SCIENCES
SORBONNE
UNIVERSITÉ

Generalization in Deep Learning

Hakim CHEKIROU

October 4, 2021

Abstract

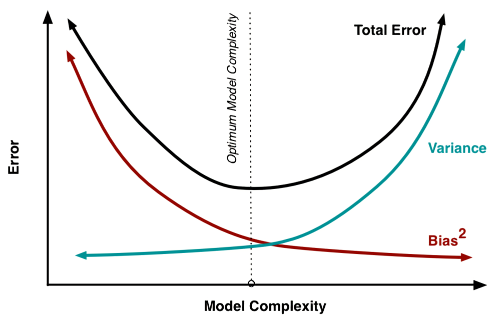
This course is centered around the long lasting puzzle of explaining generalization in deep learning. We first show the intrinsic limitations of traditional learning theoretic methods at this task. We then explore the rich line of works that originated from the idea of taking into account the implicit regularization of the learning algorithm and show troubling aspects regarding them. At the heart of this course, is a questioning of uniform convergence as the underlying technique for generalization in deep learning. We present setups were any two-sided uniform convergence bound, as cleverly applied as it may be, provably fails at explaining the generalization properties of over-parameterized deep learning models. We aim to give a good overview of the many hurdles on the way to achieving satisfying theoretical explanations of deep learning.

Contents

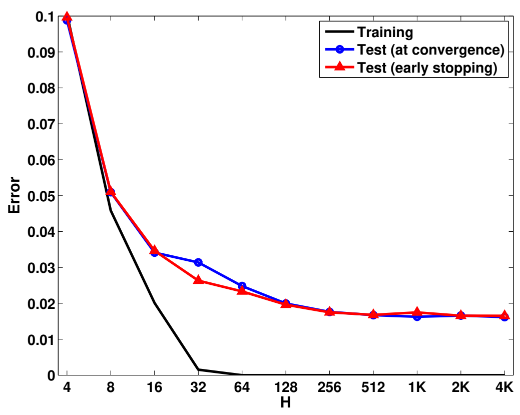
1 Introduction

Deep learning models have been established as state of the art techniques across a wide variety of machine learning tasks, starting from the very influential work in [12] focusing on image classification. From then on, larger and deeper networks were used to continuously improve performances, attaining parameter counts in the order of millions for VGG [23], or even billions in large language models [5].

In fact, the networks used in practice are over-parameterized, meaning that the number of parameters exceeds the training data size. In such a regime, they can easily fit the training data [27], but manage to achieve small generalization errors. Traditional wisdom suggests that a more complex class of models would over-fit the training data following a U-shaped curve figure 1a commonly known as the bias-variance tradeoff. However for deep learning models, even without any explicit form of regularization such as weight decay or early stopping, increasing the model complexity doesn't degrade the generalization performance [18].



(a)



(b)

Figure 1: **a**: an illustration of the common intuition for the bias-variance tradeoff [17]. **b:** Generalization behavior of 2 layer neural networks on MNIST as the complexity (number of hidden units) H increases, experiment by [18]

The natural question that arises from the empirical observations is: why do these models generalize well ? The main motivation for deriving generalization bounds is that it could permit to better asses the generalization proprieties of deep learning architectures and lead to better designs.

In this course, we will first formally define the concepts of generalization, deep learning models and other tools used later. We will then state the criteria that any generalization bound must meet to explain deep learning. In section 5.1 , We explain how the traditional learning theoretic tools,VC dimension and Rademacher complexity, fail to meet those criteria. In section 6, we review works that suggested the incorporation of the training algorithm (SGD) into learning theoretic approaches and the various resulting algorithm dependent bounds that were developed in recent years. We will then show how the weight norm dependence of many of these bounds introduces a dependence on the training set size.
The most important result of this work is presented in section 7 were we define the notion of tightest possible uniform convergence bound and present setups were any uniform convergence bound would provably results in loose guarantees.

2 Preliminaries

We consider supervised learning problems, dealing with predicting outputs belonging to an output space $\mathcal{Y}$ from inputs belonging to an input space $\mathcal{X}$, given a sample S of m input-output pairs sampled i.i.d. from $\mathcal{D}$, $S = \{(x_i, y_i)\}_{i=1}^m \sim \mathcal{D}^m$, where $x_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$ and $\mathcal{D}$ is the assumed *data distribution* on the set of input-output pairs $\mathcal{X} \times \mathcal{Y}$.

