On the Generalization of Neural Networks Trained with SGD: Information-Theoretical Bounds and Implications

Anonymous Author(s)
Affiliation
Address
email

Abstract

Understanding the generalization behaviour of deep neural networks is an important theme of modern research in machine learning. In this paper, we follow up on a recent work of Neu [49] and present new information-theoretic upper bounds for the generalization error of neural networks trained with SGD. Our bounds and experimental study provide new insights on the SGD training of neural networks. They also point to a new and simple regularization scheme which we show performs comparably to the current state of the art.

1 Introduction

The outstanding performance of deep learning has brought to the surface some intriguing properties of deep neural networks, one of which is the observation that despite their high capacity, deep neural networks tend to generalize well [80]. This contradicts classical wisdom in statistical learning theory (e.g., [71]) and has stimulated intense research interest in understanding the generalization behaviour of modern neural networks.

One theme of research focuses on the study of over-parameterized neural networks, where generalization bounds are obtained [22, 1, 6, 50, 52, 53, 2, 3] and a curious "double descent" phenomenon is observed and analyzed [9, 46, 78]. New bounding techniques for analyzing generalization have also been developed, utilizing information theoretic quantities [64, 65, 77, 5, 13, 69, 23, 7, 4, 30, 84]. The bounds provided by these techniques have the advantages of accounting for both the data structure and the learning algorithm.

The generalization ability of neural networks trained with mini-batched stochastic gradient descent [61], simply referred to SGD in this paper, has also been widely studied. Specifically, built on a connection between stability and generalization [12], a stability-based bound is first presented in [32], followed by a surge of research effort exploiting similar approaches [44, 17, 26, 43, 8]. Information-theoretic bounding techniques have also demonstrated great power in analyzing SGD-like algorithms. For example, [55] is the first to utilize information-theoretical bound in analyzing the generalization ability of SGLD [28, 74]. The bound was subsquently improved by [47, 31, 62, 72]. Inspired by the work of [55], [49] presents an information- theoretic analysis of the models trained with SGD. The analysis of [49] constructs an auxiliary weight process parallel to SGD training and upper-bounds the generalization error through this auxiliary process.

Another line of research connects the generalization of neural networks with the flatness of loss minima [35] found by SGD or its variant [40, 20, 24, 51, 16, 37, 38, 83, 27]. This understanding has led to the discovery of new SGD-based training algorithms for improved generalization. For example, in a concurrent development by [83] and [27], a local "max-pooling" operation is applied to

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

the loss landscape prior to the SGD updates. This approach, referred to as AMP[83] or SAM[27], is shown to make SGD favor flatter minima and achieve the state-of-the-art performance among various competitive regularization schemes [83].

In this paper, we focus on investigating the generalization of neural networks trained with SGD. We build upon the work of [49]. Following the same construction of the auxiliary weight process in [49], we present upper bounds of generalization error that improve upon [49] in two ways. The first improvement is via removing an unnecessary term in the bounds of [49] by invoking the HWI inequality [60]. The second improvement is via replacing a sample-level mutual information term in [49] with an instance-level mutual information term, exploiting a recent result of [13]. The bounds we obtain decompose into two terms, one measuring the impact of training trajectories ("the trajectory term") and the other measuring the impact of the flatness of the found solution ("the flatness term").

We empirically validate the derived bounds. Various insights are also obtained experimentally concerning the generalization of neural networks under SGD training. For example, the batch size of SGD appears to impact the trajectory term and the flatness term in opposite ways, which complicates the overall dependency of generalization error on batch sizes. A particular interesting observation from our experiments is that a key quantity arising in the trajectory term of the bounds, which we refer to as *gradient dispersion*[1], reveals a double descent phenomenon with respect to training epochs. Most intriguingly, the valley in the double descent curve appears to mark the great divide between the "generalization regime" and the "memorization regime" of training. Furthering from this observation, we also show that it is possible to reduce the memorization effect by dynamically clipping the gradient and reducing its dispersion.

Our bounds also inspire a natural and simple solution to alleviate generalization error. Specifically, we propose a new training scheme, referred to as *Gaussian model perturbation* (GMP), aiming at reducing the flatness term of the bounds. This scheme effectively applies a local "average pooling" to the empirical risk surface prior to SGD, greatly resembling the "max-pooling" approach adopted in AMP[83]. We demonstrate experimentally that GMP achieves a competitive performance with the current art of regularization schemes.

Length constraints precludes elaboration at places. The reader is referred to supplementary materials for proofs and additional information.

Other Related Literature Gradient dispersion is mostly studied from optimization perspectives[11, 63, 39, 75, 25]. Prior to this work, only a few works relate gradient dispersion with the generalization behaviour of the networks. In [49, 72], gradient dispersion also appears in the generalization bounds. In [38], gradient dispersion is argued to capture a notion of "flatness" of the local minima of the loss landscape, thereby correlating with generalization.

Injecting noise in the training process has been proposed in various regularization schemes, for example, [10, 14, 15, 68, 73]. But unlike the Gaussian model perturbation scheme derived in this paper, where noise is injected to the model parameters, noise in those schemes is injected either to the training data or to the network activation.

Gradient clipping is a common technique for preventing gradient exploding (see, e.g., [45, 56]). This technique is also used in [82] to accelerate training. In this paper, gradient clipping is used to investigate and control the impact of gradient dispersion on generalization error.

2 Preliminaries

Population Risk, Empirical Risk and Generalization Error Unless otherwise noted, a random variable will be denoted by a capitalized letter (e.g., Z), and its realization denoted by the corresponding lower-case letter (e.g. z). Let $\mathcal{Z}$ be the instance space of interest and μ be an unknown distribution on $\mathcal{Z}$, specifying random variable Z. Let $\mathcal{W} \subseteq \mathbb{R}^d$ be the space of hypotheses. Suppose that a training sample $S = (Z_1, Z_2, \ldots, Z_n)$ is drawn i.i.d. from μ and that a stochastic learning algorithm $\mathcal{A}$ takes S as its input and outputs a hypothesis $W \in \mathcal{W}$ according to some conditional distribution $P_{W|S}$ mapping $\mathcal{Z}^n$ to $\mathcal{W}$. Let $\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}^+$ be a loss function, where $\ell(w, z)$ measures the "unfitness" or "error" of any $z \in \mathcal{Z}$ with respect to a hypothesis $w \in \mathcal{W}$. The population risk, for any

[1]The quantity is often referred to as gradient variance in the literature [49, 72], but we prefer "dispersion" to "variance" so as to better comply with the mathematical conventions and avoid possible confusion.

