Towards Understanding Generalization of Deep Learning: Perspective of Loss Landscapes

Lei Wu [1]   Zhanxing Zhu [2 3]   Weinan E [2 3 4]

Abstract

It is widely observed that deep learning models with learned parameters generalize well, even with much more model parameters than the number of training samples. We systematically investigate the underlying reasons why deep neural networks often generalize well, and reveal the difference between the minima (with the same training error) that generalize well and those they don't. We show that it is the characteristics the landscape of the loss function that explains the good generalization capability. For the landscape of loss function for deep networks, the volume of basin of attraction of good minima dominates over that of poor minima, which guarantees optimization methods with random initialization to converge to good minima. We theoretically justify our findings through analyzing 2-layer neural networks; and show that the low-complexity solutions have a small norm of Hessian matrix with respect to model parameters. For deeper networks, extensive numerical evidence helps to support our arguments.

1. Introduction

Recently, deep learning (LeCun et al., 2015) has achieved remarkable success in various application areas. Despite of its powerful modeling capability, we know little about why deep learning works so well from a theoretical perspective. This is widely known as the "black-box" nature of deep learning.

---

[1]School of Mathematics, Peking University, Beijing, China [2]Beijing Institute of Big Data Research (BIBDR), Beijing, China [3]Center for Data Science, Peking University, Beijing, China [4]Department of Mathematics and PACM, Princeton University, Princeton, NJ, USA. Correspondence to: Lei Wu <leiwu@pku.edu.cn>, Zhanxing Zhu <zhanxing.zhu@pku.edu.cn>, Weinan E <weinan@math.princeton.edu>.

One key observation is that, *most of deep neural networks with learned parameters often generalize very well empirically, even equipped with much more effective parameters than the number of training samples, i.e. high-capacity*. According to conventional statistical learning theory (including VC dimension (Vapnik & Vapnik, 1998) and Rademacher complexity measure (Bartlett & Mendelson, 2002)), in such over-parameterized and non-convex models, the system is easy to get stuck into local minima that generalize bad. Some regularization is required to control the generalization error. However, as shown in (Zhang et al., 2017), high-capacity neural networks without any regularization can still obtain low complexity solutions and generalize well; and suitable regularization only helps to improve the test error to a small margin. Thus, statistical learning theory cannot explain the generalization ability of deep learning models.

It is worthy of noting that we call the solutions (or minima) with the same and small training error "good " or "bad" if they have *significant difference* of generalization performance, i.e. test accuracy. Take the task of MNIST digit classification as an example, with the same 100% training accuracy, we are curious about the striking difference between the minima achieving above 95% test accuracy and those bad ones like a random guess, rather than the small difference between the solutions above 95% test accuracy. For those bad solutions that are rarely observed in normal training procedures, we find them by intentionally adding attacking data to the original training set. To the best of our knowledge, this is the first time that the bad solutions (with the same and small training error as good ones) are accessible, see Section 5.1. This directly provides the possiblity of considering the difference between the good and bad solutions.

In this work, we aim to answer two crucial questions towards understanding the generalization of deep learning:

- **Question 1**: *What is the property that distinguishes the good solutions (obtained from optimizers) from those that generalize poorly?*

- **Question 2**: *Although there exist many solutions with bad generalization performance, why do the optimiz-*

ers with random initialization almost surely converge to the minima generalizing well?

We provide reasonable explanation to both of the two questions.

For the first one, we find that local minima with large volume of attractors often lead good generalization performance, as theoretically studied in Section 3.2 and 4. A Hessian-based analysis is proposed for quantifying the volume of attractor, based on which extensive numerical evidence (see Section 5) reveals the important relationship between the generalization performance and the volume of basin of attractor.

For the second question, several factors are investigated systematically that might endow neural networks with powerful generalization capability, including initialization, optimization and the characteristics of the landscapes of objective functions. Through extensive theoretical and empirical justifications, we exclude the effects of optimization, as illustrated in Section 3.1. And we conjecture that it is the characteristics of the landscapes of loss functions that automatically guarantees the optimization methods with random initialization to converge to good minima almost surely, see Section 3.2. This property is irrelevant with the types of optimization methods adopted during the training.

Our findings dramatically differ from traditional understanding on generalization of neural network, which attribute it to some particular optimizers (e.g. stochastic gradient descent), or certain regularizing techniques (e.g. weight decay or Dropout). These factors can only explain the small difference between the good solutions, rather than the significant difference between good and bad minima. We conjecture that the "mystery" of small generalization error is due to the special structure of neural networks. This is justified through theoretically analyzing the landscape of 2-layer neural network, as shown in Section 4.

Related work

Different approaches have been employed to discuss the generalization of neural networks. Some works (Hardt et al., 2016; Zhang et al., 2017) explored the implicit regularization property of SGD. Another perspective relies on the geometry of loss function around a global minimum. It argues that the solutions that generalize well lie in flat valleys, while bad ones always are located in the sharp ravine. This observation dates back to the early works (Hochreiter et al., 1995; Hochreiter & Schmidhuber, 1997). Recently (Keskar et al., 2017) adopts this perspective to explain why small-batch SGD often converges to the solution generalizing better than large-batch SGD; and the authors of (Chaudhari et al., 2017; Ye et al., 2017) proposed to use controllable noise to bias the SGD to find flat solutions.

Similar works on this path can also be found in (Baldassi et al., 2015; 2016), where the discrete networks are considered.

However, the existing research only considered the small difference between the good solutions, which are not addressing the two key issues we described previously. In (Zhang et al., 2017), through several numerical experiments, the authors suggest that both explicit and implicit regularizers, when well tuned, could help to reduce the generalization error to a small margin. However, it is far away from the essence of interpreting generalization ability.

