Whitening and Second Order Optimization Both Make Information in the Dataset Unusable During Training, and Can Reduce or Prevent Generalization

Neha S. Wadia [1] **Daniel Duckworth** [2] **Samuel S. Schoenholz** [2] **Ethan Dyer** [2] **Jascha Sohl-Dickstein** [2]

Abstract

Machine learning is predicated on the concept of generalization: a model achieving low error on a sufficiently large training set should also perform well on novel samples from the same distribution. We show that both data whitening and second order optimization can harm or entirely prevent generalization. In general, model training harnesses information contained in the sample-sample second moment matrix of a dataset. For a general class of models, namely models with a fully connected first layer, we prove that the information contained in this matrix is the *only information* which can be used to generalize. Models trained using whitened data, or with certain second order optimization schemes, have less access to this information, resulting in reduced or nonexistent generalization ability. We experimentally verify these predictions for several architectures, and further demonstrate that generalization continues to be harmed even when theoretical requirements are relaxed. However, we also show experimentally that *regularized* second order optimization can provide a practical tradeoff, where training is accelerated but less information is lost, and generalization can in some circumstances even improve.

1. Introduction

Whitening is a data preprocessing step that removes correlations between input features (see Fig. 1). It is used across many scientific disciplines, including geology (Gillespie et al., 1986), physics (Jenet et al., 2005), machine learning (Le Cun et al., 1998), linguistics (Abney, 2007), and chemistry (Bro & Smilde, 2014). It has a particularly rich history in neuroscience, where it has been proposed as a mechanism by which biological vision realizes Barlow's redundancy reduction hypothesis (Attneave, 1954; Barlow, 1961; Atick & Redlich, 1992; Dan et al., 1996; Simoncelli & Olshausen, 2001).

Whitening is often recommended since, by standardizing the variances in each direction in feature space, it typically speeds up the convergence of learning algorithms (Le Cun et al., 1998; Wiesler & Ney, 2011), and causes models to better capture contributions from low variance feature directions. Whitening can also encourage models to focus on more fundamental higher-order statistics in data, by removing second-order statistics (Hyvärinen et al., 2009). Whitening has further been a direct inspiration for deep learning techniques such as batch normalization (Ioffe & Szegedy, 2015) and dynamical isometry (Pennington et al., 2017; Xiao et al., 2018).

1.1. Whitening Destroys Information Useful for Generalization

Our argument proceeds in two parts: First, we prove that when a model with a fully connected first layer whose weights are initialized isotropically is trained with either gradient descent or stochastic gradient descent (SGD), information in the data covariance matrix is the *only* information that can be used to generalize. This result is agnostic to the choice of loss function and to the architecture of the model after the first layer. Second, we show that whitening always destroys information in the data covariance matrix.

Whitening the data and then training with gradient descent or SGD therefore results in either diminished or nonexistent generalization properties compared to the same model trained on unwhitened data. The seriousness of the effect varies with the difference between the number of datapoints n and the number of features d, worsening as $n - d$ gets smaller.

Empirically, we find that this effect holds even when the first layer is not fully connected and when its weight initialization is not isotropic - for example, in a convolutional network trained from a Xavier initialization.

[1]University of California, Berkeley [2]Google Brain. Correspondence to: Neha S. Wadia <neha.wadia@berkeley.edu>.

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

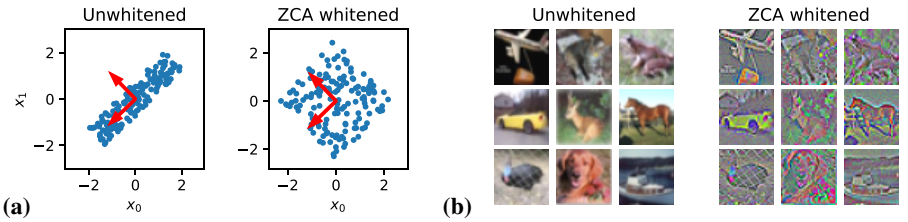

Figure 1. **Whitening removes correlations between feature dimensions in a dataset.** Whitening is a linear transformation of a dataset that sets all non-zero eigenvalues of the covariance matrix to 1. ZCA whitening is a specific choice of the linear transformation that rescales the data in the directions given by the eigenvectors of the covariance matrix, but without additional rotations or flips. *(a)* A toy 2d dataset before and after ZCA whitening. Red arrows indicate the eigenvectors of the covariance matrix of the unwhitened data. *(b)* ZCA whitening of CIFAR-10 images preserves spatial and chromatic structure, while equalizing the variance across all feature directions.

1.2. Second Order Optimization Harms Generalization Similarly to Whitening

Second order optimization algorithms take advantage of information about the curvature of the loss landscape to take a more direct route to a minimum (Boyd & Vandenberghe, 2004; Bottou et al., 2018). There are many approaches to second order or quasi-second order optimization (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970; Dennis Jr & Moré, 1977; Liu & Nocedal, 1989; Schraudolph et al., 2007; Lin et al., 2008; Sunehag et al., 2009; Martens, 2010; Byrd et al., 2011; Vinyals & Povey, 2011; Duchi et al., 2011; Tieleman & Hinton, 2012; Zeiler, 2012; Hennig, 2013; Byrd et al., 2014; Kingma & Ba, 2014; Sohl-Dickstein et al., 2014; Desjardins et al., 2015; Martens & Grosse, 2015; Grosse & Martens, 2016; Agarwal et al., 2016; Zhang et al., 2017; Botev et al., 2017; Martens et al., 2018; George et al., 2018; Lu et al., 2018; Bollapragada et al., 2018; Gupta et al., 2018; Shazeer & Stern, 2018; Berahas et al., 2019; Anil et al., 2019; Agarwal et al., 2019; Osawa et al., 2020), and there is active debate over whether second order optimization harms generalization (Wilson et al., 2017; Zhang et al., 2018; 2019; Amari et al., 2020; Vaswani et al., 2020). The measure of curvature used in these algorithms is often related to feature-feature covariance matrices of the input or of intermediate activations (Martens & Grosse, 2015). In some situations second order optimization is equivalent to steepest descent training on whitened data (Sohl-Dickstein, 2012; Martens & Grosse, 2015).

The similarities between whitening and second order optimization allow us to argue that pure second order optimization also prevents information about the input distribution from being leveraged during training, and can harm generalization (see Figs. 3, 4). Our results are strongest for unregularized, exact second order optimizers and for the large width limit of neural networks. We do find that when strongly regularized and carefully tuned, second order methods can lead to superior performance (Fig. 5).

