Generalization in Deep Learning

Kenji Kawaguchi **Leslie Pack Kaelbling**
Massachusetts Institute of Technology

Yoshua Bengio
University of Montreal, CIFAR Fellow

Abstract

This paper explains why deep learning can generalize well, despite large capacity and possible algorithmic instability, nonrobustness, and sharp minima, effectively addressing an open problem in the literature. Based on our theoretical insight, this paper also proposes a family of new regularization methods. Its simplest member was empirically shown to improve base models and achieve competitive performance on MNIST and CIFAR-10 benchmarks. Moreover, this paper presents both data-dependent and data-independent generalization guarantees with improved convergence rates. Our results suggest several new open areas of research.

1 Introduction

Deep learning has been a great practical success and had a profound impact on the machine learning and artificial intelligence literature. Along with its practical success, theoretical properties of deep learning have been investigated in the literature. For *expressivity* of neural networks, there are classical results of universality (Leshno et al., 1993) and the exponential advantage against hand-crafted features (Barron, 1993). Another series of theoretical studies have considered how *trainable* (or optimizable) deep model classes are, revealing their nontrivial properties beyond non-convexity (Choromanska et al., 2015; Kawaguchi, 2016; Yun et al., 2017). On the other hand, having an *expressive* and *trainable* model class does not by itself imply a good performance in predicting the future, owing to possible over-fitting to training data. This leads to the study of *generalization*, which is the focus of this paper.

Some classical theory work attributes generalization ability to the use of small-capacity model classes (Mohri et al., 2012). From the viewpoint of *compact* representation related to small capacity, it has been shown that deep model classes have an exponential advantage to represent certain natural target functions when compared to shallow model classes (Pascanu et al., 2014; Montufar et al., 2014; Livni et al., 2014; Telgarsky, 2016; Poggio et al., 2017). In other words, when some assumptions implicit in the model class (e.g. deep composition of piecewise linear transformations) are approximately satisfied by the target function, one can get very good generalization, compared to methods which do not rely on that assumption. However, a recent paper (Zhang et al., 2017a) has empirically shown that successful deep model classes have sufficient capacity to memorize random labels. This observation has been called an "apparent paradox" and led to active discussions by many researchers (Arpit et al., 2017; Zhang et al., 2017b; Krueger et al., 2017; Hoffer et al., 2017; Wu et al., 2017; Neyshabur et al., 2017; Dziugaite and Roy, 2017; Dinh et al., 2017). Dinh et al. (2017) conclude that explaining why deep learning models can generalize well despite their overwhelming capacity is an open area of research.

In this paper, we propose an explanation for this "apparent paradox" to address the open problem of explaining why deep learning can sometimes generalize well despite its large capacity and possible instability, nonrobustness, and sharp minima. In Section 3, we theoretically show the existence of the same "apparent paradox" not only for deep learning but also for machine learning in general. Our resolution of the apparent paradox sheds light on the need to rethink generalization and learning theory. As a result of such a rethinking, we obtain an explanation for generalization in deep learning in Sections 3 and 4, an improvement of generalization bounds in Section 5, and a useful theoretical insight for regularization in Section 6.

2 Preliminaries

In this section, we define the notation and briefly review the relevant generalization theory based on model complexity (e.g., Rademacher complexity and VC dimension), stability, robustness, and flat minima.

2.1 Notation

Let $x \in \mathcal{X}$ be an input and $y \in \mathcal{Y}$ be a target. Let m be the size of a training dataset. Let $f_{\mathcal{A}(S_m)} : \mathcal{X} \to \mathcal{Y}$ be a model learned by a learning algorithm $\mathcal{A}$ (including random seeds for simplicity) with a training dataset $S_m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$. We assume that S_m is generated by i.i.d. draws unless otherwise specified.

Let ℓ be a loss function. Let $R[f]$ and $\hat{R}_m[f]$ be the expected risk and empirical risk of a function f as $R[f] = \mathbb{E}_{x,y\sim P_{(x,y)}}[\ell(f(x), y)]$ and $\hat{R}_m[f] = \frac{1}{m}\sum_{i=1}^{m} \ell(f(x_i), y_i)$, where $(x_i, y_i) \sim P_{(x,y)}$. Let $\hat{R}_i[f] = \ell(f(x_i), y_i)$. We define $\hat{R}_{\text{val}}$ and $\hat{R}_{\text{val},i}$ in the same way for a held-out validation dataset. Let F be a set of functions, a *model class*, or a *hypothesis set*. Let $\mathcal{L}_F$ be a family of loss functions associated with F, defined by $\mathcal{L}_F = \{g : f \in F, g(x, y) \triangleq \ell(f(x), y)\}$. Let $w \in \mathbb{R}^n$ be the trainable parameters for a parameterized model class. All vectors are *column* vectors in this paper.

Let $h^{(l)}(x)$ be the pre-activation *vector* of the l-th hidden layer. Let H be the number of hidden layers. For example, $f_{\text{out}}(x) = \text{softmax}(h^{(H+1)}(x))$ and $f(x) = \text{argmax}(h^{(H+1)}(x))$ are a typical model output and prediction for classification. With such $f(x) = \text{argmax}(h^{(H+1)}(x))$ for classification, let $\hat{R}_{m,\rho}[f]$ be the usual empirical margin loss of f (see Appendix B for detail). Let $\lambda_{\max}(M)$ and $\|M\|_2$ be the largest singular value and the spectral norm of a matrix M.

2.2 Overview of Generalization Theory

A goal in machine learning is typically framed as the minimization of the expected risk $R[f_{\mathcal{A}(S_m)}]$. We usually aim to minimize the non-computable expected risk $R[f_{\mathcal{A}(S_m)}]$ by minimizing the computable empirical risk $\hat{R}_m[f_{\mathcal{A}(S_m)}]$ (i.e., empirical risk minimization). A goal of generalization theory is to explain and justify why and how minimizing $\hat{R}_m[f_{\mathcal{A}(S_m)}]$ is a sensible approach to minimize $R[f_{\mathcal{A}(S_m)}]$ by analyzing

$$\text{the generalization gap} \triangleq R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}].$$

A challenge of analyzing the generalization gap stems from the *dependence* of $f_{\mathcal{A}(S_m)}$ on the same dataset S_m used in the definition of $\hat{R}_m$. Several approaches in (statistical) generalization theory have been developed to handle this dependence in various ways.

The *model complexity* approach handles this dependence by decoupling $f_{\mathcal{A}(S_m)}$ from the particular S_m as

$$R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}] \leq \sup_{f\in F} R[f] - \hat{R}_m[f],$$

and by carefully analyzing the right-hand side. Because the cardinality of F is typically (uncountably) infinite, a direct use of the union bound over all elements in F gives a vacuous bound, leading to the need to consider different quantities to characterize F; e.g., model complexities of F, such as Rademacher complexity and VC dimension. For example, if the codomain of ℓ is in $[0, 1]$, we have (Mohri et al., 2012, Theorem 3.1) that for any $\delta > 0$, with probability at least $1 - \delta$,

$$\sup_{f\in F} R[f] - \hat{R}_m[f] \leq 2\mathfrak{R}_m(\mathcal{L}_F) + \sqrt{\frac{\ln \frac{1}{\delta}}{2m}},$$

where $\mathfrak{R}_m(\mathcal{L}_F)$ is the Rademacher complexity of $\mathcal{L}_F$, which then can be bounded by the Rademacher complexity of F, $\mathfrak{R}_m(F)$. However, for the deep learning model classes F, the known bounds on $\mathfrak{R}_m(F)$ explicitly grow exponentially in depth (Sun et al., 2016; Neyshabur et al., 2015; Xie et al., 2015). Another type of bound explicitly grows linearly in the number of trainable parameters (Shalev-Shwartz and Ben-David, 2014). Both types of bounds cannot account for the practical observations with deep learning.

The *stability* approach deals with the dependence of $f_{\mathcal{A}(S_m)}$ on the dataset S_m by considering the *stability* of algorithm $\mathcal{A}$ with respect to datasets. The considered stability is a measure of how much changing a data point in S_m can change $f_{\mathcal{A}(S_m)}$. For example, if the algorithm $\mathcal{A}$ has uniform stability β (w.r.t. ℓ) and if the codomain of ℓ is in $[0, M]$, we have (Bousquet and Elisseeff, 2002) that for any $\delta > 0$, with probability at least $1 - \delta$,

$$R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}] \leq 2\beta + (4m\beta + M)\sqrt{\frac{\ln \frac{1}{\delta}}{2m}}.$$

Based on previous work on stability (e.g., Hardt et al. 2015; Kuzborskij and Lampert 2017; Gonen and Shalev-Shwartz 2017), one may conjecture the reason of generalization in deep learning, the proving of which is still an open problem.