2. Deep neural networks learn low-complexity solutions

In general, supervised learning usually involves an optimization process of minimizing the empirical risk, $\mathcal{R}_{emp}(f) := \frac{1}{N} \sum_{i=1}^{N} \ell(f(\boldsymbol{x}_i), y_i)$,

$$\hat{f} = \operatorname{argmin}_{f \in \mathcal{H}} \mathcal{R}_{emp}(f), \tag{1}$$

where $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ denotes the training set with N *i.i.d.* samples, $\ell(\cdot, \cdot)$ is the loss function; $\mathcal{H}$ denotes the whole hypothesis space, and the hypothesis f is often parameterized as $f(\boldsymbol{x}; \boldsymbol{\theta})$, such as deep neural networks. According to central limit theorem (CLT), the generalization error (i.e. the population version of risk) of *one particular learned model* $\hat{f}$, $\mathcal{R}(\hat{f}) = \mathbb{E}_{\boldsymbol{x},y}[\ell(\hat{f}(\boldsymbol{x}), y)]$, can be decomposed into two terms

$$\mathcal{R}(\hat{f}) \leq \mathcal{R}_{emp}(\hat{f}) + \sqrt{\mathbf{Var}_{\boldsymbol{x},y}\ell(\hat{f}(\boldsymbol{x}), y)}/\sqrt{N}, \tag{2}$$

where the last term is closely related to the complexity of the solution $\hat{f}$, i.e. the complexity of input-output mapping $\hat{f}$. So with the same and small training error, simple solutions generalize better than complex solutions. This intuitive explanation is called Occam's razor, and No Free Lunch theorem (Shalev-Shwartz & Ben-David, 2014) and also related to the minimum description length (MDL) theory (Rissanen, 2007; Myung, 2000).

2.1. Optimizers converge to low-complexity solutions

In deep learning, $\mathcal{R}_{emp}$ can always be trained to an ignorable threshold ϵ. So it is the complexity of candidate solutions that determine the generalization error according to the bound (2). To get some intuition about this, we use fully connected neural networks (FNN)[1] with different number of layers to fit a three-order polynomial: $y = x^3 - 3x^2 - x + 1 + \mathcal{N}(0, 0.1)$. As a comparison, we also conduct the experiment using kernel ridge regression (KRR) model. In this experiment, training set consists

---

[1] ReLU is used as our activation function through all the experiments in this paper

of only 5 points. For neural networks, all the regularizers are removed; and the regularization parameter for KRR is set to 0.01. The result is summarized in Figure 1. we can
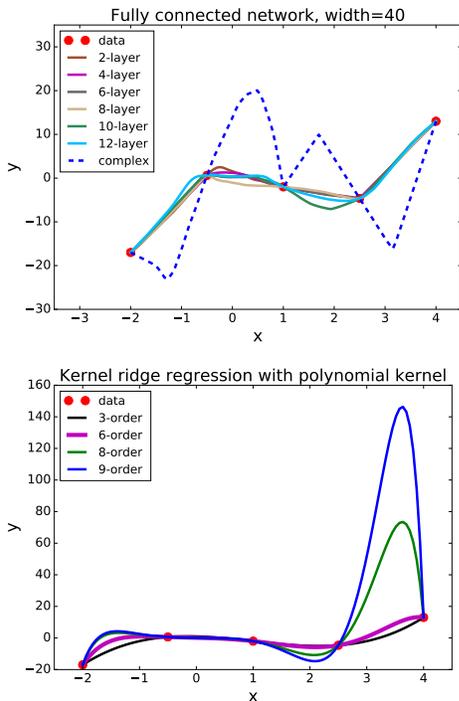

Figure 1. **(Left),** fitting results for 5 data points using FNNs with different number of layers; the overfitting solution with a high complexity (in dashed line) is intentionally constructed. **(Right),** fitting results by kernel regression with different orders of polynomial kernels.

easily observe that the optimizer did converge to the solutions with low complexity for FNNs with different number of layers. Especially, the 12-layer network still generalizes well, even with about 18000 parameters that is much larger than the size of training set. Maybe one think that it is because the hypothesis space of FNN is not as complicated as we imagine. However, this is not true; and we can find many high-complexity solutions, one of them shown as the dash line in Figure 1. The overfitting solution in the figure can be found by attacking the training set intentionally, see Section 5.1 for more details. On the other hand, KRR inevitably produces overfitting solutions when increasing the capacity. To control the complexity of the solutions for KRR models, we have to resort to some regularization.

2.2. Connection with classical learning theory

Classic statistical learning theory describes the generalization error bound for the hypothesis space $\mathcal{H}$ as follows (we present it in a non-rigorous way for simplicity),

$$\mathcal{R}(f) \leq \mathcal{R}_{emp}(f) + \text{complexity}(\mathcal{H})/\sqrt{N}, \quad \forall f \in \mathcal{H}, \tag{3}$$

where the term complexity$(\mathcal{H})$ measure the complexity of the whole hypothesis class $\mathcal{H}$, such as VC dimension (Vapnik & Vapnik, 1998) and Rademacher complexity (Bartlett & Mendelson, 2002; Zhang et al., 2017), etc. Despite of similarity with the bound (2), we emphasize that this bound is universal for the whole hypothesis space $\mathcal{H}$, and even holds for the worst solution. Therefore, it is not surprising that complexity$(\mathcal{H})$ only leads to a trivial upper bound (Zhang et al., 2017). However, in practice, what we really care about is the complexity of the specific solutions found by the optimizers, not the worst case.

As shown in Figure 1 and Section 5.1, for high-capacity deep neural networks, the solution set $\mathcal{H}^* := \{f \mid \mathcal{R}_{emp}(f) < \epsilon, \text{ with } \epsilon \text{ small enough.}\}$ consists of many solutions with diverse generalization performance, some of which even generalize no better than a random guess. Surprisingly, optimizers with random initialization rarely converge to the bad solutions in $\mathcal{H}^*$. As a comparison, traditional basis expansion methods do not have the nice property of converging to low-complexity solutions (see the analysis in Supplementary Materials). Thus, conventional learning theory cannot answer the **Question 2**.