2. Data Dependence of Training Dynamics and Test Predictions

Consider a dataset $X \in \mathbb{R}^{d \times n}$ consisting of n independent d-dimensional examples. X consists of samples from an underlying data distribution to which we do not have access. We write F for the feature-feature second moment matrix and K for the sample-sample second moment matrix:

$$F = XX^\top \in \mathbb{R}^{d \times d}, \quad K = X^\top X \in \mathbb{R}^{n \times n}. \tag{1}$$

We assume that at least one of F or K is full rank. We omit normalization factors of $1/n$ and $1/d$ in the definitions of F and K, respectively, for notational simplicity in later sections. As defined, K is also the Gram matrix of X.

We are interested in understanding the effect of whitening on the performance of a trained model when evaluated on a test set. In this section, we prove the general result that for any model with a dense, isotropically initialized first layer, the trained model depends on the training inputs only through K. In Section 3, we will show that whitening, and often second order optimization, reduce the information in K. These two results together lead to a conclusion that whitening and second order optimization limit generalization.

2.1. Training Dynamics Depend on the Training Data Only Through Its Second Moments

Consider a model f with a dense first layer Z:

$$f(X) = g_\theta(Z), \quad Z = WX, \tag{2}$$

where W denotes the first layer weights and θ denotes all remaining parameters (see Fig. 2(a)). The structure of $g_\theta(\cdot)$ is unrestricted. W is initialized from an isotropic distribution. We study a supervised learning problem, in which each vector X_i corresponds to a label Y_i.[1] We adopt the notation

[1] Our results also apply to unsupervised learning, which can be viewed as a special case of supervised learning where Y_i contains no information.

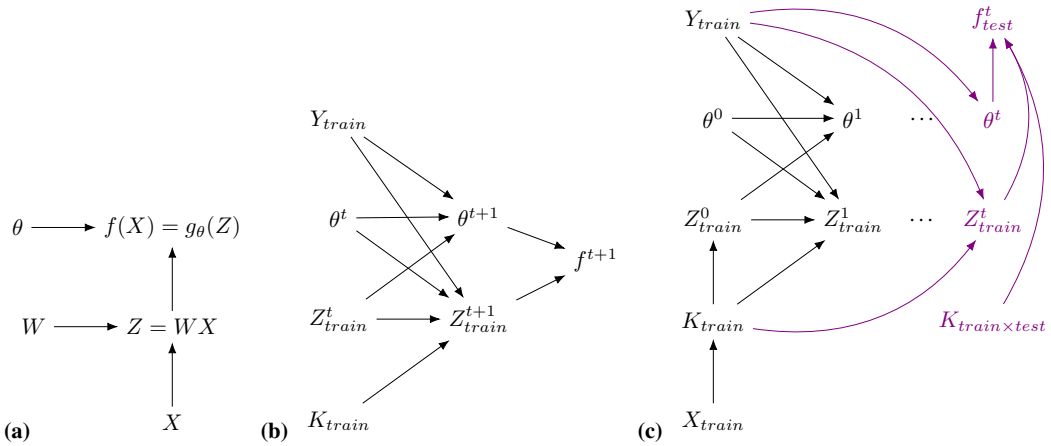

Figure 2. **Activations and weights depend on the training data only through second moments.** *(a)* Our model class consists of a linear transformation $Z = WX$, followed by a nonlinear map $g_\theta(Z)$ with parameters θ. Note that this model class includes fully connected neural networks, among other common machine learning models. *(b)* Causal dependencies for a single gradient descent update. The changes in weights, activations, and model output depend on the training data through the training sample second moment matrix, K_{train}, and the targets, Y_{train}. *(c)* Causal structure for the entire training trajectory. The final weights and training activations only depend on the training data through the training sample second moment matrix K_{train}, and the targets Y_{train}, while the test predictions (in purple) also depend on the mixed second moment matrix, $K_{\text{train}\times\text{test}}$.

$X_{\text{train}} \in \mathbb{R}^{d\times n_{\text{train}}}$ and Y_{train} for the training inputs and labels, and write the corresponding second moment matrices as F_{train} and K_{train}. We consider models with loss $L(f(X); Y)$ trained by SGD. The update rules are

$$\theta^{t+1} = \theta^t - \eta \frac{\partial L^t}{\partial \theta^t}, \tag{3a}$$

$$W^{t+1} = W^t - \eta \frac{\partial L^t}{\partial W^t} = W^t - \eta \frac{\partial L^t}{\partial Z^t_{\text{train}}} X^\top_{\text{train}}, \tag{3b}$$

where t denotes the current training step, η is the learning rate, and L^t is the loss evaluated only on the minibatch used at step t. As a result, the activations $Z_{\text{train}} = WX_{\text{train}}$ evolve as

$$Z^{t+1}_{\text{train}} = Z^t_{\text{train}} - \eta \frac{\partial L^t}{\partial Z^t_{\text{train}}} K_{\text{train}}. \tag{4}$$

Treating the weights, activations, and function predictions as random variables, with distributions induced by the initial distribution over W^0, the update rules (Eqs. 3-4) can be represented by the causal diagram in Fig. 2(b). We can now state one of our main results.

Theorem 2.1.1. *Let $f(X)$ be a function as in Eq. 2, with linear first layer $Z = WX$, and additional parameters θ. Let W be initialized from an isotropic distribution. Further, let $f(X)$ be trained via gradient descent on a training dataset X_{train}. The learned weights θ^t and first layer activations Z^t_{train} are independent of X_{train} conditioned on K_{train} and Y_{train}. In terms of mutual information $\mathcal{I}$, we have*

$$\mathcal{I}(Z^t_{train}, \theta^t; X_{train} \mid K_{train}, Y_{train}) = 0 \ \forall t. \tag{5}$$

Proof. To establish this result, we note that the first layer activation at initialization, Z^0_{train}, is a random variable due to random weight initialization, and only depends on X_{train} through K_{train}:

$$\mathcal{I}(Z^0_{\text{train}}; X_{\text{train}} \mid K_{\text{train}}) = 0. \tag{6}$$

This is a consequence of the isotropy of the initial weight distribution, explained in detail in Appendix A. Note also that the deeper layer weights at initialization are independent of X_{train}:

$$\mathcal{I}(\theta^0; X_{\text{train}}) = 0. \tag{7}$$

Combining these with Eqs. 3-4, the causal diagram for all of training is given by (the black part of) Fig. 2(c). The conditional independence of Eq. 5 follows from this diagram.

□

An alternate proof, by induction rather than using the causal diagram, is presented in Appendix B.