A loss function $\mathcal{L}$ evaluates the correctness of a prediction $\hat{y} \in \mathcal{Y}$ when compared to the observed output $y \in \mathcal{Y}$. We define the expected value of the loss on the outputs predicted by a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ as :

$$\mathcal{L}_{\mathcal{D}}(h) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathcal{L}(h(\mathbf{x}), y)] \tag{1}$$

The empirical twin of this error over a set S is given by :

$$\hat{\mathcal{L}}_S(h) = \frac{1}{m} \sum_{(x,y)\in S} \mathcal{L}(h(x), y) \tag{2}$$

For the case of classification, where $\mathcal{Y}$ is a discrete set, we are mainly interested in the 0-1 error, defined as :

$$\ell(\hat{y}, y) = \mathbb{1}_{[\hat{y} \neq y]} \tag{3}$$

The expected 0-1 loss of a hypothesis $h : \mathcal{X} \to \mathcal{Y}$, is then given by :

$$\epsilon(h) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\ell(h(\mathbf{x}), y)] = \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}(h(\mathbf{x}) \neq y) \tag{4}$$

Along with it's empirical twin over a set S :

$$\hat{\epsilon}(h, S) = \frac{1}{m} \sum_{(x,y)\in S} \mathbb{1}[h(x) \neq y] \tag{5}$$

Learning Algorithm. A learning algorithm is a mapping $\mathcal{A} : (\mathcal{X} \times \mathcal{Y})^* \to \mathcal{Y}^{\mathcal{X}}$ from training sets of any size to hypothesis functions. A minimal assumption on the algorithm is that it outputs hypotheses within a set of hypotheses called the hypotheses class, denoted $\mathcal{H}$. We also define h_S as the hypothesis output by the algorithm $\mathcal{A}$ on a data set S.

Definition 1. ***The generalization error*** *of $\mathcal{A}$ with respect to a loss function $\mathcal{L}$, is the smallest value $\epsilon_{gen}(m, \delta)$ such that:*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left[\mathcal{L}_{\mathcal{D}}(h_S) - \hat{\mathcal{L}}_S(h_S) \leq \epsilon_{gen}(m, \delta)\right] \geq 1 - \delta$$

In other words, with probability at least $1 - \delta$, $\epsilon_{\text{gen}}(m, \delta)$ bounds the difference between the empirical error of the hypothesis h_S on the train set S and the expected error of the hypothesis h_S over $\mathcal{D}$
In supervised learning the goal is then, under some assumptions made on the data distribution $\mathcal{D}$, to find an algorithm $\mathcal{A}$ yielding a hypothesis h_S with a low generalisation error $\epsilon_{\text{gen}}(m, \delta)$.

Over-fitting. We say that a hypothesis or model h_S overfits the training data when $\hat{\mathcal{L}}_S(h_S)$ is small but $\mathcal{L}_{\mathcal{D}}(h_S)$ is much larger. The goal of training a model is to learn the underlying distribution $\mathcal{D}$, not to memorize the training data itself. In the latter case, the model loses its generalisation properties, i.e., performs badly on new unseen data drawn from $\mathcal{D}$.

A bound on the generalization error of the algorithm can be obtained by applying a two-sided uniform convergence bound on the hypothesis class used by the algorithm, where, for a given draw, S, we look at convergence for all the hypotheses in $\mathcal{H}$ instead of just h_S:

Definition 2. *The* ***uniform convergence bound*** *with respect to loss $\mathcal{L}$ is the smallest value $\epsilon_{unif}(m, \delta)$ such that:*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}}\left|\mathcal{L}_{\mathcal{D}}(h) - \hat{\mathcal{L}}_S(h)\right| \leq \epsilon_{unif}(m, \delta)\right] \geq 1 - \delta$$

Finally, Let $\|.\|_F$, $\|.\|_1$, $\|.\|_2$ and $\|.\|_{2,1}$ denote the Frobenius norm, the element-wise ℓ_1 norm, the spectral norm and the $(2, 1)$ matrix norm respectively. We further denote the ℓ_p norm of a vector by $|.|_p$.