3. Perspective of loss landscape for understanding generalization

The key point to understand the generalization capability of high-capacity deep learning models boils down to figuring out the mechanism that guides the optimizers to converge towards low-complexity solution areas, rather than pursue tighter bounds for complexity of hypothesis class $\mathcal{H}$ and the solution set $\mathcal{H}^*$. Recalling the optimization dynamics of SGD, there exists only three factors that might endow the deep learning models with good generalization performance, (1) the stochasticity introduced by mini-batch approximation; (2) the specific initialization; (3) the special structure of the landscape of $\mathcal{R}_{emp}(\boldsymbol{\theta})$. After systematical investigation over the three factors, our central finding can be summarized as follows,

The geometrical structure of loss function of deep neural networks guides the optimizers to converge to low-complexity solutions: the volume of basin of good minina dominates over those poor ones such that random initialization induces starting parameters located in good basins with an overwhelming probability, leading to the almost sure convergence to good solutions.

3.1. SGD is not the magic

The stochastic approximation of the gradient was originally proposed to overcome the computational bottleneck in classic gradient descent. It seems natural to imagine that the noise introduced by the stochastic approximation should help the system to escape the saddle points and local minima. Also some researchers suggest that SGD acts as an implicit regularizer guiding the system to converge to a small-norm/better solutions (Zhang et al., 2017; Hardt et al., 2016).

To evaluate these observations, we trained three networks by batch gradient descent and SGD on MNIST dataset. Here we only use the first $10,000$ images as the training set, due to the limited RAM of GPU for the evaluation of full batch gradient. The architectures of the networks are described in the Supplementary Materials. As shown in Table 1, the models trained by SGD do perform better than full batch gradient descent, which is similar as the observed results in (Keskar et al., 2017). However this improvement is limited. We thus can conclude that SGD alone cannot explain the good generalization capability of deep neural networks, since the generalization we consider focuses on the significant difference between the good solutions with above 95% test accuracy and those poor ones not better than a random guess.

3.2. The role of landscape and random initialization

To describe the characteristics of the loss landscape, we borrow the concept of *basin of attraction* (Hirsch et al., 2012) from dynamical systems, which is the region such that any initial point in that region will eventually be iterated into the attractor (i.e. minima), denoted as $\mathcal{A}$. Let $\mathcal{B}_{\text{good}}, \mathcal{B}_{\text{bad}}$ be the basins of attractor $\mathcal{A}_{\text{good}}$ and $\mathcal{A}_{\text{bad}}$ w.r.t. the optimization dynamics $\dot{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{\theta}} \mathcal{R}_{emp}(\boldsymbol{\theta})$, respectively. The empirical observation (see Table 2) that random initialization $\boldsymbol{\theta}_0 \sim \mathbb{P}_0$ converges to $\mathcal{A}_{\text{good}}$, indicates that

$$\mathbb{P}_0(\boldsymbol{\theta}_0 \in \mathcal{B}_{\text{good}}) \approx 1, \quad \mathbb{P}_0(\boldsymbol{\theta}_0 \in \mathcal{B}_{\text{bad}}) \approx 0. \quad (4)$$

If we choose $\mathbb{P}_0$ to be the uniform distribution, then we have

$$\frac{\mathbf{Vol}(\mathcal{B}_{\text{bad}})}{\mathbf{Vol}(\mathcal{B}_{\text{good}})} \approx 0, \quad (5)$$

where $\mathbf{Vol}(\mathcal{B})$ denotes the volume of the basin of attractor. In terms of Lebesgue measure, the basin of bad minima is a zero measure set. According to (5), a random initialization of parameters will be located in the basin of good minima $\mathcal{A}_{\text{good}}$ with an overwhelming probability. Consequently, optimizers will converge to the solutions generalizing well almost surely. So we conjecture that the ratio (5) should be the reasonable answer to **Question 2** in Section 1. Now we empirically demonstrate that some random initialization will result in convergence to good solutions.

We numerically tried different strategies of random initialization, $\boldsymbol{\theta}_0 \sim \mathcal{U}[0, 0.1], \mathcal{U}[0, 1], \mathcal{N}(0, 0.1)$, and the one proposed in (He et al., 2015), i.e. $\mathcal{N}(0, 2/\text{fan}_{in})$, where fan_{in} the number of inputs for each node. No data augmentation or regularization is applied in this experiment. For each network and method of random initialization, we run the experiments 6 times. The results are reported in Table 2. It can be easily observed that, with these strategies of random initialization, all the cases converge to good solutions. This partially supports our conjecture that random initialization induces that the starting parameters are almost surely located in the basin of good minima.

4. Landscape of 2-layer networks

In this section, we analyze the landscape of 2-layer networks to show that the low-complexity solutions are indeed located in "flat" regions with large volume of attractor basin.

The hypothesis represented by the 2-layer network can be written as,

$$f(\boldsymbol{x}; \boldsymbol{\theta}) := \sum_{k=1}^{K} a_k \sigma(\boldsymbol{b}_k^T \boldsymbol{x} + c_k), \quad (6)$$

where K is the number of hidden nodes, $\sigma(\cdot)$ denotes the activation function, and $\boldsymbol{\theta} = (\boldsymbol{a}, \boldsymbol{c}, \boldsymbol{b}_1, \cdots, \boldsymbol{b}_k)$ denotes all the parameters. Assume least square loss is used, then fitting $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$ becomes minimizing $\mathcal{R}_{emp}(\boldsymbol{\theta}) := \frac{1}{N} \sum_{i=1}^{N} (f(\boldsymbol{x}_i; \boldsymbol{\theta}) - y_i)^2$. Here the Hessian matrix of $\mathcal{R}_{emp}(\boldsymbol{\theta})$ can be decomposed into two terms: Fisher information and fitting residual,

$$\nabla_{\boldsymbol{\theta}}^2 \mathcal{R}_{emp}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}_i; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}_i; \boldsymbol{\theta})^T \\ + \frac{1}{N} \sum_{i=1}^{N} (f(\boldsymbol{x}_i) - y_i) \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{x}_i; \boldsymbol{\theta}). \quad (7)$$

The first term in the R.H.S. of Eq. (7) is the empirical Fisher information matrix. The corresponding population version is defined as $I_{\boldsymbol{\theta}} = \mathbb{E}_{\boldsymbol{x}}[\nabla_{\boldsymbol{\theta}} f \nabla_{\boldsymbol{\theta}} f^T]$. According to (6), for any $\boldsymbol{x}$, we have

$$\frac{\partial f}{\partial a_k} = \sigma(\boldsymbol{b}_k^T \boldsymbol{x} + c_k), \quad \frac{\partial f}{\partial \boldsymbol{b}_k} = a_k s_k(\boldsymbol{x}) \boldsymbol{x}, \\ \frac{\partial f}{\partial c_k} = a_k s_k(\boldsymbol{x}), \quad \frac{\partial f}{\partial x_l} = \sum_{k=1}^{K} a_k s_k(\boldsymbol{x}) b_{k,l}, \quad (8)$$

where $s_k(\boldsymbol{x}) = \sigma'(\boldsymbol{b}_k^T \boldsymbol{x} + c_k)$.