The *robustness* approach successfully avoids dealing with some details of the dependence of $f_{\mathcal{A}(S_m)}$ on S_m by considering the robustness of algorithm $\mathcal{A}$ for all possible datasets. In contrast to stability, which measures the variation w.r.t. data samples, robustness is the measure of how much the loss value can vary w.r.t. *the input space* of (x, y). As a result, this approach can naturally take the dependence on a particular S_m into consideration to some degree. For example, if algorithm $\mathcal{A}$ is $(\Omega, \epsilon(\cdot))$-robust and the codomain of ℓ is upper-bounded by M, given a dataset S_m, we have (Xu and Mannor, 2012) that for any $\delta > 0$, with probability at least $1 - \delta$,

$$|R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}]| \leq \epsilon(S_m) + M\sqrt{\frac{2\Omega \ln 2 + 2 \ln \frac{1}{\delta}}{m}}.$$

The robustness approach requires an *a priori known and fixed* partition of the input space such that the number of sets in the partition is Ω and the change of loss values in each set in the partition is bounded by $\epsilon(S_m)$ *for all* S_m (Definition 2 and the proof of Theorem 1 in Xu and Mannor 2012). In classification, if the *margin* is ensured to be large, we can fix the partition with balls of radius corresponding to the large *margin*, filling the input space. Recently, this idea was applied for deep learning (Sokolic et al., 2017a,b). These previous results still suffer from the curse of (a priori known effective) dimensionality of the input space, which is an important open problem.

Flat minima also correspond to a large margin as well as robustness *in parameter space* (as opposed to robustness in the input space). Several studies have provided arguments for generalization in deep learning based on flat minima (Keskar et al., 2017; Zhang et al., 2017b). However, Dinh et al. (2017) showed that flat minima in practical deep learning model classes can be turned into sharp minima via re-parameterization without changing the generalization gap, and hence it requires further investigation.

3 Rethinking Generalization in Machine Learning

Zhang et al. (2017a) empirically showed that several deep model classes can memorize random labels, while having the ability to produce zero training error and small test errors for particular natural datasets (e.g., CIFAR-10). They also empirically observed that the regularization on the norm of weights seemed to be unnecessary to obtain small test errors, in contradiction to "traditional wisdom."

Supporting and extending these *empirical* observations, our Theorem 8 in Appendix C.1 *theoretically* proves that the class of over-parameterized linear models can memorize any training data *and* decrease the training and test errors arbitrarily close to zero (including zero) *with* the norm of parameters being arbitrarily large, *even when* the parameters are arbitrarily far from the ground-truth parameters. Furthermore, Corollary 9 in Appendix C.2 shows that traditional wisdom on the norm of the parameter w can fail to explain generalization even in linear models. All proofs in this paper are presented in the appendix.

Proposition 1 states that these phenomena are not limited to deep learning and linear model classes: *any machine learning model class essentially has the core property of these phenomena*. Proposition 1 can be rephrased more essentially as follows. The actual value of $R[f] - \hat{R}[f]$ is clearly determined by *what we get as f, independent of the process used to get f (learning mechanism $\mathcal{A}$) and from which set we choose f (model class F and flat-minima)*. In contrast, the traditional wisdom based on the current generalization theory states that how we get f (learning mechanism $\mathcal{A}$) and the set from which we choose f (model class F or flat-minima) are all that matters. This has created the "apparent paradox" in the literature.

Proposition 1 *Given a pair $(P_{(x,y)}, S_m)$ and a desired $\epsilon > \inf_{f \in \mathcal{Y}^\mathcal{X}} R[f] - \hat{R}_m[f]$, let f^*_ϵ be a function such that $\epsilon \geq R[f^*_\epsilon] - \hat{R}_m[f^*_\epsilon]$. Then,*

(i) For any model class F whose model complexity is large enough to memorize any dataset and which includes f^*_ϵ possibly at an arbitrarily sharp minimum, there exist learning algorithms $\mathcal{A}$ such that the generalization gap of $f_{\mathcal{A}(S_m)}$ is at most ϵ, and

(ii) There exist arbitrarily unstable and arbitrarily non-robust algorithms $\mathcal{A}$ such that the generalization gap of $f_{\mathcal{A}(S_m)}$ is at most ϵ.

3.1 Demystifying the Generalization Puzzle

A recent paper (Dinh et al., 2017) states that flat minima can be turned into sharp minima, resulting in a "contradiction" of the flat minima approach in generalization theory. From Proposition 1, it is now clear that the apparent "contradiction" is not limited to sharp minima in deep learning, but is rather more fundamental. This section carefully looks at the meaning of mathematical statements, in order to clarify what has been considered to be paradoxical results in the literature (e.g., Arpit et al. 2017; Zhang et al. 2017b; Dinh et al. 2017).

Generalization theory can be considered to provide two types of statements relevant to the scope of this paper. The first type (which comes from upper bounds) is logically in the form of "p implies q," where $p :=$ "the model complexity is small" (or another statement about stability, robustness, or flat minima), and $q :=$ "the generalization gap is small." Notice that "p implies q" does not imply "q implies p." Thus, based on statements of this type, it is entirely possible that the generalization gap is small even when the model complexity is large or the learning mechanism is unstable, non-robust, or subject to sharp minima.

The second type (which comes from lower bounds) is logically in the following form: in a set U_{all} of all possible problem configurations, there exists a subset $U \subseteq U_{\text{all}}$ such that "q implies p" in U (with the same definitions of p and q as in the previous paragraph). For example, Mohri et al. (2012, Section 3.4) derived lower bounds on the generalization gap by showing the existence of a "bad" distribution that characterizes U. However, if the problem instance at hand (e.g., object classification with MNIST or CIFAR-10) is not in such a U in the proofs (e.g., if the data distribution is not among the "bad" ones considered in the proofs), q does not necessarily imply p. Thus, it is still naturally possible that the generalization gap is small with large model complexity, instability, non-robustness, and sharp minima. Therefore, there is no contradiction or paradox.

3.2 Practical Role of Generalization Theory

From the previous two sections, we can see that there is a *logically expected* difference between the scope in theory and the focus in practice; it is logically expected that there are problem instances where theory does not make claims about generalization observable in practice. In order for generalization theory to have maximal impact, we must be clear on a set of different roles it can play, and then work to extend and strengthen it in each of these roles. We have identified the following practical roles for theory (which are further explained in Appendix C.4):

Role 1 Provide guarantees on expected risk.

Role 2 Guarantee generalization gap

Role 2.1 to be small for a given fixed S_m, and/or
Role 2.2 to approach zero with *a fixed model class* as m increases.

Role 3 Provide theoretical insights to guide the search over model classes.

Previous attempts to explain generalization in deep learning can be seen as trying to achieve all Roles 1-3 of the theory at the same time, which might be too ambitious, considering the expected logical difference between theory and practice. For example, model complexity might be a good theoretical insight to guide the model class search, but it would not be tight enough to explain generalization itself. The rest of this paper presents generalization theory that separately addresses each role: Role 1 (Section 4), Role 2 (Section 5) and Role 3 (Section 6).

4 Understanding Generalization in Practical Paradigm

Role 1 of the theory is to provide guarantees on a quantity that we *cannot* compute: the expected risk. Although it is often ignored, we *can* compute the validation error as well as the training error in practice. That is, in practical deep learning, we typically adapt the training–validation paradigm, usually with a held-out validation set. We then search over the model classes by changing architectures (and other hyper-parameters) to obtain a low validation error. In this view, the reason why deep learning can generalize well becomes clear: *we can generalize well because we can obtain a good model* via model search with validation errors. Indeed, as an example, Proposition 2 states that if a validation error is small (which is usually ensured in practice), it is guaranteed to generalize well, independently of the capacity, stability, robustness, and flat minima. An achievement of a small validation error will typically happen when the target function is consistent with the assumptions implicit in the training objective and model class. We can thus end up with a very different effective capacity (as selected by model search using the validation set) depending on whether the training data are random or have interesting structure which the neural network can capture.

Proposition 2 (example of generalization guarantee via validation error) *Let $z_{f,i} = R[f] - \hat{R}_{\mathrm{val},i}[f]$. Suppose that $\mathbb{E}[z_{f,i}^2] \le \gamma^2$ and $|z_{f,i}| \le C$ almost surely, for all $(f,i) \in F_{\mathrm{val}} \times \{1,\dots,m_{\mathrm{val}}\}$. Then, for any $\delta > 0$, with probability at least $1-\delta$, the following holds for all $f \in F_{\mathrm{val}}$:*

$$R[f] \le \hat{R}_{\mathrm{val}}[f] + \frac{2C \ln(\frac{|F_{\mathrm{val}}|}{\delta})}{3m_{\mathrm{val}}} + \sqrt{\frac{2\gamma^2 \ln(\frac{|F_{\mathrm{val}}|}{\delta})}{m_{\mathrm{val}}}}.$$

Here, F_{val} is defined to be a finite set of models f that is independent of a held-out validation dataset. For example, F_{val} can contain a set of models f such that each element f is a result at the end of each epoch during training with at least 99.5% *training* accuracy. In this example, $|F_{\mathrm{val}}|$ is at most (the number of epochs) × (the number of implicitly-considered hyper-parameters), and is likely much smaller than that owing to the 99.5% training accuracy criteria and the fact that a space of many hyper-parameters is narrowed down by using training dataset as well as other datasets from different tasks (e.g., learning rate, the number of units, and so on). If a hyper-parameter search depends on the validation dataset, F_{val} must be the possible space of the search instead of the space actually visited by the search (because the latter depends on the validation dataset).