3 Neural Networks

Classification Model. Consider the classification task with input domain $\mathcal{X}_{B,n} = \{\mathbf{x} \in \mathbb{R}^n \mid \sum_{i=1}^n x_i^2 \leq B^2\}$ and output domain $\mathcal{Y}$ with k labels, where the output of the model is a score for each class and the class with the maximum score will be selected as the predicted label. In this case, a hypothesis is a mapping from $\mathcal{X} \to \mathbb{R}^k$ rather than $\mathcal{Y}$.

Definition 3. *(Feed forward Neural Network) Let $f_{\mathbf{w}}(\mathbf{x}) : \mathcal{X}_{B,n} \to \mathbb{R}^k$ be the function computed by a d layer feed-forward network for the classification task with parameters $\mathbf{w} = vec\left(\{W_i\}_{i=1}^d\right)$,*

$$f_{\mathbf{w}}(\mathbf{x}) = W_d\,\phi(W_{d-1}\,\phi(....\phi(W_1\mathbf{x}))),$$

here ϕ is a non-linear activation function.

Let $f_{\mathbf{w}}^i(\mathbf{x})$ denote the output of layer i before activation and h be an upper bound on the number of output units in each layer, also called the width of the network.
The most common activation functions are the rectified linear unit (ReLU), the sigmoid function and the hyperbolic tangent (tanh) function, these function are defined in Table 1.

Name	Function: $\phi(x)$	Function output range
ReLU	max{0,x}	$[0, \infty)$
Sigmoid	$\frac{1}{1+e^{-x}}$	(0,1)
Hyperbolic tangent	$\frac{e^x-e^{-x}}{e^x+e^{-x}}$	(-1,1)

Table 1: Non-linear activation functions

Margin γ. The classification margin γ on data point $(\mathbf{x}, y)$ is the difference between the score assigned to the real class y and the maximum score over the other classes.

$$\gamma = f_{\mathbf{w}}(\mathbf{x})[y] - \max_{j \neq y} f_{\mathbf{w}}(\mathbf{x})[j] \tag{6}$$

Margin Loss. For any distribution $\mathcal{D}$ and margin $\gamma > 0$, we define the expected margin loss as follows:

$$L_\gamma(f_{\mathbf{w}}) = \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}}\left[f_{\mathbf{w}}(\mathbf{x})[y] \leq \gamma + \max_{j \neq y} f_{\mathbf{w}}(\mathbf{x})[j]\right] \tag{7}$$

Let $\widehat{L}_\gamma(f_{\mathbf{w}})$ be the empirical estimate of the above expected margin loss. Note that for $\gamma = 0$, it corresponds in this case to the classification loss, identical to what we defined in section 2.

Ramp loss. We define a more general margin-based surrogate of the 0-1 error (also called as ramp loss) in a binary classification setting. Specifically, given the classifier's logit output $y' \in \mathbb{R}$ and the true label $y \in \{-1, +1\}$, define

$$\mathcal{L}^{(\gamma)}(y', y) = \begin{cases} 1 & yy' \leq 0 \\ 1 - \frac{yy'}{\gamma} & yy' \in (0, \gamma) \\ 0 & yy' \geq \gamma. \end{cases}$$

Note that $\mathcal{L}^{(0)}$ is the 0-1 loss.

Training deep neural networks. Training deep neural network is done by empirical risk minimisation (**ERM**) by some variant of GD or SGD for some non convex function. Given the train set S, the aim is to minimize the empirical risk on this sample $\frac{1}{m}\sum_{(x,y)\in S}\mathcal{L}(f_{\mathbf{w}}(x), y)$.

4 Criteria For a Generalization Bound

For a bound to resolve the generalization problem in deep learning, the following criteria must be met :

1. The bound must be small, ideally non-vacuous (i.e., < 1).
2. Reflect the same width/depth dependence as the generalization error (e.g., become smaller with increasing width, as has been surprisingly observed in practice).
3. Apply to the network learned by SGD (without any modification or explicit regularization).
4. Increase with the proportion of randomly flipped training labels (i.e., increase with memorization).
5. The bounds should decrease with the training data set size m at the same rate as the generalization error.