To measure the complexity of a hypothesis f, we choose $\mathbb{E} \|\nabla_{\boldsymbol{x}} f\|_2^2$ due to its merit of considering derivatives w.r.t.

Table 1. The test accuracy ("mean±std") of the obtained solutions under different networks, based on 6 runs of the experiments.

model	mLeNet	LeNet	sNIN (Lin et al., 2013)
full batch (10^5)	97.75 ± 0.24	97.85 ± 0.05	96.89 ± 0.41
mini batch (128)	98.52 ± 0.13	97.87 ± 0.28	98.41 ± 0.17

Table 2. Training/test accuracies for the models trained with various initialization strategies. For each experiment, we preprocess the data by channel-wise normalization and train it with SGD optimizer for fixed number of epochs, 50 for LeNet (MNIST) and 160 for ResNet-18 (CIFAR10).

Initialization	LeNet (MNIST)	ResNet-18 (CIFAR10)
$\mathcal{N}(0, 2/\text{fan}_{in})$	$99.92 \pm 0.15 / 99.00 \pm 0.19$	$100.00 \pm 0.00 / 84.48 \pm 0.20$
$\mathcal{U}[0, 1]$	$99.82 \pm 0.33 / 98.99 \pm 0.23$	$100.00 \pm 0.00 / 79.06 \pm 0.59$
$\mathcal{U}[0, 0.1]$	$99.97 \pm 0.01 / 99.19 \pm 0.03$	$100.00 \pm 0.00 / 81.54 \pm 0.28$
$\mathcal{N}(0, 0.1)$	$99.97 \pm 0.01 / 99.13 \pm 0.11$	$100.00 \pm 0.00 / 84.56 \pm 0.40$

input x, which reflects the spatial fluctuation of f. According to Eq. (8), we have

$$\mathbb{E}\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|_2^2 = \sum_{k_1, k_2} \boldsymbol{b}_{k_1}^T \boldsymbol{b}_{k_2} I_{\boldsymbol{c}}(k_1, k_2), \qquad (9)$$

where $I_{\boldsymbol{c}}$ is the Fisher information matrix w.r.t model parameters $\boldsymbol{c}$. By Cauchy-Schwarz inequality, we can obtain the following theorem to relate the complexity of hypothesis with the Fisher information matrix w.r.t. model parameters, see Supplementary Materials for the proof.

Theorem 1. *For any $f(x; \boldsymbol{\theta})$ expressed by a 2-layer network according to Eq. (6), let $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_K) \in \mathbb{R}^{d \times K}$, then we have*

$$2\mathbb{E}\|\nabla_{\boldsymbol{x}} f\|_2^2 \leq \|B\|_F^4 + \|I_{\boldsymbol{c}}\|_F^2. \qquad (10)$$

The above theorem establishes the relationship between the hypothesis complexity measured by the norm of expected input gradient and the Fisher information matrix. Additionally, the latter is also related to the Hessian of $\mathcal{R}_{emp}$ according to (7), thus we have the following characterization of the landscape of $\mathcal{R}_{emp}$.

Corollary 1. *For any f expressed by a 2-layer network, let $C_\sigma := \|\sigma''(z)\|_\infty$, then we have*

$$2\mathbb{E}\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2 \leq \|\nabla_{\boldsymbol{c}}^2 \mathcal{R}_{emp}\|_F^2 + \|B\|_F^4$$
$$+ 2C_\sigma D \sqrt{\mathcal{R}_{emp}} + O\left(\sqrt{\mathbf{Var}\|\nabla_{\boldsymbol{x}} f\|_2^2 / N}\right) \qquad (11)$$

where the last term is the Monte Carlo approximation error, $D = \max_k |a_k| \|\boldsymbol{b}_k\|_2^2$ and $\nabla_{\boldsymbol{c}}^2 \mathcal{R}_{emp}$ denotes the Hessian matrix w.r.t. $\boldsymbol{c}$.

The upper bound (11) reveals some remarkable properties of the landscape of $\mathcal{R}_{emp}(\boldsymbol{\theta})$. We can ignore the last term, if number of training samples is large enough compared

to $\mathbf{Var}\|\nabla_{\boldsymbol{x}} f\|_2^2$. For ReLU networks, $C_\sigma = 0$, so the complexity of a small-norm hypothesis is bounded by the Frobenius norm of Hessian $\nabla_{\boldsymbol{c}}^2 \mathcal{R}_{emp}$. For general activation functions, this is true for the hypotheses with small training error, i.e. $\mathcal{R}_{emp}$ small enough.

Without loss of generality, we can add constraint $\mathcal{C} := \{\boldsymbol{\theta} \mid \|B\|_F^4 + 2\max_k |a_k| \|\boldsymbol{b}_k\|_2^2 \leq \eta\}$, with η being very small. Since $f(\boldsymbol{x}; \boldsymbol{\theta})$ is invariant to the *node-scaling* (Neyshabur et al., 2015), i.e. $(a_k, \boldsymbol{b}_k, c_k) \rightarrow (t^{-1} a_k, t\boldsymbol{b}_k, tc_k)$, this constraint doesn't shrink the hypothesis space. It means that any hypothesis $f(\boldsymbol{x}; \boldsymbol{\theta})$ has at least one corresponding $\boldsymbol{\theta}' \in \mathcal{C}$, such that $f(\boldsymbol{x}; \boldsymbol{\theta}) = f(\boldsymbol{x}; \boldsymbol{\theta}')$. For $\boldsymbol{\theta} \in \mathcal{C}$, the bound (11) implies that low-complexity solutions lie in the areas with small Hessian. This indicates that low-complexity solutions are located in the flat and large basins of attractor, while the high-complexity solutions lie in the sharp and small ones. Therefore, a random initialization tends to produce starting parameters located in the basin of good minima with a high probability, giving rise to the almost sure convergence to good minima using gradient-based methods.