$w \in \mathcal{W}$, is defined as

$$L_\mu(w) \triangleq \mathbb{E}_{Z\sim\mu}[\ell(w, Z)].$$

The goal of learning is to find a hypothesis w that minimizes the population risk. But since μ is only partially accessible via the sample S, in practice, we instead turn to the empirical risk, defined as

$$L_S(w) \triangleq \frac{1}{n}\sum_{i=1}^{n} \ell(w, Z_i).$$

The expected generalization error of the learning algorithm $\mathcal{A}$ is then defined as

$$\mathrm{gen}(\mu, P_{W|S}) \triangleq \mathbb{E}_{W,S}[L_\mu(W) - L_S(W)],$$

where the expectation is taken over the joint distribution of (S, W) (i.e., $\mu^n \otimes P_{W|S}$).

Throughout this paper, we take ℓ as a continuous function (adopting the usual notion "surrogate loss" [66].). Additionally, we assume that ℓ is differentiable almost everywhere with respect to both w and z. Furthermore we assume that $\ell(w, Z)$ is R-subgaussian[2] for any $w \in \mathcal{W}$. Note that a bounded loss is guaranteed to be subgaussian for all μ and all $w \in \mathcal{W}$. Let $I(X; Y)$ denote the mutual information [18] between any pair of random variables (X, Y). The following results are known.

Lemma 1 ([77, Theorem 1.]). *The expected generalization error of algorithm $\mathcal{A}$ is bounded by*

$$|\mathrm{gen}(\mu, P_{W|S})| \leq \sqrt{\frac{2R^2}{n} I(W; S)},$$

Lemma 2 ([13, Proposition 1.]). *The expected generalization error of algorithm $\mathcal{A}$ is bounded by*

$$|\mathrm{gen}(\mu, P_{W|S})| \leq \frac{1}{n}\sum_{i=1}^{n} \sqrt{2R^2 I(W; Z_i)},$$

Stochastic Gradient Descent We now restrict the learning algorithm $\mathcal{A}$ to be the mini-batched stochastic gradient descent (SGD) algorithm for empirical risk minimization. For each training epoch, the dataset S is randomly split into m disjoint mini-batches, each having size b, namely, $n = mb$. Based on each batch, one parameter update is performed. Specifically, let B_t denote the batch used for the t^{th} update. Define

$$g(w, B_t) \triangleq \frac{1}{b}\sum_{z\in B_t} \nabla_w \ell(w, z),$$

namely, $g(w, B_t)$ is the average gradient computed for the batch B_t with respect to parameter w. The rule for the t^{th} parameter update is then

$$W_t \triangleq W_{t-1} - \lambda_t g(W_{t-1}, B_t),$$

where λ_t is the learning rate at the step t. The initial parameter setting W_0 is assumed to be drawn from the zero-mean spherical Gaussian $\mathcal{N}(0, \sigma_0^2 \mathbf{I}_d)$ with variance σ_0^2 in each dimension. We will assume that the SGD algorithm stops after T updates and outputs W_T as the learned model parameter.

Given the training sample S, let ξ govern the randomness in the sequence $(B_1, B_2, \ldots, B_T)$ of batches. For the simplicity of notion, we will fix the configuration of ξ. That is, we will assume a fixed "batching trajectory", or a fixed way to shuffle the example indices $\{1, \ldots, n\}$ and divide them into m batches in each epoch. The presented generalization bounds of this paper can be extended to the case where the batching trajectory is uniformly random (as we set up above). This merely involves averaging over all batching trajectories or taking expectation over ξ.

Auxiliary Weight Process We now associate with the SGD algorithm an auxiliary weight process $\{\widetilde{W}_t\}$. Let σ^2 be given, and let $\sigma_1, \sigma_2, \ldots, \sigma_T$ be a sequence of positive real numbers. Define

$$\widetilde{W}_0 \triangleq W_0, \quad \text{and} \quad \widetilde{W}_t \triangleq \widetilde{W}_{t-1} - \lambda_t g(W_{t-1}, B_t) + N_t, \ \text{for } t > 0,$$

where $N_t \sim \mathcal{N}(0, \sigma_t^2 \mathbf{I}_d)$ is a Gaussian noise. The relationship between this auxiliary weight process $\{\widetilde{W}_t\}$ and the weight process $\{W_t\}$ in SGD is shown in the Bayesian network below.

[2]Recall that a random variable X is R-subgaussian [60] if for any ρ, $\log \mathbb{E}\exp\left(\rho\left(X - \mathbb{E}X\right)\right) \leq \rho^2 R^2/2$.

$$
\begin{array}{ccccccccccc}
 & & N_1 & & N_2 & & \cdots & & N_{T-1} & & N_T \\
 & & \downarrow & & \downarrow & & & & \downarrow & & \downarrow \\
\widetilde{W}_0 & \to & \widetilde{W}_1 & \to & \widetilde{W}_2 & \to & \cdots & \to & \widetilde{W}_{T-1} & \to & \widetilde{W}_T \\
\| & \nearrow & & \nearrow & & \nearrow & & & & \nearrow & \\
W_0 & \to & W_1 & \to & W_2 & \to & \cdots & \to & W_{T-1} & \to & W_T
\end{array}
$$

Let $\Delta_t = \sum_{\tau=1}^{t} N_\tau$. Noting that the weight updates in $\{\widetilde{W}_t\}$ uses the same gradient signal as that used in $\{W_t\}$ (which depends on W_{t-1} not $\widetilde{W}_{t-1}$), it is immediate that $\widetilde{W}_t = W_t + \Delta_t$. Note that this auxiliary process follows the same construction as [49], which we will use to study the generalization error of SGD.

To that end, define *gradient dispersion* at parameter w by

$$\mathbb{V}(w) \triangleq \mathbb{E}\left[||\nabla_w \ell(w, Z) - \mathbb{E}\left[\nabla_w \ell(w, Z)\right]||_2^2\right],$$

where the expectation is taken over $Z \sim \mu$.