2.2. Test Set Predictions Depend on Train and Test Inputs Only Through Their Second Moments

Let $X_{\text{test}} \in \mathbb{R}^{d\times n_{\text{test}}}$ and Y_{test} be the test data. The test predictions $f_{\text{test}} = f(X_{\text{test}})$ are determined by $Z^t_{\text{test}} = W^t X_{\text{test}}$ and θ^t. To identify sources of data dependence, we can write the evolution of the test set predictions Z_{test} over the course of training in a manner similar to Eq. 4:

$$Z^{t+1}_{\text{test}} = Z^t_{\text{test}} - \eta \frac{\partial L^t}{\partial Z^t_{\text{train}}} K_{\text{train}\times\text{test}}, \tag{8}$$

where $K_{\text{train}\times\text{test}} = X_{\text{train}}^\top X_{\text{test}} \in \mathbb{R}^{n_{\text{train}}\times n_{\text{test}}}$. The initial first layer activations are independent of the training data, and depend on X_{test} only through K_{test}:

$$\mathcal{I}(Z^0_{\text{test}}; X \mid K_{\text{test}}) = 0, \tag{9}$$

where X is the combined training and test data. If we denote the second moment matrix over this combined set by K, then the evolution of the test predictions is described by the (purple part of the) causal diagram in Fig. 2(c), from which we conclude the following.

Theorem 2.2.1. *For a function $f(X)$ as in Eq. 2, with first-layer weights initialized isotropically, trained with the update rules Eqs. 3-4, test predictions depend on the training data only through K and Y_{train}. This is summarized in the mutual information statement*

$$\mathcal{I}(f_{test}; X \mid K, Y_{train}) = 0\,. \tag{10}$$

3. Whitening, Second Order Optimization, and Generalization

In Section 2, we established that trained models with a fully connected, isotropically initialized first layer depend on the input data only through K. In this section we show that by removing information from F, whitening removes information in K that could otherwise be used to generalize. In the extreme case $n \leq d$, K is trivialized, and we show that any generalization ability in a model trained in this regime relies solely on linear interpolation between inputs. We offer a detailed theoretical study of these effects in a linear model, and we use this example to make a connection with unregularized second order optimization.

We begin with the definition of whitening.

Definition 3.0.1 (Whitening)**.** *Any linear transformation M s.t. $\hat{X} = MX$ maps the eigenspectrum of F to ones and zeros, with the multiplicity of ones given by* $\text{rank}(F)$.

It is natural to consider the two cases $n \leq d$ and $n \geq d$ (when $n = d$ both cases apply).

$$\begin{aligned} n \geq d:\ \hat{F} &= I^{d\times d}\,, \quad \hat{K} = \sum_{i=1}^{d} \hat{u}_i \hat{u}_i^\top\,. \\ n \leq d:\ \hat{F} &= \sum_{j=1}^{n} \hat{v}_j \hat{v}_j^\top\,, \quad \hat{K} = I^{n\times n}. \end{aligned} \tag{11}$$

Here, $\hat{F}$ and $\hat{K}$ denote the whitened second moment matrices, and the vectors $\hat{u}_i$ and $\hat{v}_j$ are orthogonal unit vectors of dimension n and d respectively. Eq. 11 follows directly from the fact that $X^\top X$ and $XX^\top$ share nonzero eigenvalues.

3.1. Whitening Harms Generalization

3.1.1. Full data whitening of a high dimensional dataset

We first consider a simplified setup: computing the whitening transform using the combined training and test data. We refer to this as 'full-whitening'. We consider the large feature count ($d \geq n$) regime.

Corollary 3.1.0.1. *When $d \geq n$, and when the whitening transform is computed on the full input dataset X (including both train and test points), then the whitened input data $\hat{X}$ provides no information about the predictions f_{test} of the model on test points. That is,*

$$\mathcal{I}(f_{test}; \hat{X} \mid Y_{train}) = 0\,. \tag{12}$$

Proof. By Eq. 11 we have $\hat{K} = I$. Since $\hat{K}$ is now a constant rather than a random variable, Eq. 10 simplifies directly to Eq. 12. □

To further clarify this prediction, note that Eq. 12 implies $\mathcal{I}(f_{\text{test}}; Y_{\text{test}} \mid Y_{\text{train}}) = 0$ for fully-whitened data because the true test labels are solely determined by X_{test}. Therefore *knowing the model prediction on a test point in this setting gives no information about the true test label.*

3.1.2. Training data whitening of a high dimensional dataset

In practice, we are more interested in the common setting of computing a whitening transform based only on the training data. We call data whitened in this way 'train-whitened'. As mentioned above, the test predictions of a model are entirely determined by the first layer activations Z^t_{test} and the deeper layer weights θ^t. From Theorem 2.1.1 we see that the learned weights θ^t depend on the training data only through K_{train}, and are thus independent of the training data for whitened data:

$$\mathcal{I}(\theta^t; \hat{X}_{\text{train}} \mid Y_{\text{train}}) = 0\,. \tag{13}$$

It is worth emphasizing this point because in most realistic networks the majority of model parameters are contained in these deeper weights θ^t.

Despite the deep layer weights, θ^t, being unable to extract information from the training distribution, the model is not entirely incapable of generalizing to test inputs. This is because the test activations Z_{test} will interpolate between training examples, using the information in $\hat{K}_{\text{train}\times\text{test}}$. More precisely,

$$Z^t_{\text{test}} = Z^0_{\text{test}} + \left(Z^t_{\text{train}} - Z^0_{\text{train}}\right) \hat{K}_{\text{train}\times\text{test}}. \tag{14}$$

This interpolation in Z is the *only* way in which structure

in the inputs X_{train} can drive generalization. This should be contrasted with the case of full-whitening, discussed above, where $\hat{K}_{\text{train}\times\text{test}} = 0$. We therefore predict that when whitening is performed only on the training data, there will be some generalization, but it will be much more limited than can be achieved without whitening.

3.1.3. Full data whitening of lower dimensional datasets

When the dataset size is larger than the data dimensionality, whitening continues to remove information which could otherwise be used for generalization, but it no longer removes *all* of the information in the training inputs. In this regime, by mapping the feature-feature second moment matrix F to the identity matrix, whitening also reduces the degrees of freedom in the sample-sample second moment matrix K. Because information about the training dataset is available to the model only through K (Fig. 2(c)), reducing the degrees of freedom of K also reduces the information available to the model about the training inputs.

Theorem 3.1.1. *Consider a dataset $X \in \mathbb{R}^{d\times n}$, with $n > d$, and where all submatrices formed from d columns of X are full rank (this condition holds in the generic case). Consider the same model class and training procedure as in Theorem 2.1.1. Any dataset X can be compressed to $c \leq nd$ scalar values without losing any of the information that determines the distribution over the test set predictions f_{test} of the trained model. When models are trained on unwhitened data, then $c = \min\left(nd - {}^{(d^2-d)}/_{2}, {}^{(n^2+n)}/_{2}\right)$. However, when models are trained on whitened data, then the whitened dataset can be further compressed to $\hat{c} \leq (n-d)d$ scalars.*

Data whitening therefore reduces the amount of information about the input data that can be used to generate model predictions. See Appendix C for a proof of Theorem 3.1.1.