5 Algorithm-Independent Bounds

In this section, we present bounds obtained by making no assumptions on the algorithm $\mathcal{A}$

5.1 Data-Independent Bounds: VC dimension

The most general result, that makes no assumptions on the algorithm nor the data set S is achieved by considering the VC-dimension, which characterizes uniform convergence of misclassification frequencies to probabilities [25]. The main result of this theory is that the optimal bound of this form (up to a multiplicative fixed constant) for the generalization gap, in the case of binary classification is

$$\forall\mathcal{D},\ \mathbf{P}_{S\sim\mathcal{D}^m}\left[\sup_{h\in\mathcal{H}}|\epsilon(h) - \hat{\epsilon}(h)| \leq C\sqrt{\frac{\text{VC}(\mathcal{H}) + \ln\frac{1}{\delta}}{m}}\right] \geq 1 - \delta \tag{8}$$

for some constant C, and where $\mathrm{VC}(\mathcal{H})$ is a combinatorial quantity called the Vapnik-Chervonenkis dimension [22].

The VC dimension of feedforward networks can be bounded in terms of the number of parameters. In particular, [9] give the following tight bound on the VC dimension and hence capacity of feed-forward networks with ReLU activations:

$$VC(\mathcal{H}) = \tilde{\Theta}(d^2h^2) \tag{9}$$

This yields generalization bounds of the form $\tilde{\mathcal{O}}(dh/\sqrt{m})$, ignoring logarithmic factors. ReLU neural networks used in practice often have significantly more parameters than samples [27],thus leading to vacuous bounds (> 1).

Many empirical works have shown that, once in the over-parameterized regime, deep neural networks typically show a monontonic decrease in generalization error as over-parameterization increases, unlike what the VC dimension bound would suggest, violating criterions 1 and 2. Since, the VC dimension bound is optimal among data-independent algorithm-independent bounds, these results tell us that this class of bounds is fundamentally unable to explain the generalization of over-parameterized neural networks.

5.2 Data-Dependent Bounds : Rademacher Complexity

Many works considered alternative complexity measures of a function class [4, 10, 11]. Rademacher Complexity, alternatively to VC dimension, is a data-dependent measure of the complexity of a hypothesis set of functions $\mathcal{H}$.

Definition 4. *(Rademacher complexity) Let $\mathcal{F}$ be a set of functions and $\mathcal{F} \circ S$ be the set of all possible evaluations a function $f \in \mathcal{F}$ can achieve on a sample $S = \{(z_i)\}_{i=1}^m$ i.e. $\mathcal{F} \circ S = \{(f(z_1), ..., f(z_m)) : f \in \mathcal{F}\}$. The empirical Rademacher complexity of $\mathcal{F}$ given S is :*

$$\mathcal{R}(\mathcal{F} \circ S) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^m \sigma_i f(z_i) \right], \tag{10}$$

where $\sigma_i \in \{\pm 1\}, i = 1, \dots, m$ are independent and identically distributed uniform random variables, i.e. $Pr(\sigma_i = 1) = Pr(\sigma_i = -1) = \frac{1}{2}, i = 1, \dots, m$.
More generally, given a set of vectors $A \subset \mathbb{R}^m$, we define its empirical Rademacher average :

$$\mathcal{R}(A) = \mathbb{E}_\sigma \left[\sup_{a \in A} \frac{1}{m} \sum_{i=1}^m \sigma_i a_i \right], \tag{11}$$

Rademacher complexity measures the ability of a hypothesis set of functions $\mathcal{H}$ to fit random ± 1 binary label assignments. In the case of binary classification, $h(x_i) \in \{\pm 1\}$, therefore, if $\sigma_i h(x_i) = 1$ the classification is correct and if $\sigma_i h(x_i) = -1$ the classification is incorrect hence the sum is maximised in 10.

Work has been done to derive data-dependent risk bounds expressed in terms of this quantity [4]. The following generalization error bound [22] can be derived where we assume that for all $(\mathbf{x}, y) \sim \mathcal{D}$ and $h \in \mathcal{H}$ we have that $|L(y, h(x))| < c$:

$$\mathbf{P}_{S \sim \mathcal{D}^m} \left[\sup_{h \in \mathcal{H}} |\mathcal{L}_\mathcal{D}(h) - \hat{\mathcal{L}}_\mathcal{S}(h)| \leq 2\mathcal{R}(\mathcal{L} \circ \mathcal{H} \circ S) + 4c\sqrt{\frac{2\ln(4/\delta)}{m}} \right] \geq 1 - \delta \tag{12}$$