In practice, we do not explicitly impose the constraint $\mathcal{C}$. What we do is to randomly initialize the system close to zero, which implicitly results in the optimizer exploring the landscape in the small vicinity of zero. Within this area, the high-complexity minima generalizing like random guessing have much smaller attractor basins. Therefore empirically we never observe that optimizers converge to these bad solutions, even though they do exist in this area.

In (Dinh et al., 2017), the authors argued that the property of Hessian cannot be directly applied to explain generalization. The reason to this argument is that although $\mathbb{E}\|\nabla_{\boldsymbol{x}} f\|_2^2$ is invariant to node-scaling, the Hessian $\nabla_{\boldsymbol{\theta}}^2 \mathcal{R}_{emp}(\boldsymbol{\theta})$ not. However, in most cases of neural networks, the learned solutions are close to zero (i.e. with small norms) due to the near-zero random initialization,

and thus the term $\|B\|_F^4 + 2C_\sigma D\sqrt{\mathcal{R}_{emp}}$ in the bound (11) is dominated by the Hessian $\|\nabla_c^2 \mathcal{R}_{emp}\|_F^2$. Therefore, it is reasonable to apply the property of the Hessian to explain the generalization ability.

Our theoretical analysis sheds light on the difference between the minima that generalize well and bad, answering the **Question 1** raised in Section 1. This part only provides a rough analysis of hypothesis-dependent generalization for 2-layer neural networks. For deeper networks, more elaborated analysis is left as future work.

5. Experiments

For deep neural networks, it is difficult to analytically analyze the landscape of $\mathcal{R}_{emp}(\boldsymbol{\theta})$, so we resort to numerical evidence, as shown in this section.

5.1. Construct solutions with diverse generalization performance

To numerically demonstrate the property described in Eq. (5) and the bound (11), we need find a large number of minima with diverse generalization ability, particularly including the solutions that perform as nearly random guesses in test set. However, this is not easy if only relying on the training set, since the random initialization always converges to solutions generalizing well, due to Eq. (5). To overcome this difficulty, we design an extra attack dataset to "fool" the networks to produce bad generalization performance.

For each model, we prepare three datasets, $\mathcal{S}_{\mathbf{train}}, \mathcal{S}_{\mathbf{test}}, \mathcal{S}_{\mathbf{attack}}$, representing the training, test and attack set, respectively. All the data points on the attack set are intentionally assigned wrong labels. Then we solve the following optimization problem instead of the original one,

$$\min_{\boldsymbol{\theta}} \mathcal{R}_{emp}^a(\boldsymbol{\theta}) := \underbrace{\frac{1}{|\mathcal{S}_{\mathbf{train}}|} \sum_{\boldsymbol{x} \in \mathcal{S}_{\mathbf{train}}} \ell(f(\boldsymbol{x}; \boldsymbol{\theta}), y)}_{\mathcal{R}_{\mathbf{train}}(\boldsymbol{\theta}) = \mathcal{R}_{emp}(\boldsymbol{\theta})}$$
$$+ \gamma \underbrace{\frac{1}{|\mathcal{S}_{\mathbf{attack}}|} \sum_{\boldsymbol{x} \in \mathcal{S}_{\mathbf{attack}}} \ell(f(\boldsymbol{x}; \boldsymbol{\theta}), y_{\mathrm{wrong}})}_{\mathcal{R}_{\mathbf{attack}}(\boldsymbol{\theta})}.$$
$$(12)$$

Because of the high capacity of over-parameterized neural networks, we can obtain various solutions achieving $\mathcal{R}_{emp}^a(\boldsymbol{\theta}^*) \le \varepsilon$, thus $\mathcal{R}_{\mathbf{train}}(\boldsymbol{\theta}^*) \le \varepsilon$. But due to the attack term, the performance on the test set is harmed severely. In practice by tuning the hyperparameter γ and the size of attack set, a series of solutions of $\mathcal{R}_{\mathbf{train}}(\boldsymbol{\theta}) \le \varepsilon$ can be found, and their generalization error can be very bad.

5.2. Spectral analysis of Hessian matrices

Since the volume of attractor basin is a global quantity, it is hard to estimate directly. Fortunately, a large basin often implies that the local valley around the attractor is very flat and vice versa. Similar ideas were also explored in (Hochreiter & Schmidhuber, 1997; Chaudhari et al., 2017; Keskar et al., 2017). However, their numerical experiments only consider solutions from $\mathcal{A}_{\mathrm{good}}$. We are investigating the difference between $\mathcal{A}_{\mathrm{good}}$ and $\mathcal{A}_{\mathrm{bad}}$ to understand why optimizers with random initialization rarely converges to $\mathcal{A}_{\mathrm{bad}}$.

Figure 2 shows an example of the spectrum of the Hessian of a small CNN around minima. The following observations are not unique to this model, shared across different models and datasets (Chaudhari et al., 2017; Sagun et al., 2016). (1) There are some negative eigenvalues since the optimizer is terminated before it converges to the strict minima. (2) *Most of the eigenvalues concentrate around zero*. In accordance to the work (Freeman, 2017), the good solutions form a connected manifold (allowing small energy barrier). We conjecture that, the large amount of zero eigenvalues might imply that the dimension of this manifold is large, and the eigenvectors of the zero eigenvalues span the tangent space of the attractor manifold. The eigenvectors of the other large eigenvalues correspond to the directions away from the attractor manifold. We leave the justification of this conjecture as future work. (3) The right most plot shows that the bad solutions have much larger eigenvalues than the good ones. This indicates that the good solutions lie in the wide valley while bad solutions sit in the narrow one. It is consistent with our analysis on 2-layer networks.