3.1.4. Summary of predictions

In a model with a fully connected first layer, with first layer weights initialized from an isotropic distribution, whitening the data before training with SGD is expected to result in reduced generalization ability compared to the same model trained on unwhitened data. The severity of the effect varies with the relationship of n to d.

Full data whitening when $n < d$ is a limiting case in which generalization is expected to be completely destroyed. When $n \leq d$ and the data is train-whitened, generalization is forced to rely solely on interpolation and is expected to be poor. When $n > d$ and the data is either fully or train-whitened, model predictions still depend on strictly less information than would be available had the data not been whitened, and once again generalization is expected to suffer. For $n \gg d$, the effect of whitening on generalization is expected to be minimal.

As we discuss in Section 3.3, these same predictions apply to second order optimization of linear models and of overparameterized networks (with d corresponding the number of parameters rather than the number of input dimensions).

3.2. Whitening in Linear Least Squares Models

Due to the fact that they are exactly solvable, linear models $f = WX$ provide intuition for *why* whitening can be harmful as we proved in the last section. We discuss this intuition briefly here. A detailed exposition is in Appendix D.

Consider the low dimensional case $d < n$, where the loss has a unique global optimum $W^\star$. The model predictions at this optimum are invariant to whitening. However, whitening has an effect on the *dynamics* of model predictions over the course of training. When, as is typical in real-world problems, training is performed with early stopping based on validation loss, predictions differ considerably for models trained on whitened and unwhitened data. These benefits from early stopping can be related to benefits from weight regularization (Yao et al., 2007).

We focus on the continuous-time picture because it is the clearest, but similar statements can be made for gradient descent. Recall that v_i are the eigenvectors of F_{train}. Denoting the corresponding eigenvalues by λ_i, the dynamics of W under gradient flow for a mean squared loss are given by the decomposition

$$W(t) = \sum_{i=1}^{d} v_i w_i(t), \text{ where} \tag{15}$$
$$w_i(t) = e^{-t\lambda_i} w_i(0) + (1 - e^{-\lambda_i t}) w_i^\star.$$

Eq. 15 shows that larger principal components of the data are learned faster than smaller ones. Whitening destroys this hierarchy by setting $\lambda_i = 1\ \forall i$. If, for example, the data has a simplicity bias (large principal components correspond to signal and small ones correspond to noise), whitening forces the learning algorithm to fit signal and noise directions simultaneously, which results in poorer generalization at finite times during training than would be observed without whitening.

3.3. Newton's Method is Equivalent to Training on Whitened Data for Linear Least Squares Models and for Overparameterized Neural Networks

Though in practice unregularized Newton's method is rarely used as an optimization algorithm due to computational complexity, a poorly conditioned Hessian, or poor generalization performance, it serves as the basis of and as a limiting case for most second order methods. Furthermore,

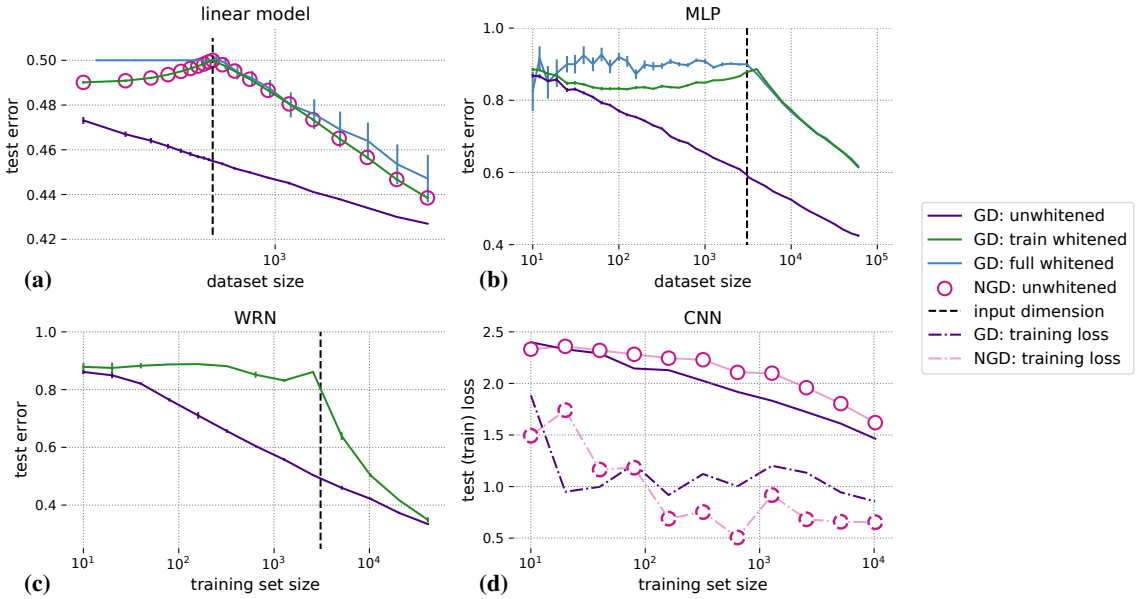

Figure 3. **Whitening and second order optimization reduce or prevent generalization.** *(a)-(c)* Models trained on both full-whitened data (blue; panes a,b) and train-whitened data (green; panes a-c) consistently underperform models trained by gradient descent on unwhitened data (purple; all panes). In (a), Newton's method on unwhitened data (pink circles) behaves identically to gradient descent on whitened data. *(d)* Second order optimization in a convolutional network results in poorer generalization properties than steepest descent. Points plotted correspond to the learning rate and training step with the best validation loss for each method; data for this experiment was unwhitened. CIFAR-10 is used for all experiments (see Appendix F for experiments on MNIST). In (c) and (d) we use a cross entropy loss (see Appendix G for details).

in the case of linear least squares models or wide neural networks, it is equivalent to Gauss-Newton descent. In this context, by relating Newton's method to whitening in linear models and wide networks, we are able to give an explanation for why unregularized second order methods have poor generalization performance. We find that our conclusions also hold empirically in a deep CNN (see Figs. 3, 4).