For a given sample $s \in \mathcal{Z}^n$, define

$$\gamma(w, s) \triangleq \mathbb{E}\left[L_s(w + \Delta_T) - L_s(w)\right],$$

where the expectation is taken over Δ_T and $L_s(w)$ is the empirical risk of s at parameter w.

In the remainder of the paper, let S' denote another sample drawn from μ^n, independent of all other random variables. The main generalization bound in [49] is re-stated below.

Lemma 3 ([49, Theorem 1.])**.** *The generalization error of SGD is upper bounded by*

$$|\mathrm{gen}(\mu, P_{W_T|S})| \leq \sqrt{\frac{2R^2}{n}\sum_{t=1}^{T}\frac{\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\Psi(W_{t-1}) + \frac{\mathbb{V}(W_{t-1})}{b}\right]} + \left|\mathbb{E}\left[\gamma(W_T, S) - \gamma(W_T, S')\right]\right|,$$

where $\Psi(w_{t-1}) \triangleq \mathbb{E}\left[||\nabla_w \ell(w_{t-1}, Z) - \nabla_w \ell(w_{t-1} + \zeta, Z)||_2^2\right]$ *and* $\zeta \sim \mathcal{N}(0, 2\sum_{i=1}^{t-1}\sigma_i^2 \mathrm{I}_d)$.

The term $\Psi(w_{t-1})$ in the bound is referred to as "local gradient sensitivity" in [49].

3 New Generalization Bounds for SGD

We first prove that the generalization bound in Lemma 3 can be tightened by removing the local gradient sensitivity term $\Psi(w_{t-1})$. The key observation is that an independence condition used for establishing Lemma 3 in [49] is unnecessary (see Lemma 4 in [49]). This requires invoking a vector version of the HWI inequality [60, Lemma 3.4.2], which we prove in this paper.

Lemma 4. *Let X and Y be two random vectors in $\mathbb{R}^d$, and let $N \sim \mathcal{N}(0, \mathbf{I}_d)$ be independent of (X, Y). Then, for every $t > 0$,* $\mathrm{D}_{\mathrm{KL}}(P_{X+\sqrt{t}N}||P_{Y+\sqrt{t}N}) \leq \frac{1}{2t}\mathbb{E}\left[||X - Y||^2\right]$.

Here D_{KL} is the KL divergence. Note that the bound in Lemma 3 relies on a similar result which however requires the independence of X and Y. Using Lemma 4, we obtain the following theorem.

Theorem 1. *The generalization error of SGD is upper bounded by*

$$|\mathrm{gen}(\mu, P_{W_T|S})| \leq \sqrt{\frac{2R^2}{nb}\sum_{t=1}^{T}\frac{\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]} + \left|\mathbb{E}\left[\gamma(W_T, S) - \gamma(W_T, S')\right]\right|.$$

The proof of this theorem, as the bounds in [49], relies on Lemma 1 and the sample-level mutual information bound therein. This theorem can be further tightened by exploiting the fact that the instance-level mutual information bound in Lemma 2 is in fact tighter than the sample-level mutual information bound in Lemma 1, as shown in [13]. The main ingredient to proceed in this direction is the following lemma.

Lemma 5. *Let $G_t = -\lambda_t g(W_{t-1}, B_t)$. If $Z_i \in B_t$, then* $I(G_t + N_t; Z_i|\widetilde{W}_{t-1}) \leq \frac{\lambda_t^2}{\sigma_t^2 b^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]$.

In this lemma, the mutual information $I(G_t + N_t; Z_i|\widetilde{W}_{t-1})$ roughly indicates the degree by which the SGD's updating signal G_t (smoothed with noise) depends on an individual training instance Z_i, when Z_i is used for computing the gradient. When this dependency is strong (giving rise to a high value of the mutual information), the model conceivably tends to overfit the individual training instances. This lemma suggests that the strength of this dependency can be upper-bounded by the expected gradient dispersion at the current weight configuration. In our experiments, we will estimate the expected gradient dispersion and validate this intuition.

It is remarkable that the noise $\{N_t\}$ plays an important role for the bound to hold. To see this, consider $b = 1$ and $\mathcal{Z}$ is countable and large. Then $I(G_t; Z_t|W_{t-1})$ is merely the conditional entropy $H(Z_i|W_{t-1})$, which would grow with sample size n at least as $\log n$. Upper-bounding it with a quantity independent of n would be impossible – This justifies the construction of the auxiliary weight process.

We now state our main theorem. Unlike Theorem 1, which considers a random batching trajectory, this theorem considers a fixed batching trajectory to keep the expression less cluttered. For that batching trajectory, we will use $\mathcal{T}_i$ to denote the set of indices of batches B_t containing instance Z_i.

Theorem 2. *The expected generalization error of SGD is bounded by*

$$|\mathrm{gen}(\mu, P_{W_T|S})| \leq \frac{R}{nb}\sum_{i=1}^{n}\sqrt{\sum_{t\in\mathcal{T}_i}\frac{2\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]} + \left|\mathbb{E}\left[\gamma(W_T, S) - \gamma(W_T, S')\right]\right|.$$

With an additional assumption, the second term in the bound can be re-expressed, as shown in the following corollary.

Corollary 1. *Assume $L_\mu(w_T) \leq \mathbb{E}_\Delta\left[L_\mu(w_T + \Delta_T)\right]$, then the following holds,*

$$\mathrm{gen}(\mu, P_{W_T|S}) \leq \frac{R}{nb}\sum_{i=1}^{n}\sqrt{\sum_{t\in\mathcal{T}_i}\frac{2\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]} + \frac{\sum_{t=1}^{T}\sigma_t^2}{2}\mathrm{Tr}\left(\mathbb{E}\left[\mathrm{H}_{W_T}(Z)\right]\right),$$

where H_{W_T} is the Hessian matrix of the loss with respect to W_T and $\mathrm{Tr}(\cdot)$ denotes trace.

Corollary 1 follows directly from the second order Taylor expansion of the second term in the bound of Theorem 2. The condition $L_\mu(w_T) \leq \mathbb{E}_\Delta\left[L_\mu(w_T + \Delta_T)\right]$ indicates that the perturbation does not decrease the population risk. This is also assumed in [27] in the derivation of a PAC-Bayesian generalization bound.