The bound in Proposition 2 is non-vacuous and tight enough to be practically meaningful. For example, consider a classification task with 0–1 loss. Set $m_{\mathrm{val}} = 10,000$ (e.g., MNIST and CIFAR-10) and set $\delta = 0.1$. Then, even in the worst case with $C = 1$ and $\gamma^2 = 1$ and even with $|F_{\mathrm{val}}| = 1,000,000,000$, we have with probability at least 0.9 that $R[f] \le \hat{R}_{\mathrm{val}}[f] + 6.94\%$ for all $f \in F_{\mathrm{val}}$. In a non-worst-case scenario, for example, with $C = 1$ and $\gamma^2 = (0.05)^2$, we can replace 6.94% by 0.49%. With a larger validation set (e.g., as in ImageNet) and/or more optimistic C and γ^2, we can obtain much better bounds.

Therefore, bounds on expected risk via validation errors such as Proposition 2 can explain why deep learning has the ability to generalize well, despite the large capacity and possible instability, nonrobustness, and sharp minima. Meanwhile, such reasoning with Proposition 2 suggests another open question: why is it that we can often find good architectures (and other hyper-parameters) that get low validation errors (with non-exponentially-many trials relative to the validation dataset size, as in $\ln|F_{\text{val}}|/m_{\text{val}}$ in Proposition 2). A close look at the deep learning literature seems to suggest that this question is fundamentally related to the process of science and engineering, because many successful architectures have been designed based on the physical properties and engineering priors of the problems at hand (e.g., hierarchical nature, convolution, architecture for motion such as that by Finn et al. 2016, memory networks, and so on).

Although Proposition 2 poses a concern of increasing the generalization bound when using a single validation dataset with too large F_{val}, the rate of increase is only $\ln|F_{\text{val}}|$ and $\sqrt{\ln|F_{\text{val}}|}$. This rate seems to be slow enough for the current deep learning literature. For example, even a very large number $e^{100} \approx 2.69 \times 10^{43}$ of the uses of a validation dataset incurs only $\ln|F_{\text{val}}| = 100$ relative to m_{val}. Nevertheless, the next section presents different generalization bounds that can avoid this possible concern (also see Ng 1997; Cawley and Talbot 2010 for more discussions on the possible over-fitting to validation dataset).

5 Generalization Guarantees Specific to Neural Networks

The previous two sections (Sections 3 and 4) provided an *explanation for generalization* in practical situations. However, it is still of great interest to obtain theoretical *guarantees of the generalization gap*, $R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}]$ (Role 2), which is the focus of this section. To closely analyze neural networks, this section presents a *direct analysis* of neural networks rather than deriving results about neural networks from more generic theory based on capacity, stability, or robustness.

5.1 Model Description via Deep Paths

We consider general neural networks with any depth that have the structure of a directed acyclic graph (DAG) with ReLU nonlinearity and/or max pooling. (e.g., any feedforward network or roll-out of a recurrent network with convolutional and/or fully connected layers). From the DAG structure, we can write the value of each output unit as the summation over all values of all paths from the input to the k-th output unit; that is, for all $k \in \{1, \dots, d_y\}$,

$$h_k^{(H+1)}(x) = \sum_{\text{path}} \bar{x}_{\text{path}} \bar{\sigma}_{\text{path}}(x, w_\sigma) \bar{w}_{\text{path},k} = [\bar{x} \circ \bar{\sigma}(x, w_\sigma)]^\top \bar{w}_k, \tag{1}$$

where $\sum_{\text{path}}$ represents the summation over all paths from the input to the k-th output unit. Here, $[\bar{x} \circ \bar{\sigma}(x, w_\sigma)]$ and $\bar{w}_k$ are the vectorized versions of the summation with the vector size being the number of the paths. Additionally, $\bar{\sigma}(x, w_\sigma)$ represents the activation of each path, corresponding to ReLU nonlinearity and max pooling; $\bar{\sigma}_j(x, w_\sigma) = 1$ if the j-th path is active, and $\bar{\sigma}_j(x, w_\sigma) = 0$ otherwise. Each entry of the vector $\bar{w}$ is the product of the weight parameters for each path, where vectors w_σ and $\bar{w}_k$ share the same weight parameters. We use $[\bar{x} \circ \bar{\sigma}(x, w_\sigma)]$ to denote the element-wise product of $\bar{x}$ and $\bar{\sigma}(x, w_\sigma)$. See Appendix E.2 for an example of equation (1).

5.2 Data-dependent Guarantee

We first focus on regression with squared loss, for which we present our *direct analysis*. Let $z_i = [\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]$ (note the dependence on x and w_σ). Let $\bar{w}_{\mathcal{A}(S_m),k}$ be $\bar{w}_k$ obtained by algorithm $\mathcal{A}$ with dataset S_m. Then, from equation (1), we can decompose the generalization gap into three terms as

$$\begin{aligned} &R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}] \\ &= \sum_{k=1}^{d_y} \left[\bar{w}_{\mathcal{A}(S_m),k}^\top \left(\mathbb{E}[zz^\top] - \frac{1}{m} \sum_{i=1}^m z_i z_i^\top \right) \bar{w}_{\mathcal{A}(S_m),k} \right] \\ &\quad + 2 \sum_{k=1}^{d_y} \left[\left(\frac{1}{m} \sum_{i=1}^m y_{ik} z_i^\top - \mathbb{E}[y_k z^\top] \right) \bar{w}_{\mathcal{A}(S_m),k} \right] \\ &\quad + \mathbb{E}[y^\top y] - \frac{1}{m} \sum_{i=1}^m y_i^\top y_i. \end{aligned}$$

Each of the three terms contains a difference between a sum of *dependent* random variables and its expectation. Accordingly, we want a dataset S_m to be "good" in the sense that these differences are small, as defined as follows.

Definition 3 (concentrated dataset) *A dataset S_m is said to be $(\beta_1, \beta_2, \beta_3)$-concentrated with respect to the square loss and w_σ if $\lambda_{\max}\left(\mathbb{E}[zz^\top] - \frac{1}{m}\sum_{i=1}^m z_i z_i^\top\right) \le \beta_1$, $\left\|\frac{1}{m}\sum_{i=1}^m y_{ik} z_i - \mathbb{E}[y_k z]\right\|_\infty \le \beta_2$, and, $\mathbb{E}[\|y\|_2^2] - \frac{1}{m}\sum_{i=1}^m \|y_i\|_2^2 \le \beta_3$.*

Here, we can see the benefit of deep learning from the viewpoint of "deep-path" feature learning; even if a given S_m is not concentrated in the original space, optimizing w_σ can result in the concentration in the space of z, defined by the *paths* of deep networks.

Proposition 4 states that if a given dataset S_m is good in the sense of Definition 3, then the generalization gap is deterministically bounded based only on the quality of what we get as w, independently of how we choose w (stability and robustness) and the set from which we select w (model complexity and flat minima), as desired (Role 2.1).

Proposition 4 (data-dependent and deterministic bound) *Suppose that S_m is $(\beta_1, \beta_2, \beta_3)$-concentrated with respect to the square loss and w_σ. Then, deterministically,*

$$R[f_{\mathcal{A}(S_m)}] \leq \hat{R}_m[f_{\mathcal{A}(S_m)}] + \beta_1 \sum_{k=1}^{d_y} \left\| \bar{w}_{\mathcal{A}(S_m),k} \right\|_2^2 + \beta_2 \sum_{k=1}^{d_y} \left\| \bar{w}_{\mathcal{A}(S_m),k} \right\|_1 + \beta_3.$$

5.3 Data-independent guarantee

A remaining question is whether we can guarantee a randomly drawn dataset to be "good" in the sense of the concentration. Notice that if z_i were independent of each other, it would have been very easy to guarantee that. However, the z_i are dependent owing to the dependence of w_σ on S_m. We then observe that each step of the (stochastic) gradient decent greedily chooses the best direction only in terms of $\bar{w}$, but not in terms of w_σ (see Appendix E.3). This observation leads to a hypothesis that learning a good w_σ does not require all of the information contained in S_m, and hence the dependence of z_i is not entirely "bad."

5.3.1 Empirical Observations

As a first step to investigate the dependence of z_i, we evaluated the following novel *two-phase* training procedure that explicitly breaks the dependence. In this procedure, we first train a network in a standard way, but only with *partial* training dataset of αm size, where $\alpha \in (0, 1)$ (standard phase). We then freeze w_σ and train only $\bar{w}$ with the *whole* training dataset of size m (freeze phase). Note that vectors w_σ and $\bar{w}$ contain the same shared parameters in the standard phase, whereas these parameters are untied in freeze phase. See Appendix E.4 for a simple implementation of this two-phase training procedure that requires at most (approximately) twice as much computational cost as the normal training procedure.

We implemented the two-phase training procedure with the MNIST and CIFAR-10 datasets. Because we are not aiming for the state-of-the-art results in this section, we used computationally less expensive settings when compared with the next section (where we achieve better performance). The test accuracies of the standard training procedure (base case) were 99.47% for MNIST (ND), 99.72% for MNIST, and 92.89% for CIFAR-10. MNIST (ND) indicates MNIST with no data augmentation. The experimental details are in Appendix E.5. Our source code is available at: `http://lis.csail.mit.edu/code/gdl.html`

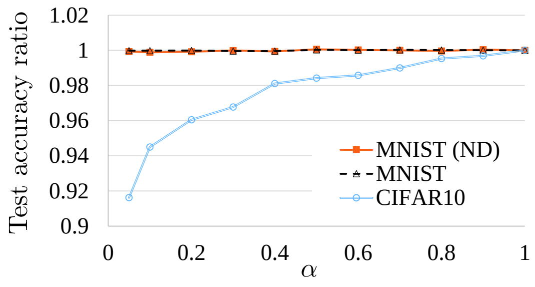

Figure 1: Test accuracy ratio (Two-phase/Base). Notice that the y-axis starts with already high accuracy, even with very small data size αm for learning w_σ.