3.3.1. Linear least squares

We compare a pure Newton update step on unwhitened data with a gradient descent update step on whitened data in a linear least squares model. The Newton update step uses the Hessian H of the model as a preconditioner for the gradient:

$$W^{t+1}_{\text{Newton}} = W^{t}_{\text{Newton}} - \eta H^{-1} \frac{\partial L^t}{\partial W^t} \,. \tag{16}$$

We allow for a general step size η, with $\eta = 1$ giving the canonical Newton update. When H is rank deficient, we take H^{-1} to be a pseudoinverse. For a linear model with mean squared error (MSE) loss, the Hessian equals the second moment matrix F_{train}, and the model output evolves as

$$f^{t+1}_{\text{Newton}}(X) = f^{t}_{\text{Newton}}(X) - \eta \frac{\partial L^t}{\partial f^t_{\text{Newton}}} X^\top_{\text{train}} F^{-1}_{\text{train}} X \,. \tag{17}$$

We can compare this with the evolution of a linear model $\hat{f}(X) = \hat{W}MX$ trained via gradient descent on whitened data $\hat{X} = MX$ with a mean squared loss:

$$\hat{f}^{t+1}(X) = \hat{f}^{t}(X) - \eta \frac{\partial L^t}{\partial \hat{f}^t} X^\top_{\text{train}} M^\top M X. \tag{18}$$

Noting that $M^\top M = F^{-1}_{\text{train}}$, Eqs. 17 and 18 give identical update rules. Thus if both functions are initialized to have the same output, Newton updates give the same predictions as gradient descent on whitened data. While this correspondence is known in the literature, we can now use it to say something further, namely that by applying the argument in Section 3.1, we expect Newton's method to produce linear models that generalize poorly. This result assumes a mean squared loss, but we find experimentally that generalization is also harmed with a cross entropy loss in Fig. 3(d).

3.3.2. OVERPARAMETERIZED NEURAL NETWORKS

Many neural network architectures, including fully connected and convolutional architectures, behave as linear models in their parameters throughout training in the large width limit (Lee et al., 2019). The large width limit occurs when the number of units or channels in intermediate network layers grows towards infinity. Because of this, *second order optimization harms wide neural networks in the same way it harms linear models*. See Appendix E for details.

In the wide network limit, d corresponds to the number of features rather than the number of input dimensions, and the number of features is equal to the number of parameters. Second order optimization is therefore predicted to be harmful for much larger dataset sizes when optimizing overparameterized neural networks.

4. Experiments

4.1. Model and Task Descriptions

Detailed methods are given in Appendix G.

The kernel of all our experiments is as follows: From a dataset, we draw a number of subsets, tiling a range of dataset sizes. Each subset is divided into train, test, and validation examples, and three copies of the subset are made, two of which are whitened. In one case the whitening transform is computed using only the training examples (train-whitening), and in the other using the training, test, and validation examples (full-whitening). Note that the test set size must be reduced in order to run experiments on small datasets, since the test set is considered part of the dataset for full whitening. Models are trained from random initialization on each of the three copies of the data using the same training algorithm and stopping criterion. Test errors and the number of training epochs are recorded.

We emphasize that in any single experiment in which whitening is performed, the same whitening transform is always applied to train, test, and validation data. Experiments differ in the specific subset of the data (train only or train + test + validation) on which the whitening transform is computed.

Linear models and MLPs. To experimentally demonstrate theoretical results, we study CIFAR-10 classification in linear models and CIFAR-10 and MNIST classification in three-layer, fully connected multilayer perceptrons (MLPs). Linear models were trained by optimizing mean squared error loss, where the model outputs were a linear map between the 512-dimensional outputs of a four layer convolutional network at random initialization on CIFAR-10, and their 10-dimensional one-hot labels. This setup is in part motivated by analogy to training the last linear readout layer of a deep neural network. We solved the gradient flow equation for the time at which the MSE on the validation set is lowest, and report the test error at that time. The experiment was repeated using continuous-time Newton's method, consisting of continuous-time gradient descent using an inverse Hessian preconditioner. MLPs were trained using SGD with constant step size until the training accuracy reaches a fixed cutoff threshold, at which point test accuracy was measured.

Convolutional networks. Since our theoretical results on the effect of whitening apply only to models with a fully connected and isotropically initialized first layer, we test whether the same qualitative behavior is observed in CNNs trained from a Xavier initialization. We chose the popular wide residual (WRN) architecture (Zagoruyko & Komodakis, 2016), trained on CIFAR-10. Training was performed using full batch gradient descent with a cosine learning rate schedule for a fixed number of epochs. Full batch training was used to remove experimental confounds from choosing minibatch sizes at different dataset sizes. A validation set was split from the CIFAR-10 training set. Test error corresponding to the parameter values with the lowest validation error was reported.

We also trained a smaller CNN (a ResNet-50 convolutional block followed by an average pooling layer and a dense linear layer) on unwhitened data with full batch gradient descent and with the Gauss-Newton method (with and without a scaled identity regularizer) to compare their respective generalization performances. A grid search was performed over learning rate, and step sizes were chosen using a backoff line search initialized at that learning rate. Test and training losses corresponding to the best achieved validation loss were reported. Note that this experiment is relatively large scale; because we perform full second order optimization to avoid confounds due to choosing a quasi-Newton approximation, iterations are cubic in the number of model parameters.

4.2. Experimental Results

Whitening and second order optimization impair generalization. In agreement with theory, in Figs. 3(a) and (b), linear models and MLPs trained on fully whitened data generalize at chance levels when the size of the dataset is smaller than the dimensionality of the data, and models trained on train-whitened data perform strictly worse than those trained on unwhitened data. Furthermore, the generalization ability of these models recovers only *gradually* as the dataset grows. On CIFAR-10, a 20% gap in performance between MLPs trained on whitened and unwhitened data persists even at the largest dataset size, suggesting that whitening can remain detrimental even when the number of training examples exceeds the number of features by an order of magnitude.

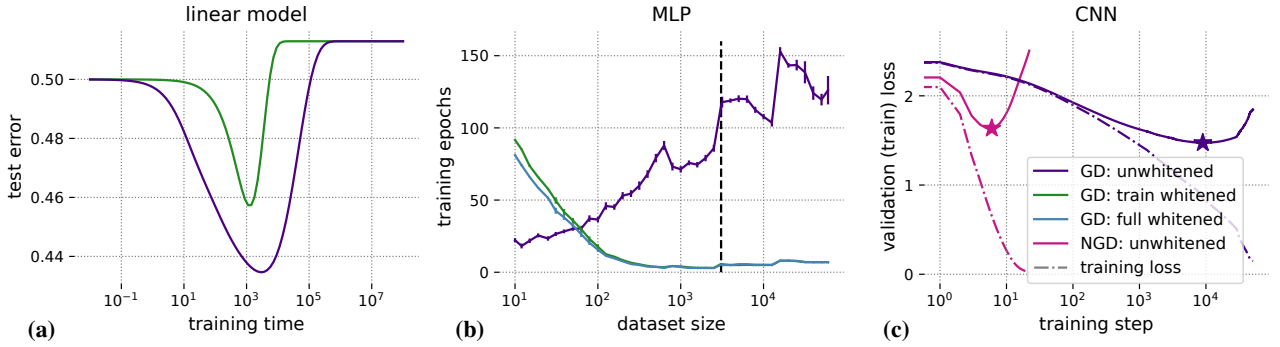

Figure 4. **Models trained on whitened data or with second order optimizers converge faster.** *(a)* Linear models trained on whitened data optimize faster, but their best test accuracy was always worse. Data plotted here is for a training set of size 2560. Similar results for smaller training set sizes are given in Fig. App.1. *(b)* Whitening the data significantly lowers the number of epochs needed to train an MLP to a fixed cutoff in training accuracy, when the learning rate and all other training parameters are kept constant. Discrete jumps in the plot data correspond to points at which the (constant) learning rate was changed. The dashed vertical line indicates the input dimensionality of the data. See Appendix G for details. *(c)* Second order optimization accelerates training on unwhitened data in a convolutional network, compared to gradient descent. Data shown is for a training set of size 10240. Stars correspond to values of the validation loss at which test and training losses are plotted in Fig. 3(d).