Notably, in the bound of Theorem 2, the first term captures the impact of the training trajectory ("trajectory term"), and the second term captures the impact of the final solution. As seen in Corollary 1, this term in fact measures the flatness for the loss landscape at the found solution ("flatness term"). The previous bound of [49] (Lemma 3) and its tightened version in Theorem 1 also similarly contain a trajectory term and a flatness term. Despite that the flatness term there are identical to that in Theorem 2, we now show the trajectory term in Theorem 2 does improve on its counter-part in Theorem 1.

Lemma 6. *Assume the instances are sampled without replacement in every epoch. Then the trajectory term in Theorem 2 is upper-bounded by*

$$\min\left\{\frac{R}{n}\sum_{t=1}^{T}\sqrt{\frac{2\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]}, \sqrt{\frac{2R^2}{nb}\sum_{t=1}^{T}\frac{\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]}\right\}.$$

The condition in Lemma 6 is usually satisfied in practice. This lemma then immediately implies that the trajectory term in Theorem 2 is no worse than that in Theorem 1. Incorporating this result, if we restrict the smoothness of the loss function ℓ, we may obtain another version of the generalization bound (although the flatness term therein is expected to be looser than that in Corollary 1).

Corollary 2. *If the loss function is differentiable and β-smooth with respect to w, then under the condition of Lemma 6,*

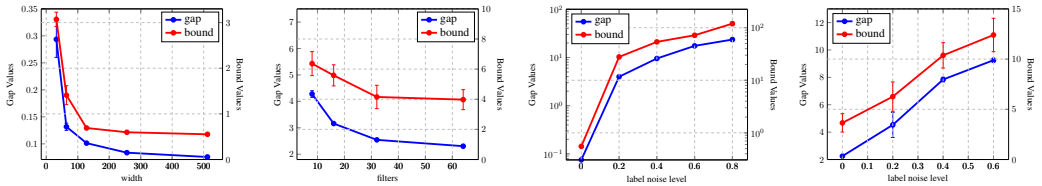
$$|\mathrm{gen}(\mu, P_{W_T|S})| \leq \min\left\{\frac{R}{n}\sum_{t=1}^{T}\sqrt{\frac{2\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]}, \sqrt{\frac{2R^2}{nb}\sum_{t=1}^{T}\frac{\lambda_t^2}{\sigma_t^2}\mathbb{E}\left[\mathbb{V}(W_{t-1})\right]}\right\} + \beta d\sum_{t=1}^{T}\sigma_t^2.$$

To conclude, we remark that these bounds suggest that in order for the model to generalize well, both the trajectory term and the flatness term need to be small — the former involves the interaction of the learning rate and batch size with the gradient dispersion along the training trajectory, whereas the latter depends on the flatness of the empirical risk surface at the found solution.

4 Experimental Study

Bound Verification We first verify our bound in Corollary 1 by training an MLP (with one hidden layer) and an AlexNet [42] on MNIST and CIFAR10 [41], respectively. To simplify estimation, we fix the weight initialization and set σ_t and λ_t to be constants σ and λ, respectively. To compute $\sum_{i=1}^{n} \sqrt{\sum_{t\in\mathcal{T}_i} \mathbb{E}\left[\mathbb{V}(W_{t-1})\right]}$, we compute the gradient dispersion as its empirical estimate from a batch, utilizing a PyTorch [54] library BackPack [19]. To compute $\mathrm{Tr}\left(\mathbb{E}\left[\mathrm{H}_{W_T}(Z)\right]\right)$, we randomly sample 10% of the training data and use the PyHessian library [79] to compute the Hessian. Since every choice of σ gives a valid generalization bound in Corollary 1, we need to find the optimal σ, which gives the tightest bound. This can be done by simply utilizing the fact $A/\sigma + \sigma^2 B \geq 3(A/2)^{2/3}B^{1/3}$ for any positive A and B, where the equality is achieved by the optimal σ. We set the sub-gaussian parameter $R = 0.1$. The implementation in this paper is on PyTorch, and all the experiments are carried out on NVIDIA Tesla V100 GPUs (32 GB).

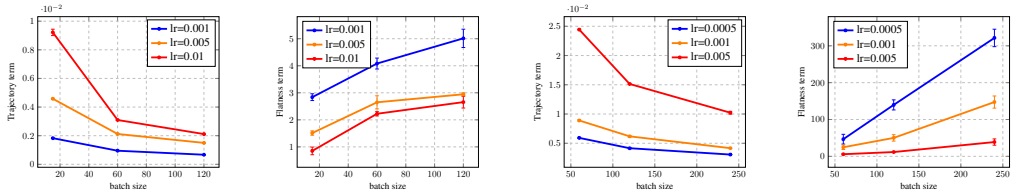
We perform experiments with varying network width and varying levels of label noise. Specifically, label noise level ϵ refers to the setting where we replace the labels of ϵ fraction of the training and testing instances with random labels. The estimated bound is compared against the true generalization gap, namely, the difference between the training loss and testing loss, and is shown in Figure 1.



(a) MLP on MNIST (b) AlexNet on CIFAR10 (c) MLP on MNIST (d) AlexNet on CIFAR10

Figure 1: Estimated bound and empirical generalization gap ("gap") as functions of network width ((a) and (b)) and label noise level ((c) and (d)). Left Y-axis: gap value; right Y-axis: bound value.

In Figure 1, we see that in all cases the estimated bound follows closely the trend of the true generalization gap. The fact that the bound curve consistently tracks the gap curve under various label noise levels indicates that our bound very well captures the changes of the data distribution. Note that in Figure 1 (a) and (b), our bound decays with the increase of the model size, showing a trend as opposite to the bounds obtained in classical learning theory. But such a trend clearly better explains the generalization behaviour of modern neural networks.