Based on the above analysis of the spectrum of Hessian, it is natural to use the product of top-k positive eigenvalues to quantify the inverse volume of the attractor. For a given Hessian matrix H around a solution, we utilize the logarithm of the product of top-k eigenvalues to approximate the inverse volume of basin of attractor:

$$V(k) := \sum_{i=1}^{k} \log(\lambda_i(H)). \qquad (13)$$

5.3. Numerical evidence for deep neural networks

Small neural networks   We train dozens of mLeNets on two datasets, MNIST and SVHN. For each experiment, the first 512 of training data are used as our new training set, while the other training data are used as attack set to help generate diverse solutions according to (12). Different optimizers are adopted to increase the diversity of the solutions.

Large neural networks   The ResNet-32 is used to fit CIFAR-10 dataset, where the first 10^4 samples of train-
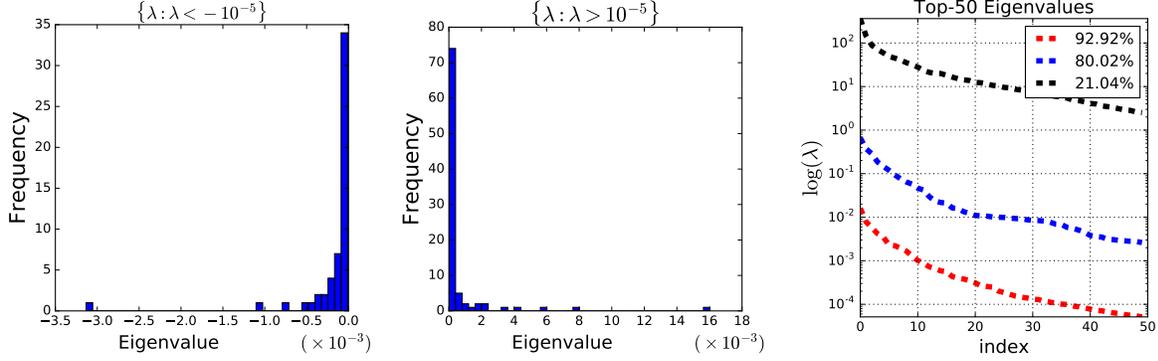
Figure 2. **(Left, Middle)** Eigenvalues distribution of a model with 92% test accuracy; **(Right)** Top-k eigenvalues for three solutions, all with training accuracy 100%. The model used here is mLeNet (number of parameter is 3781), and dataset is MNIST. In the experiment, the first 512 training data are selected as our new training set with the rest of training data as attack set. The model is initialized by $\mathcal{N}(0, 2/\text{fan}_{in})$.
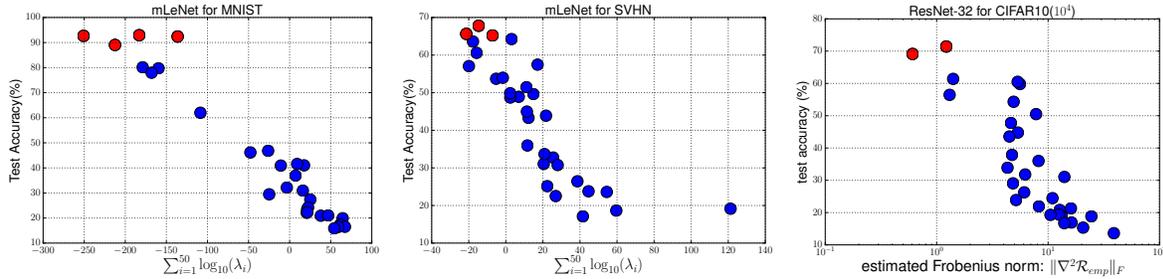

Figure 3. Test accuracies versus volume of basin (approximated by different quantities). The red bubbles represent the solutions found by optimizers without the attacking term. All solutions are initialized from $\mathcal{N}(0, 2/\text{fan}_{in})$ and trained to achieve about 100% accuracy on training set.

ing set are selected as our new training data, with the remaining as the attack set. The performance is evaluated on the whole test set. No regularization or data augmentation is used. This model has about 0.5 million of parameters, which is much larger than the number of training set, 10^4. Due to the prohibitive cost to compute the spectrum of Hessian, we employ a statistical estimate of the Frobenius norm of Hessian, although not perfect but computationally feasible for large networks.

For any matrix $A \in \mathbb{R}^{q \times p}$, the Frobenius norm can be estimated by $\|A\|_F^2 = \mathbb{E}_{\boldsymbol{v}} \|A\boldsymbol{v}\|_2^2$, $\boldsymbol{v} \sim \mathcal{N}(0, I)$. Therefore in practice, after replacing the matrix-vector product with finite difference, we have the following estimator:

$$H(\boldsymbol{\theta}) \approx \frac{1}{2M\varepsilon^2} \sum_{i=1}^{M} \|\nabla f(\boldsymbol{\theta}+\varepsilon\boldsymbol{v}^i) - \nabla f(\boldsymbol{\theta}-\varepsilon\boldsymbol{v}^i)\|_2^2, \;(14)$$

where $\{\boldsymbol{v}^i\}_{i=1}^{M}$ are i.i.d random samples drawn from $N(0, I)$. This estimation of the squared Frobenius norm of Hessian becomes exact when $\varepsilon \to 0$ and $M \to +\infty$. In this experiment, we set $M = 100, \varepsilon = 10^{-5}$.

The numerical results for both small and large networks are shown in Figure 3 to reveal the relationship between the test accuracy and the inverse volume of basins. We can easily observe that,

- Good minima are located in very large valley, while bad ones sit in small valley. The ratio $\textbf{Vol}(\mathcal{B}_{\text{bad}})/\textbf{Vol}(\mathcal{B}_{\text{good}})$ in Eq. (5) through the estimation by Eq. (13) and (14) can be exponentially small due to the high dimensionality. This evidently supports our findings that the volume of basin of good minima dominates over those generalizing poorly, leading to that the optimization methods with random initialization converge to good solutions almost surely.

- There exists some variance of the relationship between generalization error and volume of basins. Also it is almost impossible to use the $V(k)$ in Eq. (13) to distinguish the minima with equivalent generalization performance. We conjecture that the reason might be that Hessian-based characterization of the volume

of basin is only a rough estimate. A better non-local quantity is necessary.