In Fig. 3(c) we see a generalization gap in the high dimensional regime between WRNs trained on train-whitened versus unwhitened data, which persists when the size of the dataset grows beyond its dimensionality. This is despite the fact that the convolutional input layer violates the theoretical requirement of a fully connected first layer, and that we used a Xavier initialization scheme, therefore also violating the theoretical requirement for an isotropic first-layer weight initialization. We note that these results are consistent with the whitening experiments in the original WRN paper (Zagoruyko & Komodakis, 2016). Generalization ability begins to recover before the size of the training set reaches its input dimensionality, suggesting that the effect of whitening can be countered by engineering knowledge of the data statistics into the model architecture.

In Fig. 3(a), we demonstrate experimentally the correspondence we proved in Section 3.3. In Fig. 3(d), we observe that pure second order optimization similarly harms generalization even in a convolutional network. Despite training to lower values of the training loss, a CNN trained with an unregularized Gauss-Newton method exhibits higher test loss (at the training step with best validation loss) than the same model trained with gradient descent.

Whitening and second order optimization accelerate training. In Figs. 4(a) and App.1, linear models trained on whitened data or with a second order optimizer converge to their final loss faster than models trained on unwhitened data, but their best test performance is always worse. In Fig. 4(b), MLPs trained on whitened CIFAR-10 data take fewer epochs to reach the same training accuracy cutoff than models trained on unwhitened data, except at very small (< 50) dataset sizes. The effect is stark at large dataset sizes, where the gap in the number of training epochs is two orders of magnitude large. Second order optimization similarly speeds up training in a convolutional network. In Fig. 4(c), unregularized Gauss-Newton descent achieves its best validation loss two orders of magnitude faster (as measured in the number of training steps) than gradient descent.

Regularized second order optimization can simultaneously accelerate training and improve generalization. In Fig. 5 we perform full batch second order optimization with preconditioner $((1-\lambda)B + \lambda I)^{-1}$, where $\lambda \in [0, 1]$ is a regularization coefficient, and B^{-1} is the unregularized Gauss-Newton preconditioner. $\lambda = 0$ corresponds to unregularized Gauss-Newton descent, while $\lambda = 1$ corresponds to full batch steepest descent. At all values of λ, regularized Gauss-Newton achieves its lowest validation loss in fewer training steps than steepest descent (Fig. 5(b)). For some values of λ, the regularized Gauss-Newton method additionally produces lower test loss values than steepest descent (Fig. 5(a)).

Writing the preconditioner in terms of the eigenvectors, $\hat{e}_i$, and eigenvalues, μ_i, of B,

$$((1-\lambda)B + \lambda I)^{-1} = \sum_i \frac{1}{(1-\lambda)\mu_i + \lambda} \hat{e}_i \hat{e}_i^T \,, \quad (19)$$

we see that regularized Gauss-Newton optimization acts similarly to unregularized Gauss-Newton in the subspace spanned by eigenvectors with eigenvalues larger than $\lambda/(1-$

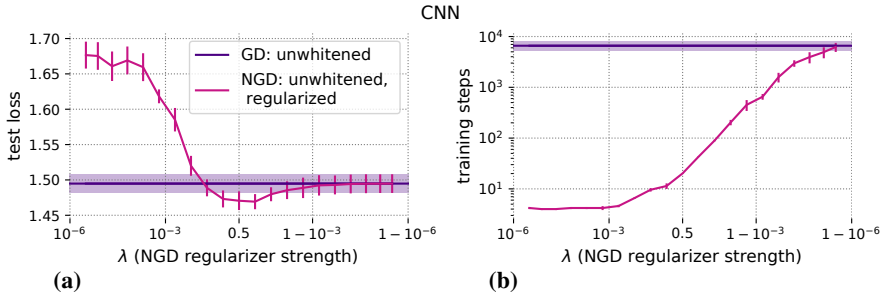

Figure 5. **Regularized second order methods can train faster than gradient descent, with minimal or even positive impact on generalization.** Models were trained on a size 10240 subset of CIFAR-10 by minimizing a cross entropy loss. Error bars indicate twice the standard error in the mean. *(a)* Test loss as a function of regularizer strength. At intermediate values of λ, the second order optimizer produces *lower* values of the test loss than gradient descent. Test loss is measured at the training step corresponding to the best validation performance for both algorithms. See text for further discussion. *(b)* At all values of $\lambda < 1$, the second order optimizer requires fewer training steps to achieve its best validation performance.

λ), and similarly to steepest descent in the subspace spanned by eigenvectors with eigenvalues smaller than $\lambda/(1-\lambda)$. We therefore suggest that regularized Gauss-Newton should be viewed as discarding information in the large-eigenvector subspace, though our theory does not formally address this case. As λ increases from zero to one, the ratio $\lambda/(1-\lambda)$ increases from zero to infinity. Regularized Gauss-Newton method therefore has access to information about the relative magnitudes of more and more of the principal components in the data as λ grows larger. We interpret the improved test performance with regularized Gauss-Newton at about $\lambda = 0.5$ in Fig. 5(a) as suggesting that this loss of information within the leading subspace is actually beneficial for the model on this dataset, likely due to aspects of the model's inductive bias which are actively harmful on this task.

5. Discussion

Are whitening and second order optimization a good idea? Our work suggests that whitening and second order optimization come with costs – a likely reduction in the best achievable generalization. However, both can drastically decrease training time – an effect we also see in our experiments. As compute is often a limiting factor on performance (Shallue et al., 2018), there are many scenarios where faster training may be worth the reduction in generalization. Additionally, the negative effects may be largely resolved if the whitening transform or second order preconditioner are regularized, as is often done in practice (Grosse & Martens, 2016). We observe benefits from regularized second order optimization in Fig. 5, and similar results have been observed for whitening (Lee et al., 2020).