Figure 1 shows the test accuracy ratio for varying α: the test accuracy of the standard training procedure divided by the test accuracy of this two-phase training procedure. The plot in Figure 1 starts with $\alpha = 0.05$, for which $\alpha m = 3000$ in MNIST and $\alpha m = 2500$ in CIFAR-10. Somewhat surprisingly, using a much smaller dataset for learning w_σ still resulted in competitive performance. Notably, a dataset on which we can more easily get a better generalization (i.e., MNIST) allowed us to use smaller α to achieve competitive performance. This is consistent with our discussion above.

5.3.2 Theoretical Results

Our two-phase training procedure (standard phase plus freeze phase) forces $z_{\alpha m+1}, \ldots, z_m$ to be independent random variables (each z_i is dependent over *coordinates*, which does not matter here), while maintaining competitive practical performance. Thus, by using this two-phase training procedure, one can obtain the following guarantees on practical models, addressing some issues left open in previous studies. Let $m_\sigma = (1-\alpha)m$ and $S_{m_\sigma} = \{(x_{\alpha m+1}, y_{\alpha m+1}), \ldots, (x_m, y_m)\}$.

Assumption 1 *Let $A^{(i)} = \mathbb{E}_x[zz^\top] - z_i z_i^\top$, $B^{(i)}_{kk'} = y_{ik} z_{i,k'} - \mathbb{E}_{x,y}[y_k z_{k'}]$, and $R^{(i)} = \mathbb{E}_y[\|y\|_2^2] - \|y_i\|_2^2$. Assume that for all $i \in \{\alpha m+1, \ldots, m\}$,*

- *$C_{zz} \geq \lambda_{\max}(A^{(i)})$ and $\gamma_{zz}^2 \geq \|\mathbb{E}_x[(A^{(i)})^2]\|_2$*
- *$C_{yz} \geq \max_{k,k'} |B^{(i)}_{kk'}|$ and $\gamma_{yz}^2 \geq \max_{k,k'} \mathbb{E}_x[(B^{(i)}_{kk'})^2])$*
- *$C_y \geq |R^{(i)}|$ and $\gamma_y^2 \geq \mathbb{E}_x[(R^{(i)})^2])$.*

Lemma 5 *Suppose that Assumption 1 holds, and use its notation. Set $\beta_1 = \frac{2C_{zz}}{3m_\sigma} \ln \frac{3d_z}{\delta} + \sqrt{\frac{2\gamma_{zz}^2}{m_\sigma} \ln \frac{3d_z}{\delta}}$, $\beta_2 = \frac{2C_{yz}}{3m_\sigma} \ln \frac{6d_y d_z}{\delta} + \sqrt{\frac{\gamma_{yz}^2}{m_\sigma} \ln \frac{6d_y d_z}{\delta}}$, and $\beta_2 = \frac{2C_y}{3m_\sigma} \ln \frac{3}{\delta} + \sqrt{\frac{2\gamma_y^2}{m_\sigma} \ln \frac{3}{\delta}}$. Then, for any $\delta > 0$, with probability at least $1-\delta$, a dataset S_{m_σ} is $(\beta_1, \beta_2, \beta_3)$–concentrated with respect to the squared loss and a frozen w_σ.*

Theorem 6 (probabilistic bound via random matrix theory) *Suppose that Assumption 1 holds. Set $(\beta_1, \beta_2, \beta_3)$ to be those in Lemma 5. Consider any algorithm $\mathcal{A}$ such that if S_{m_σ} is $(\beta_1, \beta_2, \beta_3)$–concentrated with respect to the squared loss and a frozen w_σ, we have $\sum_k^{d_y} \left\| \bar{w}_{\mathcal{A}(S_m),k} \right\|_1 \leq C_{w1}$ and $\sum_k^{d_y} \left\| \bar{w}_{\mathcal{A}(S_m),k} \right\|_2^2 \leq C_{w2}$ (otherwise, the norms need not be bounded). Then, for any $\delta > 0$, with probability at least $1-\delta$,*

$$R[f_{\mathcal{A}(S_m)}] \leq \hat{R}_{m_\sigma}[f_{\mathcal{A}(S_m)}] + C_{w2}\beta_1 + C_{w1}\beta_2 + \beta_3.$$

Theorem 6 only requires the norms to be bounded for "good" datasets, allowing unbounded norms for "bad" datasets. For classification with 0–1 loss, we adapt the model complexity approach for the two-phase training procedure, yielding Theorem 7.

Theorem 7 *Fix $\rho > 0$. Suppose that $\mathbb{E}_x[\|\bar{x} \circ \bar{\sigma}(x, w_\sigma)\|_2^2] \leq C_\sigma^2$ and $\max_k \|\bar{w}_k\|_2 \leq C_w$ for all $f \in F$. Then, for any $\delta > 0$, with probability at least $1-\delta$, the following holds for all $f \in F$:*

$$R[f] \leq \hat{R}_{m_\sigma,\rho}[f] + \frac{2d_y^2(1-\alpha)^{-1/2}C_\sigma C_w}{\rho\sqrt{m}} + \sqrt{\frac{\ln \frac{1}{\delta}}{2m_\sigma}}.$$

Our main results in this section are Theorems 6 and Proposition 4, which are derived without previous notation of model complexity, stability, robustness and flat minima. Theorems 6 and 7 provide the first generalization bounds for practical deep learning models that do not necessarily have dependence on the number of weights (unlike previous VC dimension approaches), and exponential dependence on depth (unlike Sun et al. 2016; Neyshabur et al. 2015; Xie et al. 2015) and effective input dimensionality (unlike Sokolic et al. 2017a,b). Although the sizes of the vectors corresponding to $C_\sigma C_w$ can be exponentially large in depth, the norms of the vectors (and $C_\sigma C_w$ itself) need not be; indeed, $h_k^{(H+1)}(x) = \|\bar{x} \circ \bar{\sigma}(x, w_\sigma)\|\|\bar{w}\| \cos\theta$ and hence $C_\sigma C_w \approx h_k^{(H+1)}(x)/\cos\theta$. For example, the comparable parts to $(1-\alpha)^{-1/2}C_\sigma C_w$ in our Theorems 7 are in the form of $2^H C_X \prod_{\ell=1}^{H+1} C_{W^{(\ell)}}$ in previously known bounds where C_X and $C_{W^{(\ell)}}$ correspond to the upper-bounds on the norms of the input x and of the weight matrix at the ℓ-th layer (Sun et al., 2016; Neyshabur et al., 2015; Xie et al., 2015). By using a constant α, our bounds can provide significantly improved asymptotic convergence rates when compared to the known rates (Role 2.2).

6 Theoretical Insight for Practical Usage

The previous two sections focused on theoretical bounds on generalization. However, in general, theoretical bounds can be too loose to be directly used in practice (even if they are tight in the sense of theoretical lower bounds. See Section 3.1). Accordingly, this section illustrates the use of generalization theory to provide theoretical insight to guide model class search in practice (Role 3).

In this section, we focus on multi-class classification problems with d_y classes, such as object classification with images. Accordingly, we analyze the expected risk with 0–1 loss as $R[f] = \mathbb{E}_x[\mathbb{1}\{f(x) = y(x)\}]$, where $f(x) = \mathrm{argmax}_{k\in\{1,\ldots,d_y\}}(h_k^{(H+1)}(x))$ is the model prediction, and $y(x) \in \{1, \ldots, d_y\}$ is the true label of x (see Section 2.4.1 in Mohri et al. 2012 for an extension to stochastic labels).

6.1 Theoretical Insight

An application of the result by Koltchinskii and Panchenko (2002) yields the following statement: given a fixed $\rho > 0$, for any $\delta > 0$, with probability at least $1-\delta$, for all $f \in F$,

$$R[f] \leq \hat{R}_{m,\rho}[f] + \frac{2d_y^2}{\rho m}\mathfrak{R}'_m(F) + \sqrt{\frac{\ln \frac{1}{\delta}}{2m}},$$

where $\mathfrak{R}'_m(F)$ is a model complexity defined as

$$\mathfrak{R}'_m(F) = \mathbb{E}_{S_m,\xi}\left[\sup_{k,h_k^{(H+1)}} \sum_{i=1}^m \xi_i h_k^{(H+1)}(x_i)\right].$$

Here, ξ_i are the Rademacher variables (i.e., independent uniform random variables in $\{-1, +1\}$), and the supremum is taken over all $k \in \{1, \ldots, d_y\}$ and all $h_k^{(H+1)}$ allowed in F.

