062	0.012	1.504	0.044	0.003	4.027	14600.616
14	SkipInit	SGD	False	4	8	0.0016	44.62	0.497	0.07	0.015	1.709	0.055	0.004	3.543	22479.484
15	SkipInit	SGD	False	4	8	0.001	44.34	0.474	0.076	0.017	1.87	0.06	0.005	3.449	31695.754
16	SkipInit	SGD	False	3	16	0.01	50.3	0.495	0.033	0.009	0.996	0.032	0.001	629.41	313.288
17	SkipInit	SGD	False	3	16	0.006	51.29	0.486	0.034	0.011	1.09	0.038	0.001	281.739	359.107
18	SkipInit	SGD	False	3	16	0.003	49.67	0.502	0.04	0.012	1.27	0.046	0.003	66.701	403.658
19	SkipInit	SGD	False	3	16	0.0016	48.78	0.512	0.048	0.01	1.469	0.052	0.004	24.893	314.565
20	SkipInit	SGD	False	3	16	0.001	47.68	0.523	0.056	0.009	1.63	0.057	0.005	13.668	303.072
21	SkipInit	SGD	False	3	12	0.01	48.29	0.515	0.04	0.008	1.063	0.03	0.001	664.832	491.162
22	SkipInit	SGD	False	3	12	0.006	49.76	0.502	0.04	0.01	1.147	0.036	0.001	293.995	193.124
23	SkipInit	SGD	False	3	12	0.003	47.7	0.522	0.046	0.011	1.308	0.044	0.003	77.539	317.143
24	SkipInit	SGD	False	3	12	0.0016	46.79	0.532	0.054	0.009	1.485	0.052	0.004	24.767	584.345
25	SkipInit	SGD	False	3	12	0.001	45.94	0.54	0.062	0.008	1.633	0.057	0.005	14.329	381.643

Table A.9: Results from the MLP group, experiments 1-25.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E__50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
26	SkipInit	SGD	False	3	8	0.01	45.68	0.484	0.056	0.013	1.184	0.032	0.001	466.082	8295.984
27	SkipInit	SGD	False	3	8	0.006	45.35	0.503	0.057	0.011	1.263	0.036	0.002	175.477	10187.107
28	SkipInit	SGD	False	3	8	0.003	44.94	0.542	0.061	0.012	1.404	0.044	0.003	34.855	3560.694
29	SkipInit	SGD	False	3	8	0.0016	45.05	0.503	0.068	0.014	1.572	0.052	0.004	11.919	16687.102
30	SkipInit	SGD	False	3	8	0.001	45.92	0.44	0.075	0.017	1.715	0.059	0.005	9.255	48689.891
31	SkipInit	SGD	False	2	16	0.01	50.71	0.492	0.032	0.011	0.96	0.032	0.001	979.61	121.623
32	SkipInit	SGD	False	2	16	0.006	50.01	0.499	0.034	0.012	1.048	0.038	0.002	675.352	125.423
33	SkipInit	SGD	False	2	16	0.003	49.98	0.5	0.039	0.011	1.205	0.048	0.003	333.369	202.665
34	SkipInit	SGD	False	2	16	0.0016	48.8	0.512	0.047	0.009	1.377	0.053	0.004	146.072	137.994
35	SkipInit	SGD	False	2	16	0.001	47.76	0.522	0.055	0.008	1.516	0.059	0.005	87.997	212.765
36	SkipInit	SGD	False	2	12	0.01	49.4	0.505	0.038	0.012	1.036	0.032	0.001	950.871	123.99
37	SkipInit	SGD	False	2	12	0.006	49.12	0.508	0.039	0.012	1.111	0.038	0.002	631.624	83.382
38	SkipInit	SGD	False	2	12	0.003	47.34	0.526	0.045	0.01	1.251	0.046	0.003	299.746	153.501
39	SkipInit	SGD	False	2	12	0.0016	47.37	0.526	0.053	0.009	1.407	0.052	0.004	138.318	211.409
40	SkipInit	SGD	False	2	12	0.001	46.32	0.523	0.062	0.008	1.54	0.06	0.005	88.979	12762.151
41	SkipInit	SGD	False	2	8	0.003	44.0	0.549	0.06	0.011	1.332	0.044	0.003	256.468	3730.373
42	SkipInit	SGD	False	2	8	0.006	44.9	0.544	0.056	0.011	1.213	0.037	0.002	550.549	1316.74
43	SkipInit	SGD	False	2	8	0.01	44.74	0.478	0.056	0.013	1.151	0.032	0.001	788.199	12273.049
44	SkipInit	SGD	False	2	8	0.001	44.4	0.449	0.074	0.017	1.586	0.057	0.005	81.555	40711.32
45	SkipInit	SGD	False	2	8	0.0016	44.79	0.499	0.067	0.014	1.47	0.053	0.004	122.721	20811.758