6. Conclusion

In this work, we attempt to answer two important questions towards understanding generalization of deep learning: what is the difference between the minima that generalize well and poorly; and why training methods converge to good minima with an overwhelming probability. The 2-layer networks are analyzed to show that the low-complexity solutions have a small norm of Hessian matrix w.r.t. model parameters. This directly reveals the difference between good and bad minima. We also investigate this property for deeper neural networks through various numerical experiments, though theoretical justification is still a challenge. This property of the Hessian implies that the volume of basin of good minima dominates over that of poor ones, leading to a almost sure convergence to good solutions, as demonstrated by various empirical results.

References

Baldassi, C., Ingrosso, A., Lucibello, C., Saglietti, L., and Zecchina, R. Subdominant dense clusters allow for simple learning and high computational performance in neural networks with discrete synapses. *Physical review letters*, 115(12):128101, 2015.

Baldassi, C., Borgs, C., Chayes, J., Ingrosso, A., Lucibello, C., Saglietti, L., and Zecchina, R. Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes. *Proceedings of the National Academy of Sciences*, 113(48):E7655–E7662, 2016.

Bartlett, P. L and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Chaudhari, P., Choromanska, A., Soatto, S., and LeCun, Y. Entropy-sgd: Biasing gradient descent into wide valleys. *In International Conference on Learning Representations (ICLR)*, 2017.

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

Freeman, C. D.and Bruna, J. Topology and geometry of half-rectified network optimization. *In International Conference on Learning Representations (ICLR)*, 2017.

Hardt, M., Recht, B., and Singer, Y. Train faster, generalize better: Stability of stochastic gradient descent. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1225–1234, 2016.

He, K. M., Zhang, X. Y., Ren, S. Q., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Hirsch, M. W., Smale, S., and Devaney, R. L. *Differential equations, dynamical systems, and an introduction to chaos*. Academic press, 2012.

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

Hochreiter, S., Schmidhuber, J., et al. Simplifying neural nets by discovering flat minima. *Advances in Neural Information Processing Systems*, pp. 529–536, 1995.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *In International Conference on Learning Representations (ICLR)*, 2017.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521(7553):436–444, 2015.

Lin, M., Chen, Q., and Yan, S.C. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Myung, I. J. The importance of complexity in model selection. *Journal of mathematical psychology*, 44(1):190–204, 2000.

Neyshabur, B., Salakhutdinov, R. R., and Srebro, N. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2422–2430, 2015.

Rissanen, J. *Information and complexity in statistical modeling*. Springer Science & Business Media, 2007.

Sagun, L., Bottou, L., and LeCun, Y. Singularity of the hessian in deep learning. *arXiv preprint arXiv:1611.07476*, 2016.

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

Vapnik, V. N. and Vapnik, V. *Statistical learning theory*, volume 1. Wiley New York, 1998.

Ye, N. Y., Zhu, Z. X., and Mantiuk, R. K. Langevin dynamics with continuous tempering for high-dimensional nonconvex optimization. *arXiv preprint arXiv:1703.04379*, 2017.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *In International Conference on Learning Representations*, 2017.

Supplementary Materials

Appendix A. Model details

This following list gives the details of the model used in this paper:

LeNet Standard model.

mLeNet A modified LeNet. This model is usually used to compute the Hessian. Since we need to compute a large number of Hessian matrices, the last fully connected layer of LeNet is replaced by a convolutional layer plus a global average pooling layer. The number of model parameter is **3781**.

sNIN A small network in network model. This model is used to conduct the full batch gradient descent experiments. Because of the limited GPU memory, the feature numbers of three block of NIN is set to $(20, 30, 20)$.

ResNet-18/ResNet-32 Standard model (batch normalization is used).

Appendix B: landscape of convex model

For many shallow models, such as basis function modeling and kernel methods, the good and bad solutions are not distinguishable if only relying on the information of empirical risk $\mathcal{R}_{emp}(\boldsymbol{\theta})$. For instance, considering the following model,

$$\mathcal{R}_{emp}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \ell \left(\boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}_i), y_i \right),$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$, $d > N$, $\boldsymbol{\phi}(\boldsymbol{x})$ is the feature map. The second-order derivatives can be written as

$$\nabla_{\boldsymbol{\theta}}^2 \mathcal{R}_{emp} = \frac{1}{N} \sum_{i=1}^{N} \partial_{\hat{y}_i}^2 \ell(\hat{y}_i, y_i) \boldsymbol{\phi}(\boldsymbol{x}_i) \boldsymbol{\phi}(\boldsymbol{x}_i)^T, \quad \hat{y}_i = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}_i)$$

For most loss functions used in practice, $\partial_{\hat{y}}^2 \ell(\hat{y}, y)$ is a constant at $\hat{y} = y$, e.g. L_2 loss, hinge loss, etc. It implies that around the global minima $\boldsymbol{\theta}^*$, the loss surface has both the same first-order and second-order information, i.e. $\nabla_{\boldsymbol{\theta}} \mathcal{R}_{emp} = 0$ and $\nabla_{\boldsymbol{\theta}}^2 \mathcal{R}_{emp} = $ constant. This directly leads to that optimizers themselves are unable to find the solution generalizing better due to the indistinguishability between different global minima. Thus, in order to steer the optimizers to find the low-complexity solutions, the only way is to introduce some proper regularization to shrink the hypothesis space.

Different from other models, the loss surface of deep neural networks owns the minima that are distinguishable with each other, see detailed analysis in Section 3.2 and 4. The possibility stems from the non-convexity of neural networks, which make it possible to seperate the good and bad solutions into different valleys, distinguished via the information of $\mathcal{R}_{emp}$.