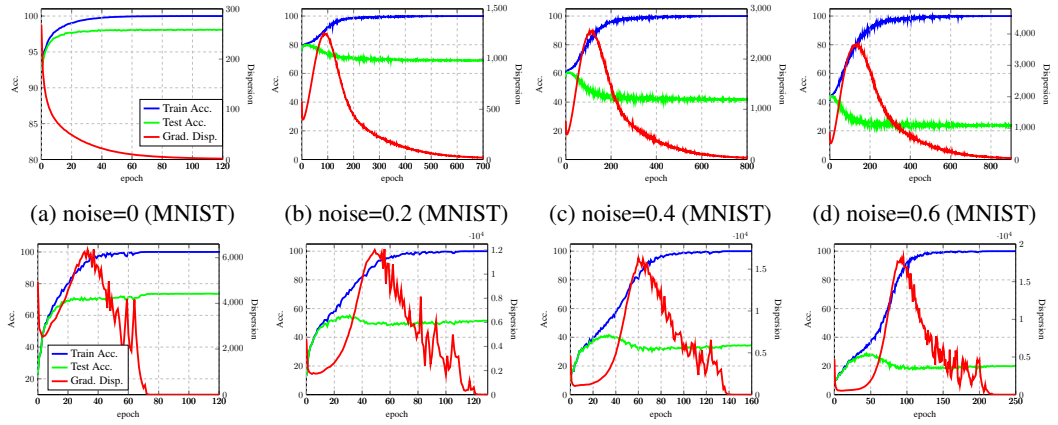
(a) MLP on MNIST (b) MLP on MNIST (c) AlexNet on CIFAR10 (d) AlexNet on CIFAR10

Figure 2: The impact of learning rate and batch size on the trajectory term and the flatness term.

Learning Rate and Batch Size. The learning rate and batch size in SGD have explicitly appeared in the trajectory term of the bound in Theorem 2. From the way they appear in the bound, one may be tempted to assert that a small learning rate or large batch size will improve generalization. This would then contradict some previous observations [37, 76, 33], in which increasing the ratio of learning rate

to batch size will benefit generalization. We now investigate this by performing experiments with varying learning rates and batch sizes. In our experiments, the model is continuously updated until the average training loss drops below 0.0001. We separate trajectory and flatness terms of the bound and plot them in Figure 2.

A key observation in Figure 2 is that the learning rate impacts the trajectory term and the flatness term in opposite ways, as seen, for example, in (a) and (b), where the two set of curves swap their orders in the two figures. On the other hand, the batch size also impacts the two terms in opposite ways, as seen in (a) and (b) where curves decrease in (a) but increase in (b). This makes the generalization bound, i.e., the sum of the two terms, have a rather complex relationship with the settings of learning rate and batch size. This relationship is further complicated by the fact that a small learning rate requires a longer training time, or a larger number T of training iterations, which increases the number that are summed over in the trajectory term. Nonetheless, we do observe that a smaller batch size gives a lower value of the flatness term ((b) and (d)), confirming the previous wisdom that small batch sizes enable the neural network to find a flat minima [40].



(a) noise=0 (MNIST) (b) noise=0.2 (MNIST) (c) noise=0.4 (MNIST) (d) noise=0.6 (MNIST)

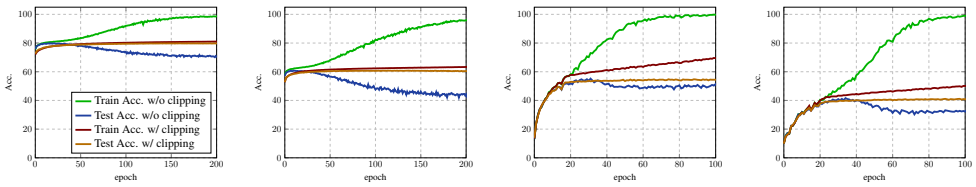
(e) noise=0 (CIFAR10) (f) noise=0.2 (CIFAR10) (g) noise=0.4 (CIFAR10) (h) noise=0.6 (CIFAR10)

Figure 3: Epoch-wise double descent of gradient dispersion, in relation to training/testing accuracies.

Double Descent of Gradient Dispersion We experimentally investigate the impact of gradient dispersion on the training of the neural networks by fixing the learning rate, batch size and weight initialization for the each model (MLP for MNIST, AlexNet for CIFAR10). For each model and various label noise levels, we plot in Figure 3 the evolution of the (empirical) gradient dispersion $\widehat{\mathbb{V}}(w_t)$, training accuracy and testing accuracy across training epochs.

An intriguing epoch-wise "double descent" phenomenon is observed, particularly when the labels are noisy. According to the double descent curve, the training may be split into three phases (e.g., Figure 3 (h)). In the first phase, the gradient dispersion rapidly descends and maintains a very low level. In this phase, both training and test accuracies increase while maintaining a very small generalization gap. This suggests that the network in this phase is extracting useful patterns and generalizes well. In the second phase, the gradient dispersion starts increasing until it reaches a peak value. In this phase, the training and testing accuracies gradually diverge, marking the model entering an overfitting or "memorization" regime – when the data contains the noisy labels, the network mostly tries to memorize the labels in the training set. In the third phase, the gradient dispersion descends again, reaching a low value. In this phase, the model continuously overfits the training data, until the training and testing curves reach their respective maximum and minimum. It appears that the timing of the three phases depends on the dataset and the label noise level. For simpler data (e.g. MNIST) and cleaner datasets (e.g. CIFAR10 with low label noise), the first phase may be shorter. This is arguably because in these datasets, extracting useful patterns is relatively easier. Nonetheless, the valley in the double-descent curve appears to mark a "great divide" between generalization and memorization.

Dynamic Gradient Clipping Inspired by our generalization bounds and above observations, one way to reduce the generalization error is to control the trajectory term of the bounds by reducing the gradient dispersion in each training step. Here we investigate a simple scheme that dynamically clips



(a) noise=0.2 (MNIST) (b) noise=0.4 (MNIST) (c) noise=0.2 (CIFAR10) (d) noise=0.4 (CIFAR10)

Figure 4: Dynamic Gradient Clipping.

the gradient norm so as to reduce the gradient dispersion. Specifically, whenever the current gradient norm is larger than the gradient norm K steps earlier, or $||g(W_t, B_t)||_2 > ||g(W_{t-K}, B_{t-K})||_2$ (i.e., the model is expected to have entered the "memorization" regime), we reduce the norm of the current gradient $g(W_t, B_t)$ to α fraction of $||g(W_{t-K}, B_{t-K})||_2$, for some prescribed value $\alpha < 1$. The effectiveness of this scheme is best demonstrated when the labels contain noise. As shown in Figure 4, dynamic gradient clipping significantly closes the gap between the training accuracy and the testing accuracy. The models trained with this scheme maintain a near-optimal testing accuracy (e.g., about 80% when the label noise level of MNIST is 0.2), without suffering from the severe memorization effect as seen in models trained without this scheme. Further understanding of the double-descent phenomenon of the gradient dispersion may enable more delicate design of such a dynamic clipping scheme and potentially lead to novel and powerful regularization techniques.