Table A.10: Results from the MLP group, experiments 26-45.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E_50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
1	SkipInit	SGD	False	4	16	0.01	76.53	0.233	0.013	0.013	0.365	0.082	0.013	81898.086	7660.669
2	SkipInit	SGD	False	4	16	0.006	74.71	0.252	0.015	0.015	0.447	0.085	0.016	57898.148	6223.824
3	SkipInit	SGD	False	4	16	0.003	70.5	0.294	0.019	0.019	0.599	0.09	0.018	48900.586	9115.702
4	SkipInit	SGD	False	4	16	0.0016	68.98	0.31	0.023	0.022	0.777	0.097	0.018	41345.898	8052.807
5	SkipInit	SGD	False	4	16	0.001	65.84	0.341	0.028	0.018	0.939	0.1	0.018	41381.27	7761.956
6	SkipInit	SGD	False	4	12	0.01	75.94	0.238	0.015	0.015	0.417	0.079	0.013	42013.156	12266.437
7	SkipInit	SGD	False	4	12	0.006	73.62	0.262	0.017	0.017	0.505	0.085	0.015	39064.832	10458.587
8	SkipInit	SGD	False	4	12	0.003	70.56	0.293	0.021	0.021	0.672	0.088	0.017	37752.777	8064.518
9	SkipInit	SGD	False	4	12	0.0016	68.78	0.312	0.026	0.018	0.869	0.094	0.017	33769.797	10636.099
10	SkipInit	SGD	False	4	12	0.001	66.43	0.335	0.031	0.015	1.033	0.095	0.018	32879.043	6620.653
11	SkipInit	SGD	False	4	8	0.01	74.87	0.249	0.018	0.018	0.538	0.072	0.011	24909.215	6850.614
12	SkipInit	SGD	False	4	8	0.006	72.33	0.275	0.02	0.019	0.641	0.082	0.013	25576.996	5254.141
13	SkipInit	SGD	False	4	8	0.003	69.55	0.304	0.026	0.016	0.846	0.088	0.013	23793.76	9284.09
14	SkipInit	SGD	False	4	8	0.0016	67.4	0.325	0.033	0.012	1.055	0.093	0.014	22558.457	18295.979
15	SkipInit	SGD	False	4	8	0.001	65.91	0.341	0.04	0.011	1.21	0.095	0.015	21359.785	4168.822
16	SkipInit	SGD	False	3	16	0.01	75.65	0.242	0.014	0.014	0.378	0.078	0.013	41089.785	6312.835
17	SkipInit	SGD	False	3	16	0.006	74.74	0.251	0.016	0.016	0.456	0.082	0.014	40352.473	4085.466
18	SkipInit	SGD	False	3	16	0.003	69.98	0.299	0.019	0.019	0.61	0.089	0.016	32636.953	9757.208
19	SkipInit	SGD	False	3	16	0.0016	69.13	0.308	0.024	0.02	0.808	0.09	0.016	30909.346	5928.891
20	SkipInit	SGD	False	3	16	0.001	67.46	0.325	0.029	0.017	0.969	0.095	0.016	28530.816	3120.503
21	SkipInit	SGD	False	3	12	0.01	74.69	0.251	0.016	0.016	0.461	0.072	0.013	25005.49	8862.343
22	SkipInit	SGD	False	3	12	0.006	74.06	0.258	0.018	0.018	0.555	0.079	0.014	24353.754	5497.137
23	SkipInit	SGD	False	3	12	0.003	70.44	0.294	0.022	0.02	0.741	0.088	0.016	23230.01	9198.185
24	SkipInit	SGD	False	3	12	0.0016	66.83	0.331	0.028	0.015	0.947	0.093	0.017	22727.375	5775.296
25	SkipInit	SGD	False	3	12	0.001	65.19	0.348	0.034	0.014	1.105	0.094	0.018	21250.248	10038.787