Directions for future work. The practice of whitening has, in the machine learning community, largely been replaced by batch normalization, for which it served as inspiration (Ioffe & Szegedy, 2015). Studying connections between whitening and batch normalization, and especially understanding the degree to which batch normalization destroys information about the data distribution, may be particularly fruitful. Indeed, some results already exist in this direction (Huang et al., 2018).

Most second order optimization algorithms involve regularization, structured approximations to the Hessian, and often non-stationary online approximations to curvature. Understanding the implications of our theory results for practical second order optimization algorithms should prove to be an extremely fruitful direction for future work. It is our suspicion that more mild loss of information about the training inputs will occur for many of these algorithms. In addition, it would be interesting to understand how to relax the large width requirement in our theoretical analysis.

Recent work analyzes deep neural networks through the lens of information theory (Banerjee, 2006; Tishby & Zaslavsky, 2015; Alemi et al., 2016; Bassily et al., 2017; Shwartz-Ziv & Tishby, 2017; Achille & Soatto, 2017; Kolchinsky et al., 2018; Amjad & Geiger, 2018; Achille & Soatto, 2019; Saxe et al., 2019; Schwartz-Ziv & Alemi, 2019), often computing measures of mutual information similar to those we discuss. Our result that the only usable information in a dataset is contained in its sample-sample second moment matrix K may inform or constrain this type of analysis.

Acknowledgements

We thank Jeffrey Pennington for help formulating the project, and Justin Gilmer, Roger Grosse, Nicolas Le Roux, and Jesse Livezey for detailed feedback on a manuscript draft. This work was done while Neha Wadia was an intern at Google Brain.

References

Abney, S. *Semisupervised learning for computational linguistics*. Chapman and Hall/CRC, 2007.

Achille, A. and Soatto, S. Emergence of Invariance and Disentangling in Deep Representations. *Proceedings of the ICML Workshop on Principled Approaches to Deep Learning*, 2017.

Achille, A. and Soatto, S. Where is the information in a deep neural network? *arXiv preprint arXiv:1905.12213*, 2019.

Agarwal, N., Bullins, B., and Hazan, E. Second-order stochastic optimization for machine learning in linear time. *arXiv preprint arXiv:1602.03943*, 2016.

Agarwal, N., Bullins, B., Chen, X., Hazan, E., Singh, K., Zhang, C., and Zhang, Y. Efficient full-matrix adaptive regularization. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 102–110, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Aitken, K. and Gur-Ari, G. On the asymptotics of wide networks with polynomial activations. To appear.

Alemi, A. A., Fischer, I., Dillon, J. V., and Murphy, K. Deep variational information bottleneck. *arXiv:1612.00410*, 2016.

Amari, S., Ba, J., Grosse, R., Li, X., Nitanda, A., Suzuki, T., Wu, D., and Xu, J. When does preconditioning help or hurt generalization?, 2020.

Amjad, R. A. and Geiger, B. C. How (not) to train your neural network using the information bottleneck principle. *arXiv preprint arXiv:1802.09766*, 2018.

Andreassen, A. and Dyer, E. Asymptotics of wide convolutional neural networks. To appear.

Anil, R., Gupta, V., Koren, T., and Singer, Y. Memory-efficient adaptive optimization for large-scale learning. *arXiv preprint arXiv:1901.11150*, 2019.

Atick, J. J. and Redlich, A. N. What does the retina know about natural scenes? *Neural Comput.*, 4(2):196–210, March 1992.

Attneave, F. Some informational aspects of visual perception. *Psychol. Rev*, pp. 183–193, 1954.

Banerjee, A. On bayesian bounds. In *Proceedings of the 23rd international conference on Machine learning*, pp. 81–88. ACM, 2006.

Barlow, H. Possible principles underlying the transformations of sensory messages. *Sensory Communication*, 1, 01 1961.

Bassily, R., Moran, S., Nachum, I., Shafer, J., and Yehudayoff, A. Learners that use little information. *arXiv preprint arXiv:1710.05233*, 2017.

Bell, A. J. and Sejnowski, T. J. The "independent components" of natural scenes are edge filters. *Vision Research*, 37(23):3327 – 3338, 1997.

Berahas, A. S., Jahani, M., and Takáč, M. Quasi-newton methods for deep learning: Forget the past, just sample. *arXiv preprint arXiv:1901.09997*, 2019.

Bollapragada, R., Nocedal, J., Mudigere, D., Shi, H.-J., and Tang, P. T. P. A progressive batching l-BFGS method for machine learning. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 620–629, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Botev, A., Ritter, H., and Barber, D. Practical gauss-newton optimisation for deep learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 557–565. JMLR.org, 2017.

Bottou, L., Curtis, F. E., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

Boyd, S. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, USA, 2004. ISBN 0521833787.

Bro, R. and Smilde, A. K. Principal component analysis. *Analytical Methods*, 6(9):2812–2831, 2014.

Broyden, C. The convergence of a class of double-rank minimization algorithms 2. The new algorithm. *IMA Journal of Applied Mathematics*, 1970.

Byrd, R., Hansen, S., Nocedal, J., and Singer, Y. A Stochastic Quasi-Newton Method for Large-Scale Optimization. *arXiv preprint arXiv:1401.7020*, 2014.

Byrd, R. H., Chin, G. M., Neveitt, W., and Nocedal, J. On the use of stochastic hessian information in optimization methods for machine learning. *SIAM Journal on Optimization*, 21(3):977–995, 2011.

Dan, Y., Atick, J. J., and Reid, R. C. Efficient coding of natural scenes in the lateral geniculate nucleus: Experimental test of a computational theory. *Journal of Neuroscience*, 16(10):3351–3362, 1996.

Dennis Jr, J. E. and Moré, J. J. Quasi-Newton methods, motivation and theory. *SIAM review*, 19(1):46–89, 1977.

Desjardins, G., Simonyan, K., Pascanu, R., and kavukcuoglu, k. Natural neural networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28*, pp. 2071–2079. Curran Associates, Inc., 2015.

Duchi, J., Hazan, E., and Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Dyer, E. and Gur-Ari, G. Asymptotics of wide networks from feynman diagrams. *ArXiv*, abs/1909.11304, 2020.

Fletcher, R. A new approach to variable metric algorithms. *The computer journal*, 1970.

George, T., Laurent, C., Bouthillier, X., Ballas, N., and Vincent, P. Fast approximate natural gradient descent in a kronecker-factored eigenbasis. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 9573–9583, Red Hook, NY, USA, 2018. Curran Associates Inc.

Gillespie, A. R., Kahle, A. B., and Walker, R. E. Color enhancement of highly correlated images. i. decorrelation and hsi contrast stretches. *Remote Sensing of Environment*, 20(3):209–235, 1986.