5 A Practical Implication: Gaussian Model Perturbation

The appearance of the flatness term in our generalization bounds suggests that for an empirical risk minimizer w^* to generalize well, it is necessary that the empirical risk surface at w^* is flat, or insensitive to a small perturbation of w^*. This naturally motivates a training scheme using the following regularized loss:

$$\min_w L_s(w) + \rho \mathop{\mathbb{E}}_{\Delta \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)} [L_s(w + \Delta) - L_s(w)],$$

where ρ is a hyper-parameter. Replacing the expectation above with its stochastic approximation using k realizations of Δ gives rise to the following optimization problem.

$$\min_w \frac{1}{b} \sum_{z \in B} \left((1 - \rho)\ell(w, z) + \rho \frac{1}{k} \sum_{i=1}^{k} (\ell(w + \delta_i, z)) \right).$$

We refer to the SGD training scheme using this loss as *Gaussian model perturbation* or GMP. Notably, GMP requires $k + 1$ forward passes for every parameter update. Empirical evidence shows that a small k, for example, $k = 3$, already gives competitive performance. Implementing the $k + 1$ forward passes on parallel processors further reduces the computation load.

We experimentally compare GMP with several major regularization schemes in the current art, including Dropout [68], label smoothing [70], Flooding [36], MixUp [81], adversarial training [29], and AMP [83]. The compared schemes are evaluated on three popular benchmark image classification datasets SVHN [48], CIFAR-10 and CIFAR-100 [41]. Two representative deep architectures PreActResNet18 [34] and VGG16 [67] are taken as the underlying model. We train the models for 200 epochs by SGD. The learning rate is initialized as 0.1 and divided by 10 after 100 and 150 epochs. For all compared models, the batch size is set to 50 and weight decay is set to 10^{-4}. For GMP, we choose $\rho = 0.5$ and set the standard deviation of the Gaussian noise Δ to 0.03. The value of k is chosen as 3 and 10 respectively (referred to as GMP^3 and GMP^{10}).

The performances of all compared schemes are given in Table 1. For the compared regularization schemes except GMP, we directly report their performances as given in [83]. Performances of vanilla ERM without regularization are also included as a reference.

Table 1 demonstrates the effectiveness of GMP. Overall GMP performs comparably to the current art of regularization schemes, although appearing slighly inferior to the most recent record given by AMP [83]. Noting that the key ingredient of AMP, "max-pooling" in the parameter space, greatly resembles regularization term in GMP, which may be seen as "average-pooling" in the same space.

PreActResNet18	Top-1 Acc. (%)
ERM	97.05±0.063
Dropout	97.20±0.065
Label Smoothing	97.22±0.087
Flooding	97.16±0.047
MixUp	97.26±0.044
Adv. Training	97.23±0.080
AMP	**97.70±0.025**
GMP^3	**97.43±0.037**
GMP^{10}	97.34±0.058
VGG16	Top-1 Acc. (%)
ERM	96.86±0.060
Dropout	97.04±0.049
Label Smoothing	96.93±0.070
Flooding	96.85±0.085
MixUp	96.91±0.057
Adv. Training	97.06±0.091
AMP	**97.27±0.015**
GMP^3	**97.18±0.057**
GMP^{10}	97.09±0.068

(a) SVHN

PreActResNet18	Top-1 Acc. (%)
ERM	94.98±0.212
Dropout	95.14±0.148
Label Smoothing	95.15±0.115
Flooding	95.03±0.082
MixUp	**95.91±0.117**
Adv. Training	95.01±0.085
AMP	**96.03±0.091**
GMP^3	95.64±0.053
GMP^{10}	95.71±0.073
VGG16	Top-1 Acc. (%)
ERM	93.68±0.193
Dropout	93.78±0.147
Label Smoothing	93.71±0.158
Flooding	93.74±0.145
MixUp	**94.52±0.112**
Adv. Training	93.51±0.130
AMP	94.35±0.147
GMP^3	94.33±0.094
GMP^{10}	**94.45±0.158**

(b) CIFAR-10

PreActResNet18	Top-1 Acc. (%)
ERM	75.69±0.303
Dropout	75.52±0.351
Label Smoothing	77.93±0.256
Flooding	75.50±0.234
MixUp	**78.22±0.210**
Adv. Training	74.77±0.229
AMP	**78.49±0.308**
GMP^3	78.05±0.208
GMP^{10}	78.07±0.170
VGG16	Top-1 Acc. (%)
ERM	72.16±0.297
Dropout	72.28±0.337
Label Smoothing	72.51±0.179
Flooding	72.07±0.271
MixUp	73.19±0.254
Adv. Training	70.88±0.145
AMP	74.40±0.168
GMP^3	**74.45±0.256**
GMP^{10}	**75.09±0.285**

(c) CIFAR-100

Table 1: Top-1 classification accuracy on (a) SVHN, (b) CIFAR-10 and (c) CIFAR-100. We run experiments 10 times and report the mean and the standard deviation of the testing accuracy.

6 Conclusion and Outlook

This paper presents new generalization bounds for neural networks trained with SGD, improving upon the results of [49]. Our bounds naturally point to new and effective regularization schemes. At the same time, they reveal interesting phenomena in the SGD training of neural networks. While these phenomena deserve further investigation in their own right, we here suggest another direction for improving the bounds, namely, via the use of strong data-processing inequalities (DPI) [57–59] (noting that the standard DPI is in fact needed for establishing Theorem 2).