Previous theories (Sun et al., 2016; Neyshabur et al., 2015; Xie et al., 2015) characterize the generalization gap by the upper bounds on the model complexities via the norms of the weight matrices and exponential dependence on the depth 2^H. A close look at the proofs reveals that the norms of weight matrices come from the Cauchy–Schwarz inequality, which can induce a very loose bound. Moreover, the exponential factor 2^H comes from bounding the effect of nonlinearity at each layer, which can also induce a very loose bound, resulting in a gap between theory and practice. Avoiding the use of these possibly loose bounds seems to be challenging if we aim at Roles 1–3 at the same time.

By focusing on Role 3, this section proposes to solve this issue by directly *approximating* the model complexity $\mathfrak{R}'_m(F)$, instead of deriving a possibly too-loose bound on it (for worst possible measures). Here, the need for the approximation comes from the fact that $\mathfrak{R}'_m(F)$ contains the expectation with unknown measure on dataset S_m. The approximation of $\mathfrak{R}'_m(F)$ essentially reduces to the approximation of the expectation over the dataset S_m. In contrast to the worst-case bounds that motivated previous methods, the approximated $\mathfrak{R}'_m(F)$ and whole generalization gap do not necessarily grow along with 2^H and the norms of the weights, as desired.

6.2 Method

The theoretical insight in the previous section suggests the following family of methods: given any architecture and method, add a new regularization term for each mini-batch as

Table 1: Test error (%). A standard variant of LeNet (LeCun et al., 1998) and ResNeXt-29(16×64d) (Xie et al., 2016) are used for MNIST and CIFAR-10, and compared with the addition of the studied regularizer.

Method	MNIST	CIFAR-10
Baseline	0.26	3.52
DARC1	$\underline{0.20}$	$\underline{3.43}$

$$\text{loss} = \text{original loss} + \frac{\lambda}{\bar{m}} \left| \max_k \sum_{i=1}^{\bar{m}} \xi_i h_k^{(H+1)}(x_i) \right|,$$

where x_i is drawn from some distribution approximating the true distribution of x, $\xi_1, \dots, \xi_{\bar{m}}$ are independently and uniformly drawn from $\{-1, 1\}$, $\bar{m}$ is a mini-batch size and λ is a hyper-parameter. Importantly, the approximation of the true distribution of x is only used for regularization purposes and hence needs not be precisely accurate (as long as it plays its role for regularization). For example, it can be approximated by populations generated by a generative neural network and/or an extra data augmentation process. For simplicity, we call this family of methods as Directly Approximately Regularizing Complexity (DARC).

In this paper, we evaluated only a very simple version of the proposed family of methods as a first step. That is, our experiments employed the following simple and easy-to-implement method, called DARC1:

$$\text{loss} = \text{original loss} + \frac{\lambda}{\bar{m}} \left(\max_k \sum_{i=1}^{\bar{m}} |h_k^{(H+1)}(x_i)| \right), \quad (2)$$

where x_i is the i-th sample in the training mini-batch. The additional computational cost and programming effort due to this new regularization is almost negligible because $h_k^{(H+1)}(x_i)$ is already used in computing the original loss. This simplest version was derived by approximating the true distribution of x with the empirical distribution of the training data and the effect of the Rademacher variables via absolute values.

6.3 Experimental Results

We evaluated the proposed method (DARC1) by simply adding the new regularization term in equation (2) to existing standard codes for MNIST and CIFAR-10. For all the experiments, we fixed $(\lambda/\bar{m}) = 0.001$ with $\bar{m} = 64$. We used a single model without ensemble methods. The experimental details are in Appendix F.1. The source code is available at: `http://lis.csail.mit.edu/code/gdl.html`

Table 2: Test error ratio (DARC1/Base)

	MNIST (ND)		MNIST		CIFAR-10	
	mean	stdv	mean	stdv	mean	stdv
Ratio	0.89	0.61	0.95	0.67	0.97	0.79

Table 3: Values of $\frac{1}{m} \left(\max_k \sum_{i=1}^{m} |h_k^{(H+1)}(x_i)| \right)$

Method	MNIST (ND)		MNIST		CIFAR-10	
	mean	stdv	mean	stdv	mean	stdv
Base	17.2	2.40	8.85	0.60	12.2	0.32
DARC1	1.30	0.07	1.35	0.02	0.96	0.01

Table 1 shows the error rates comparable with previous results. To the best of our knowledge, the previous state-of-the-art classification error is 0.23% for MNIST with a single model (Sato et al., 2015) (and 0.21% with an ensemble by Wan et al. 2013). To further investigate the improvement, we ran 10 random trials with computationally less expensive settings, to gather mean and standard deviation (stdv). For MNIST, we used fewer epochs with the same model. For CIFAR-10, we used a smaller model class (pre-activation ResNet with only 18 layers). Table 2 summarizes the improvement ratio: the new model's error divided by the base model's error. We observed the improvements for all cases. The test errors (standard deviations) of the base models were 0.53 (0.029) for MNIST (ND), 0.28 (0.024) for MNIST, and 7.11 (0.17) for CIFAR-10 (all in %).

Table 3 summarizes the values of the regularization term $\frac{1}{m}(\max_k \sum_{i=1}^{m} |h_k^{(H+1)}(x_i)|)$ for each obtained model. The models learned with the proposed method were significantly different from the base models in terms of this value. Interestingly, a comparison of the base cases for MNIST (ND) and MNIST shows that data augmentation by itself *implicitly* regularized what we explicitly regularized in the proposed method.

7 Conclusion

By embracing a logically expected difference between theory and practice, we separated several roles for generalization theory. Our contribution in each separated role is to illustrate the benefit of decomposing the role of theory. However, each contribution of this paper does not fully fulfill each role, leaving the further development of theory in each role as an open problem to future work.

Sections 3 and 4 explained and proved why deep learning ***can*** generalize despite large capacity and possible algorithmic instability, nonrobustness, and sharp minima. However, this paper did not tightly identify the factors, with which deep learning models would be ***guaranteed a priori*** to generalize well in practice. This open problem is also left to future work.

Acknowledgements

We gratefully acknowledge support from NSF grants 1420316, 1523767 and 1723381, from AFOSR FA9550-17-1-0165, from ONR grant N00014-14-1-0486, and from ARO grant W911NF1410433, as well as support from NSERC, CIFAR and Canada Research Chair. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of our sponsors.

References

Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. (2017). A closer look at memorization in deep networks. In *International Conference on Machine Learning*.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *Journal of Machine Learning Research*, 2(Mar):499–526.

Cawley, G. C. and Talbot, N. L. (2010). On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(Jul):2079–2107.

Choromanska, A., Henaff, M., Mathieu, M., Ben Arous, G., and LeCun, Y. (2015). The loss surfaces of multilayer networks. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 192–204.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. (2017). Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*.

Dziugaite, G. K. and Roy, D. M. (2017). Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*.

Finn, C., Goodfellow, I., and Levine, S. (2016). Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems*, pages 64–72.

Gonen, A. and Shalev-Shwartz, S. (2017). Fast rates for empirical risk minimization of strict saddle problems. *arXiv preprint arXiv:1701.04271*.

Hardt, M., Recht, B., and Singer, Y. (2015). Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240*.

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

Hoffer, E., Hubara, I., and Soudry, D. (2017). Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *arXiv preprint arXiv:1705.08741*.

Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*.

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

Koltchinskii, V. and Panchenko, D. (2002). Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, pages 1–50.

Krueger, D., Ballas, N., Jastrzebski, S., Arpit, D., Kanwal, M. S., Maharaj, T., Bengio, E., Fischer, A., and Courville, A. (2017). Deep nets don't learn via memorization. In *Workshop Track of International Conference on Learning Representations*.

Kuzborskij, I. and Lampert, C. (2017). Data-dependent stability of stochastic gradient descent. *arXiv preprint arXiv:1703.01678*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867.

Livni, R., Shalev-Shwartz, S., and Shamir, O. (2014). On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pages 855–863.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2012). *Foundations of machine learning*. MIT press.

Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. (2017). Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems.* to appear.

Neyshabur, B., Tomioka, R., and Srebro, N. (2015). Norm-based capacity control in neural networks. In *Proceedings of The 28th Conference on Learning Theory*, pages 1376–1401.

Ng, A. Y. (1997). Preventing" overfitting" of cross-validation data. In *In Proceedings of the Fourteenth International Conference on Machine Learning.*

Pascanu, R., Montufar, G., and Bengio, Y. (2014). On the number of response regions of deep feed forward networks with piece-wise linear activations. In *International Conference on Learning Representations*.

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. (2017). Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, pages 1–17.

Sato, I., Nishimura, H., and Yokoi, K. (2015). Apac: Augmented pattern classification with neural networks. *arXiv preprint arXiv:1505.03229.*

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

Sokolic, J., Giryes, R., Sapiro, G., and Rodrigues, M. (2017a). Generalization error of invariant classifiers. In *Artificial Intelligence and Statistics*, pages 1094–1103.

Sokolic, J., Giryes, R., Sapiro, G., and Rodrigues, M. R. (2017b). Robust large margin deep neural networks. *IEEE Transactions on Signal Processing.*

Sun, S., Chen, W., Wang, L., Liu, X., and Liu, T.-Y. (2016). On the depth of deep neural networks: a theoretical view. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2066–2072. AAAI Press.

Telgarsky, M. (2016). Benefits of depth in neural networks. In *29th Annual Conference on Learning Theory*, pages 1517–1539.

Tropp, J. A. (2012). User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434.

Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In *Proceedings of the 30th international conference on machine learning (ICML-13)*, pages 1058–1066.

Wu, L., Zhu, Z., et al. (2017). Towards understanding generalization of deep learning: Perspective of loss landscapes. *arXiv preprint arXiv:1706.10239.*

Xie, P., Deng, Y., and Xing, E. (2015). On the generalization error bounds of neural networks under diversity-inducing mutual angular regularization. *arXiv preprint arXiv:1511.07110.*

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2016). Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431.*

Xu, H. and Mannor, S. (2012). Robustness and generalization. *Machine learning*, 86(3):391–423.

Yun, C., Sra, S., and Jadbabaie, A. (2017). Global optimality conditions for deep neural networks. *arXiv preprint arXiv:1707.02444.*

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

Zhang, C., Liao, Q., Rakhlin, A., Sridharan, K., Miranda, B., Golowich, N., and Poggio, T. (2017b). Theory of deep learning iii: Generalization properties of sgd. Technical report, Center for Brains, Minds and Machines (CBMM).

Appendix

A Additional future work

Our two-phase training procedure in Section 5.3 made the theoretical analysis easier with no unrealistic assumption. However, it will be interesting to see if future work can obtain a tight bound without using the two-phase procedure.

Our theoretical insight in Section 6 led to a family of new generalization methods, DARC. Our empirical evaluation of its simplest version showed several promising results for further investigation of DARC.

B Additional notation

The usual empirical margin loss $\hat{R}_{m,\rho}[f]$ is defined as $\hat{R}_{m,\rho}[f] = \frac{1}{m}\sum_{i=1}^{m} \ell_{\text{margin},\rho}(f(x_i), y_i)$, where $\ell_{\text{margin},\rho}$ is defined as follows:

$$\ell_{\text{margin},\rho}(f(x), y) = \ell^{(2)}_{\text{margin},\rho}(\ell^{(1)}_{\text{margin},\rho}(f(x), y))$$

where

$$\ell^{(1)}_{\text{margin},\rho}(f(x), y) = h_y^{(H+1)}(x) - \max_{y \neq y'} h_{y'}^{(H+1)}(x) \in \mathbb{R},$$

and

$$\ell^{(2)}_{\text{margin},\rho}(z) = \begin{cases} 0 & \text{if } \rho \le z \\ 1 - z/\rho & \text{if } 0 \le z \le \rho \\ 1 & \text{if } z \le 0. \end{cases}$$

We use the convention that the training model output $\hat{Y}$ and the training target Y in *matrix* form are real m by d_y matrices, where d_y is the dimension of the targets. The training input matrix X is real m by d_x matrix, where d_x is the dimension of the input. For test dataset, we define $\hat{Y}_{\text{test}}, Y_{\text{test}}$, and X_{test} in the same way with the size of test dataset m_{test}.

For any matrix M, let $\text{Col}(M)$ and $\text{Null}(M)$ be the column space and null space of M.

C Appendix for Section 3

C.1 Theorem 8

Theorem 8 (traditional theory *can* fail to explain generalization of over-parameterized linear models) *Consider a linear model with the training output matrix $\hat{Y} = \Phi w$ where Φ is a fixed input future of X. Let $\hat{Y}_{\text{test}} = \Phi_{\text{test}} w$ where Φ_{test} is a fixed input future of X_{test}. Let $M = [\Phi^\top, \Phi_{\text{test}}^\top]^\top$. Let w^* be the ground-truth parameters. Then, if $n > m$ and if* $\text{rank}(\Phi) \ge m$ *and* $\text{rank}(M) < n$,

(i) For any target $Y \in \mathbb{R}^{m \times d_y}$, there exist parameters w such that $\hat{Y} = Y$, and

(ii) For any $\epsilon, \delta \ge 0$, there exist parameters w and paris of training dataset (Φ, Y) and test dataset $(\Phi_{\text{test}}, Y_{\text{test}})$ such that

(a) $\hat{Y} = Y + \epsilon A$ for some matrix A with $\|A\|_F \le 1$, and

(b) $\hat{Y}_{\text{test}} = Y_{\text{test}} + \epsilon B$ for some matrix B with $\|B\|_F \le 1$, and

(c) $\|w\|_F \ge \delta$ and $\|w - w^*\|_F \ge \delta$.

Proof Since $\text{rank}(\Phi) \ge m$ and $\Phi \in \mathbb{R}^{m \times n}$, the set of its columns span $\mathbb{R}^m$, which proves statement (i). Let $w^* = w_1^* + w_2^*$ where $\text{Col}(w_1^*) \subseteq \text{Col}(M^T)$ and $\text{Col}(w_2^*) \subseteq \text{Null}(M)$. For statement (ii), set the parameter as $w := w_1^* + \epsilon C_1 + \alpha C_2$ where $\text{Col}(C_1) \subseteq \text{Col}(M^T)$, $\text{Col}(C_2) \subseteq \text{Null}(M)$, $\alpha \ge 0$ and $C_2 = \frac{1}{\alpha} w_2^* + \bar{C}_2$. Since $\text{rank}(M) < n$, $\text{Null}(M) \neq \{0\}$ and there exist non-zero $\bar{C}_2$. Then,

$$\hat{Y} = Y + \epsilon \Phi C_1,$$

and

$$\hat{Y}_{\text{test}} = Y_{\text{test}} + \epsilon \Phi_{\text{test}} C_1.$$

By setting $A = \Phi C_1$ and $B = \Phi_{\text{test}} C_1$ with a proper normalization of C_1 yields (a) and (b) in statement (ii) (note that C_1 has an arbitrary freedom in the bound on its scale because its only condition is $\text{Col}(C_1) \subseteq \text{Col}(M^T)$). At the same time with the same parameter, since $\text{Col}(w_1^* + \epsilon C_1) \perp \text{Col}(C_2)$,

$$\|w\|_F^2 = \|w_1^* + \epsilon C_1\|_F^2 + \alpha^2 \|C_2\|_F^2,$$

and

$$\|w - w^*\|_F^2 = \|\epsilon C_1\|_F^2 + \alpha^2 \|\bar{C}_2\|_F^2,$$

which grows unboundedly as $\alpha \to \infty$ without changing A and B, proving (c) in statement (ii). ■

C.2 Corollary 9

Corollary 9 (traditional wisdom *can* fails to explain generalization of linear models) *If $n \le m$ and if* $\text{rank}(M) < n$, *statement (ii) in Theorem 8 holds.*

Proof It follows the fact that the proof in Theorem 8 uses the assumption of $n > m$ and $\text{rank}(\Phi) \ge m$ only for statement (i). ■

C.3 Proof of Proposition 1

Proof Consider statement (i). Given such a F, consider any $\mathcal{A}$ such that $\mathcal{A}$ takes F and S_m as input and outputs f_ϵ^*. Clearly, there are many such algorithms $\mathcal{A}$. For example, given a S_m, fix $\mathcal{A}$ such that $\mathcal{A}$ takes F and S_m as input and outputs f_ϵ^* (which already proves the statement), or even $f_\epsilon^* + \delta$ where δ becomes zero by the right choice of hyper-parameters and of small variations of F (e.g., architecture search in deep learning) such that F still satisfy the condition in the statement. This proves statement (i).

Consider statement (ii). Given any dataset S'_m, consider a look-up algorithm $\mathcal{A}'$ that always outputs f_ϵ^* if $S_m = S'_m$, and outputs f_1 otherwise such that f_1 is arbitrarily non-robust and $|\ell(f_\epsilon^*(x), y) - \ell(f_1(x), y)|$ is arbitrarily large (i.e., arbitrarily non-stable). This proves statement (ii).

■

Note that while A' in the above proof suffices to prove statement (ii), we can also generate other non-stable and non-robust algorithms by noticing the essence of this proof discussed in the main text right before Proposition 1 as a rephrase of the proposition.

C.4 Additional Explanation for Role of Generalization Theory

Role 2.2 is important and has been a more focus in theoretical literature. However, practical deep learning approach may change a model class flexibly as m increases in the way that is not considered in theory. For example, theory with the focus on Role 2.2 would conclude that the generalization gap is upper bounded by $O(\sqrt{n/m})$ with n being the number of parameters, and hence approaches to zero as m increases. However, n may increase as m increase in practice, while still producing a good generalization gap. Thus, we believe that Role 2.1 is also important as well as Role 2.2.