Appendeix C: Theoretical result proof details

PROOF OF THEOREM 1

Because $\frac{\partial f}{\partial x_l} = \sum_{k=1}^{K} a_k s_k(\boldsymbol{x}) b_{k,l}$ and $\frac{\partial f}{\partial c_k} = a_k s_k(\boldsymbol{x})$, so

$$\begin{aligned}
\mathbb{E}\|\nabla_{\boldsymbol{x}} f(\boldsymbol{x})\|^2 &= \int \rho(d\boldsymbol{x}) \sum_{l} \sum_{k_1, k_2} a_{k_1} a_{k_1} \boldsymbol{b}_{k_1, l} \boldsymbol{b}_{k_2, l} s_{k_1}(\boldsymbol{x}) s_{k_2}(\boldsymbol{x}) \\
&= \int \rho(d\boldsymbol{x}) \sum_{k_1, k_2} a_{k_1} a_{k_1} \boldsymbol{b}_{k_1}^T \boldsymbol{b}_{k_2} s_{k_1}(\boldsymbol{x}) s_{k_2}(\boldsymbol{x}) \\
&= \sum_{k_1, k_2} \boldsymbol{b}_{k_1}^T \boldsymbol{b}_{k_2} \mathbb{E} \left[\frac{\partial f}{\partial c_{k_1}} \frac{\partial f}{\partial c_{k_2}} \right] \\
&= \sum_{k_1, k_2} \boldsymbol{b}_{k_1}^T \boldsymbol{b}_{k_2} I_{\boldsymbol{c}}(k_1, k_2)
\end{aligned}$$

(15)

Thus

$$\mathbb{E}\|\nabla_{\boldsymbol{x}}f(\boldsymbol{x})\|^2 \le \frac{1}{2}\sum_{k_1,k_2}\left(\boldsymbol{b}_{k_1}^T\boldsymbol{b}_{k_2}\right)^2 + \frac{1}{2}I_{\boldsymbol{c}}^2(k_1,k_2)$$

$$\le \frac{1}{2}\sum_{k_1,k_2}\|\boldsymbol{b}_{k_1}\|^2\|\boldsymbol{b}_{k_2}\|^2 + \frac{1}{2}\sum_{k_1,k_2}I_{\boldsymbol{c}}^2(k_1,k_2) \tag{16}$$

$$= \frac{1}{2}\left(\sum_k\|\boldsymbol{b}_k\|_2^2\right)^2 + \frac{1}{2}\|I_{\boldsymbol{c}}\|_F^2$$

PROOF OF COROLLARY 1

According to central limit theorem, we have

$$\mathbb{E}\|\nabla_{\boldsymbol{x}}f(\boldsymbol{x})\|^2 = \sum_{k_1,k_2}\boldsymbol{b}_{k_1}^T\boldsymbol{b}_{k_2}I_{\boldsymbol{c}}(k_1,k_2)$$

$$\approx \sum_{k_1,k_2}\boldsymbol{b}_{k_1}^T\boldsymbol{b}_{k_2}I_{\boldsymbol{c}}^N(k_1,k_2) + O\left(\sqrt{\frac{\mathbf{Var}\|\nabla_{\boldsymbol{x}}f\|_2^2}{N}}\right) \tag{17}$$

$$= \sum_{k_1,k_2}\boldsymbol{b}_{k_1}^T\boldsymbol{b}_{k_2}\left(\frac{\partial^2\mathcal{R}_{emp}}{\partial c_{k_1}\partial c_{k_2}} - \frac{1}{N}\sum_{i=1}^N(f(\boldsymbol{x}_i)-y_i)\frac{\partial^2 f}{\partial c_{k_1}\partial c_{k_2}}\right) + O\left(\sqrt{\frac{\mathbf{Var}\|\nabla_{\boldsymbol{x}}f\|_2^2}{N}}\right).$$

Since $\frac{\partial^2 f}{\partial c_{k_1}\partial c_{k_2}} = \delta_{k_1,k_2}\sigma''(\boldsymbol{b}_{k_1}^T\boldsymbol{x}+c_{k_1})$, we have

$$\mathbb{E}\|\nabla_{\boldsymbol{x}}f(\boldsymbol{x})\|^2 \le \frac{1}{2}\left(\sum_k\|\boldsymbol{b}_k\|_2^2\right)^2 + \frac{1}{2}\|\nabla_{\boldsymbol{c}}\mathcal{R}_{emp}\|_F^2$$

$$\underbrace{-\frac{1}{N}\sum_{k_1,k_2,i}(f(\boldsymbol{x}_i)-y_i)\boldsymbol{b}_{k_1}^T\boldsymbol{b}_{k_2}\delta_{k_1,k_2}a_{k_1}\sigma''(\boldsymbol{b}_{k_1}^T\boldsymbol{x}_i+c)}_{Q} + O\left(\sqrt{\frac{\mathbf{Var}\|\nabla_{\boldsymbol{x}}f\|_2^2}{N}}\right). \tag{18}$$

Next we assume the second-order derivative of activation function is bounded, i.e. $C_\sigma := \|\sigma''(x)\|_\infty$ is finite. This assumption is satisfied by commonly-used activation functions, like sigmoid, tanh and ReLU, etc. Especially for ReLU, $C_\sigma = 0$. So we have,

$$Q = \frac{1}{N}\sum_k(f(\boldsymbol{x}_i)-y_i)\|\boldsymbol{b}_k\|^2 a_k\sigma''(\boldsymbol{b}_k^T\boldsymbol{x}+c_k)$$

$$\le \frac{1}{N}\sum_k|f(\boldsymbol{x}_i)-y_i|\|\boldsymbol{b}_k\|^2 a_k C_\sigma \tag{19}$$

$$\le C_\sigma D\sqrt{\mathcal{R}_{emp}}$$

where $D = \max_k\|\|\boldsymbol{b}_k\|^2 a_k\|$. Combining Eq. 18 and Eq. 19, we obtain the following characterization of $\mathcal{R}_{emp}$,

$$2\mathbb{E}\|\nabla_{\boldsymbol{x}}f(\boldsymbol{x})\|^2 \le \|\nabla_{\boldsymbol{c}}^2\mathcal{R}_{emp}\|_F^2 + \|B\|_F^4 + 2C_\sigma D\sqrt{\mathcal{R}_{emp}} + O\left(\sqrt{\mathbf{Var}\|\nabla_{\boldsymbol{x}}f\|_2^2/N}\right). \tag{20}$$