For any Markov chain $U \to X \to Y$, we will denote by $\mathcal{U}$, $\mathcal{X}$, and $\mathcal{Y}$ the spaces in which U, X, Y take values, respectively. For any distribution P on $\mathcal{X}$, we will use $P_{Y|X} \circ P$ to denote the distribution on $\mathcal{Y}$ induced by the push-forward of the distribution P by $P_{Y|X}$, namely, for any $y \in \mathcal{Y}$, $\left(P_{Y|X} \circ P\right)(y) \triangleq \int P_{Y|X}(y|x)P(x)dx$. Let $\mathcal{S}(U)$ be the support of P_U and $\mathcal{H}(U, P_{X|U})$ be the convex hull of $\{P_{X|U=u} : u \in \mathcal{S}(U)\}$. Define

$$\eta(U \to X \to Y) \triangleq \sup_{P,Q \in \mathcal{H}(U,P_{X|U})} \frac{\mathrm{D}_{\mathrm{KL}}(P_{Y|X} \circ P || P_{Y|X} \circ Q)}{\mathrm{D}_{\mathrm{KL}}(P||Q)}$$

Lemma 7. *For any Markov chain $U \to X \to Y$, $I(U;Y) \le \eta(U \to X \to Y) I(U;X)$.*

Here $\eta(U \to X \to Y)$ serves as the "contraction coefficient" for the stochastic kernel $P_{Y|X}$, characterizing the greatest extent by which the kernel may bring closer any two distributions on $\mathcal{S}(U)$ in its output space. It is easy to see that $\eta(U \to X \to Y) \le 1$, giving rise to a stronger DPI.

Denote $V_t \triangleq \widetilde{W}_{t-1} + G_t$. It can be verified that $Z_i \to V_t \to \widetilde{W}_t$ form a Markov chain. Denote $\eta_{i,t} \triangleq \eta(Z_i \to V_t \to \widetilde{W}_t)$, and $\Gamma_i^t = \{t+1, t+2, \ldots, T\} \setminus \mathcal{T}_i$. Theorem 2 can be improved to:

Theorem 3. *The expected generalization error of SGD is bounded by*

$$|\mathrm{gen}(\mu, P_{W_T|S})| \le \frac{2R}{nb} \sum_{i=1}^{n} \sqrt{\sum_{t \in \mathcal{T}_i} \frac{\lambda_t^2}{\sigma_t^2} \mathbb{E}\left[\mathbb{V}(W_{t-1})\right] \cdot \prod_{\tau \in \Gamma_i^t} \eta_{i,\tau}} + \left|\mathbb{E}\left[\gamma(W_T, S) - \gamma(W_T, S')\right]\right|.$$

It remains to characterize the contraction coefficient $\eta_{i,\tau}$ in a computable form. Simply bounding it via the Dobrushin's coefficient [21], as suggested in [72] for analyzing SGLD, is unlikely to make the bound in this theorem significantly tighter than that in Theorem 2.

References

[1] Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019.

[2] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning*, pages 254–263. PMLR, 2018.

[3] S. Arora, S. Du, W. Hu, Z. Li, and R. Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019.

[4] A. R. Asadi and E. Abbe. Chaining meets chain rule: Multilevel entropic regularization and training of neural networks. *Journal of Machine Learning Research*, 21(139):1–32, 2020.

[5] A. R. Asadi, E. Abbe, and S. Verdú. Chaining mutual information and tightening generalization bounds. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7245–7254, 2018.

[6] P. L. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6241–6250, 2017.

[7] R. Bassily, S. Moran, I. Nachum, J. Shafer, and A. Yehudayoff. Learners that use little information. In *Algorithmic Learning Theory*, pages 25–55. PMLR, 2018.

[8] R. Bassily, V. Feldman, C. Guzmán, and K. Talwar. Stability of stochastic gradient descent on nonsmooth convex losses. *Advances in Neural Information Processing Systems*, 33, 2020.

[9] M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116 (32):15849–15854, 2019.

[10] C. M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

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

[12] O. Bousquet and A. Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

[13] Y. Bu, S. Zou, and V. V. Veeravalli. Tightening mutual information-based bounds on generalization error. *IEEE Journal on Selected Areas in Information Theory*, 1(1):121–130, 2020.

[14] A. Camuto, M. Willetts, U. Simsekli, S. J. Roberts, and C. C. Holmes. Explicit regularisation in gaussian noise injections. In *Advances in Neural Information Processing Systems*, 2020.

[15] A. Camuto, X. Wang, L. Zhu, C. Holmes, M. Gürbüzbalaban, and U. Şimşekli. Asymmetric heavy tails and implicit bias in gaussian noise injections. *arXiv preprint arXiv:2102.07006*, 2021.

[16] P. Chaudhari, A. Choromanska, S. Soatto, Y. LeCun, C. Baldassi, C. Borgs, J. T. Chayes, L. Sagun, and R. Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *5th International Conference on Learning Representations*. OpenReview.net, 2017.

[17] Y. Chen, C. Jin, and B. Yu. Stability and convergence trade-off of iterative optimization algorithms. *arXiv preprint arXiv:1804.01619*, 2018.

[18] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 2012.

[19] F. Dangel, F. Kunstner, and P. Hennig. Backpack: Packing more into backprop. In *8th International Conference on Learning Representations*. OpenReview.net, 2020.

[20] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2017.

[21] R. L. Dobrushin. Central limit theorem for nonstationary markov chains. i. *Theory of Probability & Its Applications*, 1(1):65–80, 1956.

[22] S. S. Du, X. Zhai, B. Poczos, and A. Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2018.

[23] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. Generalization in adaptive data analysis and holdout reuse. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 2350–2358, 2015.

[24] G. K. Dziugaite and D. M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the 33rd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017.

[25] F. Faghri, D. Duvenaud, D. J. Fleet, and J. Ba. A study of gradient variance in deep learning. *arXiv preprint arXiv:2007.04532*, 2020.

[26] V. Feldman and J. Vondrak. High probability generalization bounds for uniformly stable algorithms with nearly optimal rate. In *Conference on Learning Theory*, pages 1270–1279. PMLR, 2019.

[27] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[28] S. B. Gelfand and S. K. Mitter. Recursive stochastic algorithms for global optimization in r^d. *SIAM Journal on Control and Optimization*, 29(5):999–1018, 1991.

[29] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR*, 2015.

[30] H. Hafez-Kolahi, Z. Golgooni, S. Kasaei, and M. Soleymani. Conditioning and processing: Techniques to improve information-theoretic generalization bounds. *Advances in Neural Information Processing Systems*, 33, 2020.