Goldfarb, D. A family of variable-metric methods derived by variational means. *Mathematics of computation*, 1970.

Grosse, R. and Martens, J. A kronecker-factored approximate fisher matrix for convolution layers. *arXiv preprint arXiv:1602.01407*, 2016.

Gupta, V., Koren, T., and Singer, Y. Shampoo: Preconditioned stochastic tensor optimization. *CoRR*, abs/1802.09568, 2018.

Hennig, P. Fast probabilistic optimization from noisy gradients. *International Conference on Machine Learning*, 2013.

Huang, J. and Yau, H. B. Dynamics of deep neural networks and neural tangent hierarchy. *ArXiv*, abs/1909.08156, 2019.

Huang, L., Yang, D., Lang, B., and Deng, J. Decorrelated batch normalization. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 791–800, 2018.

Hyvärinen, A., Hurri, J., and Hoyer, P. O. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision.* Springer Publishing Company, Incorporated, 1st edition, 2009.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 448–456. JMLR.org, 2015.

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

Jenet, F. A., Hobbs, G. B., Lee, K., and Manchester, R. N. Detecting the stochastic gravitational wave background using pulsar timing. *The Astrophysical Journal Letters*, 625(2):L123, 2005.

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

Kolchinsky, A., Tracey, B. D., and Van Kuyk, S. Caveats for information bottleneck in deterministic scenarios. *arXiv preprint arXiv:1808.07593*, 2018.

Kraft, H. and Procesi, C. *Classical invariant theory: a primer*. 1996.

Le Cun, Y., Bottou, L., Orr, G. B., and Müller, K.-R. Efficient backprop. In *Neural Networks, Tricks of the Trade*, Lecture Notes in Computer Science LNCS 1524. Springer Verlag, 1998.

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in neural information processing systems*, pp. 8570–8581, 2019.

Lee, J., Schoenholz, S. S., Pennington, J., Adlam, B., Xiao, L., Novak, R., and Sohl-Dickstein, J. Finite versus infinite neural networks:an empirical study. *in preparation*, 2020.

Lin, C.-J., Weng, R. C., and Keerthi, S. S. Trust region newton method for logistic regression. *The Journal of Machine Learning Research*, 9:627–650, 2008.

Littwin, E., Galanti, T., and Wolf, L. On the optimization dynamics of wide hypernetworks. *ArXiv*, abs/2003.12193, 2020.

Liu, D. C. D. and Nocedal, J. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

Lu, Y., Harandi, M., Hartley, R. I., and Pascanu, R. Block mean approximation for efficient second order optimization. *ArXiv*, abs/1804.05484, 2018.

Martens, J. Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, volume 951, 2010.

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

Martens, J., Ba, J., and Johnson, M. Kronecker-factored curvature approximations for recurrent neural networks. In *International Conference on Learning Representations*, 2018.

Osawa, K., Tsuji, Y., Ueno, Y., Naruse, A., Foo, C., and Yokota, R. Scalable and practical natural gradient for large-scale deep learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

Pennington, J., Schoenholz, S., and Ganguli, S. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In *Advances in neural information processing systems*, pp. 4785–4795, 2017.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F. A., Bengio, Y., and Courville, A. On the spectral bias of neural networks. *arXiv preprint arXiv:1806.08734*, 2018.

Ronen, B., Jacobs, D., Kasten, Y., and Kritchman, S. The convergence rate of neural networks for learned functions of different frequencies. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 4761–4771. Curran Associates, Inc., 2019.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In Bengio, Y. and LeCun, Y. (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., and Cox, D. D. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, Dec 2019.

Schraudolph, N., Yu, J., and Günter, S. A stochastic quasi-Newton method for online convex optimization. *AIstats*, 2007.

Schwartz-Ziv, R. and Alemi, A. A. Information in infinite ensembles of infinitely-wide neural networks. *arXiv preprint arXiv:1911.09189*, 2019.

Shallue, C. J., Lee, J., Antognini, J., Sohl-Dickstein, J., Frostig, R., and Dahl, G. E. Measuring the effects of data parallelism on neural network training. *arXiv preprint arXiv:1811.03600*, 2018.

Shanno, D. Conditioning of quasi-Newton methods for function minimization. *Mathematics of computation*, 1970.

Shazeer, N. and Stern, M. Adafactor: Adaptive learning rates with sublinear memory cost. *arXiv preprint arXiv:1804.04235*, 2018.

Shwartz-Ziv, R. and Tishby, N. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

Simoncelli, E. P. and Olshausen, B. A. Natural image statistics and neural representation. *Annual Review of Neuroscience*, 24(1):1193–1216, 2001.

Sohl-Dickstein, J. The natural gradient by analogy to signal whitening, and recipes and tricks for its use. *arXiv preprint arXiv:1205.1828*, 2012.

Sohl-Dickstein, J., Poole, B., and Ganguli, S. Fast large-scale optimization by unifying stochastic gradient and quasi-newton methods. In *International Conference on Machine Learning*, pp. 604–612, 2014.

Sunehag, P., Trumpf, J., Vishwanathan, S. V. N., and Schraudolph, N. Variable metric stochastic approximation theory. *arXiv preprint arXiv:0908.3529*, August 2009.

Tieleman, T. and Hinton, G. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE, 2015.

Vaswani, S., Babanezhad, R., Gallego, J., Mishkin, A., Lacoste-Julien, S., and Roux, N. L. To each optimizer a norm, to each norm its generalization, 2020.

Vinyals, O. and Povey, D. Krylov subspace descent for deep learning. *arXiv preprint arXiv:1111.4259*, 2011.

Wiesler, S. and Ney, H. A convergence analysis of log-linear training. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, pp. 657–665, USA, 2011. Curran Associates Inc.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pp. 4148–4158, 2017.

Xiao, L., Bahri, Y., Sohl-Dickstein, J., Schoenholz, S. S., and Pennington, J. Dynamical isometry and a mean field theory of CNNs: How to train 10,000-layer vanilla convolutional neural networks. *arXiv preprint arXiv:1806.05393*, 2018.

Yao, Y., Rosasco, L., and Caponnetto, A. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *CoRR*, abs/1605.07146, 2016.

Zeiler, M. D. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

Zhang, G., Wang, C., Xu, B., and Grosse, R. Three mechanisms of weight decay regularization. *arXiv preprint arXiv:1810.12281*, 2018.

Zhang, G., Martens, J., and Grosse, R. B. Fast convergence of natural gradient descent for over-parameterized neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8082–8093. Curran Associates, Inc., 2019.

Zhang, H., Xiong, C., Bradbury, J., and Socher, R. Block-diagonal hessian-free optimization for training neural networks. *CoRR*, abs/1712.07296, 2017.