D Appendix for Section 4

D.1 Proof of Proposition 2

Proof Consider a single fixed $f \in F_{\text{val}}$. Since F_{val} is independent from the validation dataset, $z_{f,1}, \dots, z_{f,m_{\text{val}}}$ are independent zero-mean random variables, given a fixed $f \in F_{\text{val}}$. Thus, we can apply Bernstein inequality, yielding

$$\mathbb{P}\left(\frac{1}{m_{\text{val}}}\sum_{i=1}^{m_{\text{val}}} z_{f,i} > \epsilon\right) \le \exp\left(-\frac{\epsilon^2 m_{\text{val}}/2}{\gamma^2 + \epsilon C/3}\right).$$

By taking union bound over all elements in F_{val},

$$\mathbb{P}\left(\cup_{f\in F_{\text{val}}}\left\{\frac{1}{m_{\text{val}}}\sum_{i=1}^{m_{\text{val}}} z_{f,i} > \epsilon\right\}\right) \le |F_{\text{val}}|\exp\left(-\frac{\epsilon^2 m_{\text{val}}/2}{\gamma^2 + \epsilon C/3}\right).$$

By setting $\delta = |F_{\text{val}}|\exp\left(-\frac{\epsilon^2 m_{\text{val}}/2}{\gamma^2+\epsilon C/3}\right)$ and solving for ϵ (via quadratic formula),

$$\epsilon = \frac{2C\ln(\frac{|F_{\text{val}}|}{\delta})}{6m_{\text{val}}} \pm \frac{1}{2}\sqrt{\left(\frac{2C\ln(\frac{|F_{\text{val}}|}{\delta})}{3m_{\text{val}}}\right)^2 + \frac{8\gamma^2\ln(\frac{|F_{\text{val}}|}{\delta})}{m_{\text{val}}}}.$$

By noticing that the solution of ϵ with the minus sign results in $\epsilon < 0$, which is invalid for Bernstein inequality, we obtain the valid solution with the plus sign. Then, we have

$$\epsilon \le \frac{2C\ln(\frac{|F_{\text{val}}|}{\delta})}{3m_{\text{val}}} + \sqrt{\frac{2\gamma^2\ln(\frac{|F_{\text{val}}|}{\delta})}{m_{\text{val}}}},$$

where we used that $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$. By tanking the negation of the statement, we obtain that for any $\delta > 0$, with probability at least $1-\delta$, for all $f \in F_{\text{val}}$,

$$\frac{1}{m_{\text{val}}}\sum_{i=1}^{m_{\text{val}}} z_{f,i} \le \frac{2C\ln(\frac{|F_{\text{val}}|}{\delta})}{3m_{\text{val}}} + \sqrt{\frac{2\gamma^2\ln(\frac{|F_{\text{val}}|}{\delta})}{m}},$$

where $\frac{1}{m_{\text{val}}}\sum_{i=1}^{m_{\text{val}}} z_{f,i} = R[f] - \hat{R}_{\text{val}}[f]$. ■

Note that F_{val} needs to be independent from validation dataset as defined in the main text, and otherwise the proof of Proposition of 2 breaks down.

E Appendix for Section 5

We use the following lemma in the proof of Lemma 5.

Lemma 10 (Matrix Bernstein inequality: corollary to theorem 1.4 in Tropp 2012) *Consider a finite sequence $\{M_i\}$ of independent, random, self-adjoint matrices with dimension d. Assume that each random matrix satisfies that $\mathbb{E}[M_i] = 0$ and $\lambda_{\max}(M_i) \le R$ almost surely. Let $\gamma^2 = \|\sum_i \mathbb{E}[M_i^2]\|_2$. Then, for any $\delta > 0$, with probability at least $1-\delta$,*

$$\lambda_{\max}\left(\sum_i M_i\right) \le \frac{2R}{3}\ln\frac{d}{\delta} + \sqrt{2\gamma^2\ln\frac{d}{\delta}}.$$

Proof Theorem 1.4 by Tropp (2012) states that for all $t \ge 0$,

$$\mathbb{P}\left[\lambda_{\max}\left(\sum_i M_i\right) \ge t\right] \le d \cdot \exp\left(\frac{-t^2/2}{\gamma^2 + Rt/3}\right).$$

Setting $\delta = d \exp\left(-\frac{t^2/2}{\gamma^2+Rt/3}\right)$ implies

$$-t^2 + \frac{2}{3}R(\ln d/\delta)t + 2\gamma^2 \ln d/\delta = 0.$$

Solving for t with the quadratic formula and bounding the solution with the subadditivity of square root on non-negative terms (i.e., $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for all $a, b \geq 0$),

$$t \leq \frac{2}{3}R(\ln d/\delta) + 2\gamma^2 \ln d/\delta.$$

■

E.1 Proof of Proposition 4

Proof From the definition of induced matrix norm and the Cauchy-Schwarz inequality,

$$\begin{aligned}
&R[f_{\mathcal{A}(S_m)}] - \hat{R}_m[f_{\mathcal{A}(S_m)}] \\
&\leq \sum_{k=1}^{d_y} \left\|\bar{w}_{\mathcal{A}(S_m),k}\right\|_2^2 \lambda_{\max}\left(\mathbb{E}[zz^\top] - \frac{1}{m}\sum_{i=1}^m z_i z_i^\top\right) \\
&\quad + 2\sum_{k=1}^{d_y} \|\bar{w}_{\mathcal{A}(S_m),k}\|_1 \left\|\frac{1}{m}\sum_{i=1}^m y_{ik} z_i - \mathbb{E}[y_k z]\right\|_\infty \\
&\quad + \mathbb{E}[y^\top y] - \frac{1}{m}\sum_{i=1}^m y_i^\top y_i.
\end{aligned}$$

Then, the statement directly follows the conditions on S_m and $\mathcal{A}$. ■

E.2 From Deep Models to Sum of Paths Expression

Here, we present a simple example for equation (1) in the main text. For fully-connected feedforward networks with ReLU and max pooling, we can write $\sum_{\text{path}}$ in equation (1) simply as

$$\begin{aligned}
&\sum_{\text{path}} \bar{x}_{\text{path}} \bar{\sigma}_{\text{path}}(x, w_\sigma) \bar{w}_{\text{path},k} \\
&= \sum_{k_0=0}^{d_x} \sum_{k_1=1}^{n_1} \cdots \sum_{k_H=1}^{n_H} x_{k_0} \bar{\sigma}_{k_1,\dots,k_H}(x, w_\sigma) \bar{w}_{k1,\dots,k_H,k}
\end{aligned}$$

where $\bar{w}_{k1,\dots,k_H,k} = \left(\prod_{l=1}^{H+1} w_{k_{l-1}k_l}^{(l)}\right)$ is the product of weights for each path ($w_{k_{l-1}k_l}^{(l)}$ is the weight between k_{l-1}-th unit and k_l-th unit at the l-th layer), and $n_1, \dots, n_H$ are the number of hidden unites at each layer. Here, $\bar{\sigma}_{k_1,\dots,k_H}(x, w_\sigma)$ represent ReLU nonlinearity and/or max pooling for each path. That is, $\bar{\sigma}_{k_1,\dots,k_H}(x, w_\sigma) = 1$ if all the units in the path is active and $\bar{\sigma}_{k_1,\dots,k_H}(x, w_\sigma) = 0$ otherwise.

E.3 SGD Chooses Direction only in terms of $\bar{w}$ but not in terms of w_σ

Each step of (stochastic) gradient decent is the steepest decent direction in terms of the first order term of Taylor expansion of the objective function L at a current parameter w:

$$L(w + d) = L(w) + \nabla_w L(w)^T d + \|d\| g(w, d),$$

where $\lim_{d \to 0} g(w, d) = 0$. For sufficiently small $\|d\|$, the first order term $\nabla_w L(w)^T d$ becomes dominant, which is minimized with (stochastic) gradient decent direction $d = -\nabla_w L(w)$. Recall that

$$h_k^{(H+1)}(x) = [\bar{x} \circ \bar{\sigma}(x, w_\sigma)]^\top \bar{w}.$$

Note that $\sigma(x, w_\sigma)$ is 0 or 1 for max-pooling and/or ReLU nonlinearity. Thus, the derivative of $[\bar{x} \circ \bar{\sigma}(x, w_\sigma)]$ with respect to w is zero everywhere (except at the measure zero set where the derivative does not exists). Thus, by the chain rule (and power rule), $\nabla_w L(w)^T d$ only contain the contribution from derivative of $h_k^{(H+1)}$ with respect to $\bar{w}$, but not with respect to w_σ.

E.4 Simple Implementation of Two-Phase Training Procedure

Directly implementing equation (1) requires the summation over all paths, which can be computationally expensive. To avoid it, we implemented it by creating two deep neural networks, one of which defines $\bar{w}$ paths hierarchically, and another of which defines w_σ paths hierarchically, resulting in the computational cost at most (approximately) twice as much as the original cost. We tied w_σ and $\bar{w}$ in the two networks during standard phase, and untied them during freeze phase.

For example, for ReLU nonlinearity σ_{ReLU}, the computation of the network can be re-written as:

$$\begin{aligned}
h^{(l)}(x) &= \sigma_{\text{ReLu}}(W^{(l)} h^{(l-1)}(x)) \\
&= \dot{\sigma}_{\text{ReLu}}(W^{(l)} h^{(l-1)}(x)) \circ W^{(l)} h^{(l-1)}(x) \\
&= \dot{\sigma}_{\text{ReLu}}(W_\sigma^{(l)} h_\sigma^{(l-1)}(x)) \circ W^l h^{(l-1)}(x)
\end{aligned}$$

where $W_\sigma^{(l)} = W^{(l)}$, $h_\sigma^{(l-1)} = \sigma_{\text{ReLu}}(W_\sigma^{(l)} h_\sigma^{(l-1)}(x))$ and $\dot{\sigma}_{\text{ReLu}}(M)_{ij} = 0$ if $M_{ij} < 0$ and $\dot{\sigma}_{\text{ReLu}}(M)_{ij} = 1$ if $M_{ij} \geq 0$. Note that if $W_\sigma^{(l)} = W^{(l)}$, we have that $h_\sigma^{(l-1)} = h^{(l)}$. In the two-phase training procedure, we created two networks for $W_\sigma^{(l)} h_\sigma^{(l-1)}(x)$ and $W^{(l)} h^{(l-1)}(x)$ separately. We then set $W_\sigma^{(l)} = W^{(l)}$ during standard phase, and frozen $W_\sigma^{(l)}$ and only trained $W^{(l)}$ during freeze phase. Our source code is available at: `http://lis.csail.mit.edu/code/gdl.html`

E.5 Experimental detail in Section 5.3.1

For all MNIST(ND), MNIST and CIFAR-10, we used the same settings as those for Tables 2 and 3 in Section 6. That is, for all experiments, we still used the same fixed value of $(\lambda/\bar{m}) = 0.001$ with $\bar{m} = 64$. Other experimental detail is also identical to what is described in Appendix F.1, except that we used 1000 epochs for each standard and freeze phase.