Table A.11: Results from the ResNet group, experiments 1-25.

Exp. No.	Arch.	Optim.	Aug.	Depth	Width	LR	Test acc.(%)	Gen. error	SOTL	SOTL -E__50	TSE -EMA	PAC-Bayes sharpness	Value sens.	Initial grad. var.	Final grad. var.
26	SkipInit	SGD	False	3	8	0.01	75.35	0.245	0.018	0.018	0.561	0.073	0.011	17269.584	3604.696
27	SkipInit	SGD	False	3	8	0.006	72.54	0.273	0.021	0.017	0.68	0.08	0.013	17729.357	5877.638
28	SkipInit	SGD	False	3	8	0.003	69.36	0.305	0.027	0.014	0.878	0.082	0.013	16687.762	6480.051
29	SkipInit	SGD	False	3	8	0.0016	67.96	0.32	0.034	0.012	1.077	0.086	0.014	15219.077	3950.61
30	SkipInit	SGD	False	3	8	0.001	66.16	0.338	0.042	0.011	1.224	0.092	0.015	14938.932	4832.864
31	SkipInit	SGD	False	2	16	0.01	76.45	0.235	0.014	0.014	0.404	0.067	0.009	19150.77	2213.968
32	SkipInit	SGD	False	2	16	0.006	74.29	0.256	0.016	0.016	0.494	0.073	0.01	18303.113	1737.045
33	SkipInit	SGD	False	2	16	0.003	71.99	0.28	0.02	0.02	0.665	0.082	0.011	14547.531	4953.809
34	SkipInit	SGD	False	2	16	0.0016	69.86	0.301	0.026	0.017	0.873	0.082	0.012	13270.118	1996.882
35	SkipInit	SGD	False	2	16	0.001	68.44	0.315	0.032	0.014	1.037	0.088	0.012	11683.412	2214.317
36	SkipInit	SGD	False	2	12	0.01	76.31	0.235	0.017	0.017	0.487	0.067	0.008	12164.296	1937.338
37	SkipInit	SGD	False	2	12	0.006	74.65	0.253	0.019	0.019	0.596	0.07	0.008	12666.271	1732.43
38	SkipInit	SGD	False	2	12	0.003	72.81	0.271	0.024	0.016	0.79	0.076	0.01	12005.268	2716.768
39	SkipInit	SGD	False	2	12	0.0016	70.22	0.298	0.031	0.013	0.997	0.082	0.01	9388.286	2822.228
40	SkipInit	SGD	False	2	12	0.001	69.27	0.307	0.038	0.011	1.15	0.083	0.011	8001.839	2646.85
41	SkipInit	SGD	False	2	8	0.01	74.99	0.248	0.021	0.016	0.643	0.064	0.007	8041.948	4009.591
42	SkipInit	SGD	False	2	8	0.006	73.18	0.267	0.024	0.014	0.763	0.073	0.008	7861.652	3052.567
43	SkipInit	SGD	False	2	8	0.003	70.05	0.299	0.031	0.011	0.99	0.076	0.01	7903.873	2224.266
44	SkipInit	SGD	False	2	8	0.0016	68.64	0.314	0.04	0.01	1.193	0.076	0.011	5705.083	1784.172
45	SkipInit	SGD	False	2	8	0.001	67.76	0.322	0.049	0.009	1.332	0.082	0.012	4423.192	2977.202

Table A.12: Results from the ResNet group, experiments 26-45.