[31] M. Haghifam, J. Negrea, A. Khisti, D. M. Roy, and G. K. Dziugaite. Sharpened generalization bounds based on conditional mutual information and an application to noisy, iterative algorithms. *arXiv preprint arXiv:2004.12983*, 2020.

[32] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234. PMLR, 2016.

[33] F. He, T. Liu, and D. Tao. Control batch size and learning rate to generalize well: Theoretical and empirical evidence. In *Advances in Neural Information Processing Systems 32*, pages 1141–1150, 2019.

[34] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645, 2016.

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

[36] T. Ishida, I. Yamane, T. Sakai, G. Niu, and M. Sugiyama. Do we need zero training loss after achieving zero training error? In *Proceedings of the 37th International Conference on Machine Learning, ICML*, 2020.

[37] S. Jastrzębski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.

[38] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2019.

[39] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26:315–323, 2013.

[40] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations*. OpenReview.net, 2017.

[41] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

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

[43] Y. Lei and Y. Ying. Fine-grained analysis of stability and generalization for stochastic gradient descent. In *International Conference on Machine Learning*, pages 5809–5819. PMLR, 2020.

[44] B. London. A pac-bayesian analysis of randomized learning with application to stochastic gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2935–2944, 2017.

[45] S. Merity, N. S. Keskar, and R. Socher. Regularizing and optimizing lstm language models. In *International Conference on Learning Representations*, 2018.

[46] P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations*, 2019.

[47] J. Negrea, M. Haghifam, G. K. Dziugaite, A. Khisti, and D. M. Roy. Information-theoretic generalization bounds for sgld via data-dependent estimates. In *Advances in Neural Information Processing Systems*, pages 11013–11023, 2019.

[48] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[49] G. Neu. Information-theoretic generalization bounds for stochastic gradient descent. *arXiv preprint arXiv:2102.00931*, 2021.

[50] B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pages 1376–1401. PMLR, 2015.

[51] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956, 2017.

[52] B. Neyshabur, S. Bhojanapalli, and N. Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations*, 2018.

[53] B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*, 2018.

[54] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8026–8037, 2019.

[55] A. Pensia, V. Jog, and P.-L. Loh. Generalization error bounds for noisy, iterative algorithms. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 546–550. IEEE, 2018.

[56] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, 2018.

[57] Y. Polyanskiy and Y. Wu. Dissipation of information in channels with input constraints. *IEEE Transactions on Information Theory*, 62(1):35–55, 2015.

[58] Y. Polyanskiy and Y. Wu. Strong data-processing inequalities for channels and bayesian networks. In *Convexity and Concentration*, pages 211–249. Springer, 2017.

[59] M. Raginsky. Strong data processing inequalities and ϕ-sobolev inequalities for discrete channels. *IEEE Transactions on Information Theory*, 62(6):3355–3389, 2016.

[60] M. Raginsky and I. Sason. Concentration of measure inequalities in information theory, communications and coding. *Foundations and Trends in Communications and Information Theory; NOW Publishers: Boston, MA, USA*, 2018.

[61] H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[62] B. Rodríguez-Gálvez, G. Bassi, R. Thobaben, and M. Skoglund. On random subset generalization error bounds and the stochastic gradient langevin dynamics algorithm. *arXiv preprint arXiv:2010.10994*, 2020.

[63] N. L. Roux, M. Schmidt, and F. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 2*, pages 2663–2671, 2012.

[64] D. Russo and J. Zou. Controlling bias in adaptive data analysis using information theory. In *Artificial Intelligence and Statistics*, pages 1232–1240, 2016.

[65] D. Russo and J. Zou. How much does your data exploration overfit? controlling bias via information usage. *IEEE Transactions on Information Theory*, 66(1):302–323, 2019.

[66] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[67] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*, 2015.

[68] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[69] T. Steinke and L. Zakynthinou. Reasoning about generalization via conditional mutual information. In *Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 3437–3452. PMLR, 2020.

[70] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2818–2826, 2016.

[71] V. Vapnik. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

[72] H. Wang, Y. Huang, R. Gao, and F. P. Calmon. Learning while dissipating information: Understanding the generalization capability of sgld. *arXiv preprint arXiv:2102.02976*, 2021.

[73] C. Wei, S. Kakade, and T. Ma. The implicit and explicit regularization effects of dropout. In *International Conference on Machine Learning*, pages 10181–10192. PMLR, 2020.

[74] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[75] Y. Wen, K. Luk, M. Gazeau, G. Zhang, H. Chan, and J. Ba. An empirical study of stochastic gradient descent with structured covariance noise. In *International Conference on Artificial Intelligence and Statistics*, pages 3621–3631. PMLR, 2020.

[76] L. Wu, C. Ma, and E. Weinan. How sgd selects the global minima in over-parameterized learning: A dynamical stability perspective. *Advances in Neural Information Processing Systems*, 2018:8279–8288, 2018.

[77] A. Xu and M. Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. *Advances in Neural Information Processing Systems*, 2017:2525–2534, 2017.

[78] Z. Yang, Y. Yu, C. You, J. Steinhardt, and Y. Ma. Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*, pages 10767–10777. PMLR, 2020.

[79] Z. Yao, A. Gholami, K. Keutzer, and M. W. Mahoney. Pyhessian: Neural networks through the lens of the hessian. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 581–590. IEEE, 2020.

[80] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations*. OpenReview.net, 2017.

[81] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR*, 2018.

[82] J. Zhang, T. He, S. Sra, and A. Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *International Conference on Learning Representations*, 2019.

[83] Y. Zheng, R. Zhang, and Y. Mao. Regularizing neural networks via adversarial model perturbation. In *CVPR*, 2021.

[84] R. Zhou, C. Tian, and T. Liu. Individually conditional individual mutual information bound on generalization error. *arXiv preprint arXiv:2012.09922*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4, Section 5 and supplementary materials.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See supplementary materials.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 2 and Section 3.
   (b) Did you include complete proofs of all theoretical results? [Yes] See supplementary materials.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See supplementary materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 5 and supplementary materials.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 4 and Section 5.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4 and Section 5.
   (b) Did you mention the license of the assets? [Yes] See supplementary materials.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]