E.6 Proof of Lemma 5

Proof We use the fact that for data pints $i \in \{\alpha m + 1, \dots, m\}$ (its size is $m_\sigma = (1-\alpha)m$), $z_{\alpha m+1}, \dots, z_m$ are independent random variables, because w_σ is frozen and independent from $x_{\alpha m+1}, \dots, x_m$.

For β_1: Matrix Bernstein inequality (Lemma 10) states that for any $\delta > 0$, with probability at least $1 - \delta/3$,

$$\lambda_{\max}\left(\mathbb{E}[zz^\top] - \frac{1}{m_\sigma}\sum_{i=1}^{m_\sigma} z_i z_i^\top\right) \le \frac{2C_{zz}}{3m_\sigma}\ln\frac{3d_z}{\delta} + \sqrt{\frac{2\gamma_{zz}^2}{m_\sigma}\ln\frac{3d_z}{\delta}}.$$

Here, Matrix Bernstein inequality was applied as follows. Let $M_i = (\frac{1}{m_\sigma}A^{(i)})$. Then, $\sum_{i=1}^{m_\sigma} M_i = \mathbb{E}[zz^\top] - \frac{1}{m_\sigma}\sum_{i=1}^{m_\sigma} z_i z_i^\top$. We have that $\mathbb{E}[M_i] = 0$ for all i. Also, $\lambda_{\max}(M_i) \le \frac{1}{m_\sigma}C_{zz}$ and $\|\sum_i \mathbb{E}[M_i^2]\|_2 \le \frac{1}{m_\sigma}\gamma_{zz}^2$.

For β_2: We apply Bernstein inequality to each $(k, k') \in \{1, \dots, d_y\} \times \{1, \dots, d_z\}$ and take union bound over $d_y d_z$ events, obtaining that for any $\delta > 0$, with probability at least $1 - \delta/3$, for all $k \in \{1, 2, \dots, d_y\}$,

$$\left\|\frac{1}{m_\sigma}\sum_{i=1}^{m_\sigma} y_{ik} z_i - \mathbb{E}[y_k z]\right\|_\infty \le \frac{2C_{yz}}{3m_\sigma}\ln\frac{6d_y d_z}{\delta} + \sqrt{\frac{\gamma_{yz}^2}{m_\sigma}\ln\frac{6d_y d_z}{\delta}}$$

For β_3: From Bernstein inequality, with probability at least $1 - \delta/3$,

$$\mathbb{E}[y^\top y] - \frac{1}{m_\sigma}\sum_{i=1}^{m_\sigma} y_i^\top y_i \le \frac{2C_y}{3m}\ln\frac{3}{\delta} + \sqrt{\frac{2\gamma_y^2}{m}\ln\frac{3}{\delta}}.$$

■

E.7 Proof of Theorem 6

Proof Notice that there is a dependence between z_i and $\bar{w}$ in the form of $\|g(z_i)\|\|\bar{w}\|$ for some g, because $\bar{w}$ is learned with all training dataset (instead of $(1-\alpha)m$ sized partial dataset). This dependence is taken care of by the condition of the algorithm $\mathcal{A}$ in the statement for the bounded norms. Unlike model complexity approach, the norms need not be bounded given the event where a dataset is not concentrated. From Lemma 5, we know that a event where dataset is concentrated happens with probability at least $1 - \delta$. Thus, the norms are required to be bounded only conditioned on such a event, to ensure that the occurrence of such an event immediately implies our bound. Hence, the statement directly follows Proposition 4 and Lemma 5. ■

E.8 Proof of Theorem 7

Proof Recall the following fact: using the result by Koltchinskii and Panchenko (2002), we have that for any $\delta > 0$, with probability at least $1-\delta$, the following holds for all $f \in F$:

$$R[f] \le \hat{R}_{m_\sigma,\rho}[f] + \frac{2d_y^2}{\rho m_\sigma}\mathfrak{R}'_{m_\sigma}(F) + \sqrt{\frac{\ln\frac{1}{\delta}}{2m_\sigma}},$$

where $\mathfrak{R}'_{m_\sigma}(F)$ is Rademacher complexity defined as

$$\mathfrak{R}'_{m_\sigma}(F) = \mathbb{E}_{S_{m_\sigma},\xi}\left[\sup_{k, h_k^{(H+1)}}\sum_{i=1}^{m_\sigma}\xi_i h_k^{(H+1)}(x_i)\right].$$

Here, ξ_i is the Rademacher variable, and the supremum is taken over all $k \in \{1, \dots, d_y\}$ and all $h_k^{(H+1)}$ allowed in F. Then, for our parameterized model classes with any frozen w_σ,

$$\begin{aligned}
&\mathfrak{R}'_{m_\sigma}(F)\\
&= \mathbb{E}_{S_{m_\sigma},\xi}\left[\sup_{k,\bar{w}_k}\sum_{i=1}^{m_\sigma}\xi_i[\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]^\top \bar{w}_k\right]\\
&\le \mathbb{E}_{S_{m_\sigma},\xi}\left[\sup_{k,\bar{w}_k}\left\|\sum_{i=1}^{m_\sigma}\xi_i[\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]\right\|_2 \|\bar{w}_k\|_2\right]\\
&\le C_w \mathbb{E}_{S_{m_\sigma},\xi}\left[\left\|\sum_{i=1}^{m_\sigma}\xi_i[\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]\right\|_2\right].
\end{aligned}$$

Because square root is concave in its domain, by using Jensen's inequality and linearity of expectation,

$$\begin{aligned}
&\mathbb{E}_{S_{m_\sigma},\xi}\left[\left\|\sum_{i=1}^{m_\sigma}\xi_i[\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]\right\|_2\right]\\
&\le \left(\mathbb{E}_{S_{m_\sigma}}\sum_{i=1}^{m_\sigma}\sum_{j=1}^{m_\sigma}\mathbb{E}_\xi[\xi_i\xi_j][\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]^\top[\bar{x}_j \circ \bar{\sigma}(x_j, w_\sigma)]\right)^{1/2}\\
&= \left(\sum_{i=1}^{m_\sigma}\mathbb{E}_{S_{m_\sigma}}\left[\|[\bar{x}_i \circ \bar{\sigma}(x_i, w_\sigma)]\|_2^2\right]\right)^{1/2}\\
&\le C_\sigma\sqrt{m_\sigma}.
\end{aligned}$$

Putting together, we have that $\mathfrak{R}'_m(F) \leq C_\sigma C_w \sqrt{m_\sigma}$.
■

F Appendix for Section 6

F.1 Experimental detail in Section 6.3

For MNIST:

We used the following fixed architecture:

Layer 1	Convolutional layer with 32 filters with filter size of 5 by 5, followed by max pooling of size of 2 by 2 and ReLU.
Layer 2	Convolution layer with 32 filters with filter size of 5 by 5, followed by max pooling of size of 2 by 2 and ReLU.
Layer 3	Fully connected layer with output 1024 units, followed by ReLU and Dropout with its probability being 0.5.
Layer 4	Fully connected layer with output 10 units.

Layer 4 outputs $h^{(H+1)}$ in our notation. For training purpose, we use softmax of $h^{(H+1)}$. Also, $f(x) = \text{argmax}(h^{(H+1)}(x))$ is the label prediction.

We fixed learning rate to be 0.01, momentum coefficient to be 0.5, and optimization algorithm to be (standard) stochastic gradient decent (SGD). We fixed data augmentation process as: random crop with size 24, random rotation up to ± 15 degree, and scaling of 15%. We used 3000 epochs for Table 1, and 1000 epochs for Tables 2 and 3.

For CIFAR-10:

For data augmentation, we used random horizontal flip with probability 0.5 and random crop of size 32 with padding of size 4.

For Table 1, we used ResNeXt-29(16×64d) (Xie et al., 2016). We set initial learning rate to be 0.05 and decreased to 0.005 at 150 epochs, and to 0.0005 at 250 epochs. We fixed momentum coefficient to be 0.9, weight decay coefficient to be 5×10^{-4}, and optimization algorithm to be stochastic gradient decent (SGD) with Nesterov momentum. We stopped training at 300 epochs.

For Tables 2 and 3, we used pre-activation ResNet with only 18 layers (pre-activation ResNet-18) (He et al., 2016). We fixed learning rate to be 0.001 and momentum coefficient to be 0.9, and optimization algorithm to be (standard) stochastic gradient decent (SGD). We used 1000 epochs.