Where $\mathcal{R}(\mathcal{L} \circ \mathcal{H} \circ S)$ is the empirical Rademacher complexity of $\mathcal{L} \circ \mathcal{H}$ over $S = \{(\mathbf{x_i}, y_i)\}_{i=1}^m$, where $\mathcal{L} \circ \mathcal{H} := \{(\mathbf{x}, y) \in S \mapsto \mathcal{L}(y, h(\mathbf{x})) : h \in \mathcal{H}\}$, or identically the empirical Rademacher average of the set of vectors $\mathcal{L} \circ \mathcal{H} \circ S = \{(\mathcal{L}(y_i, h(x_i)))_{i=1}^m : h \in \mathcal{H}\} \subseteq \mathbb{R}^m$ where $(\mathbf{x_i}, y_i)$ are the input points in S.
The bound given by this theorem is data-dependent, that is the bound depends on a quantity derivable from S : S is used both for learning a hypothesis from $\mathcal{H}$ and for estimating the quality of it.
The optimality of this bound for data-dependent uniform generalization gap bounds depends on the rate of this Rademacher complexity with m. For deep neural networks, upper bounds on Rademacher complexity often rely on a bound on the weight norms of the network. [8] gives, assuming norm constraints on their weight matrices and under several assumptions, a bound independent from the height and the width of the network on the Rademacher complexity ($\mathcal{R}(\mathcal{L} \circ \mathcal{H} \circ S)$). This bound is upper bounded by

$$\tilde{\mathcal{O}}\left(R\sqrt{\frac{\log\{\frac{R}{\Gamma}\}}{\sqrt{m}}} \right), \tag{13}$$

where R is an upper bound on the product of the Frobenius norms of the network's weight matrices and Γ is a lower bound on the product of the spectral norms of the network's weight matrices and $\tilde{\mathcal{O}}$ means an upper bound up to a logarithmic factor.

Effective capacity of DNNs. In order to assess the effective model capacity of feed forward neural networks, [27] explored, through a series of experiments, the behaviour of several deep neural network architectures for an image classification task with different levels of corrupted training data (random labels, shuffled labels, shuffled pixels, random pixels, random pixels generated using a Gaussian distribution with mean and variance matching the original labels), leaving all other properties of the models unchanged. The results of these experiments are depicted in figure 2.

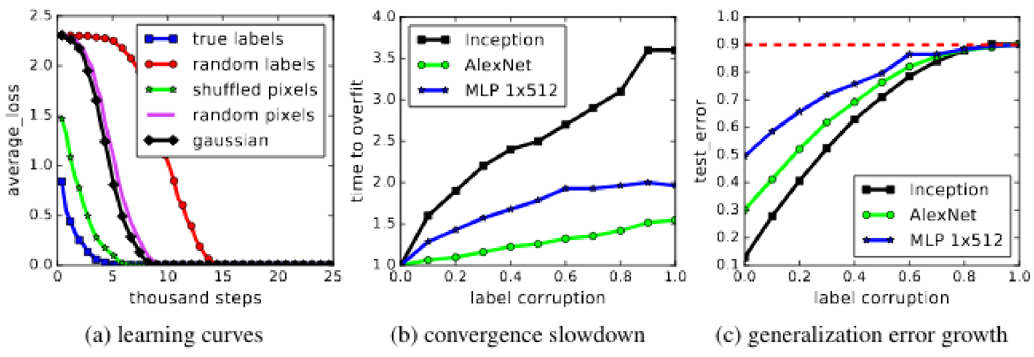


Figure 2: (a) shows the training loss across the different data randomizations along with the training steps. (b) shows the convergence time with different ratios of label corruption. (c) shows the test error (generalization error since the train error is reduced to 0) with different ratios of label corruption. Experiments by [27].

The different deep architectures tested were able to perfectly fit the training set even when breaking the relationship between the images and the labels (i.e. the underlying distribution $\mathcal{D}$). Moreover, optimization on corrupted data remains easy. Once the model starts learning it converges quickly, and training time increases by a small constant factor compared with training on the true labels. In parallel, a deterioration of the generalisation error is obviously observed. This proves that *deep neural networks are able to represent functions that generalize badly*.

As said earlier Rademacher complexity measures the ability of a hypothesis set of functions $\mathcal{H}$ to fit random ± 1 binary label assignments. Given the configuration of the different randomizations that were operated on the training data in the above-mentioned experiments, Rademacher complexity bounds are vacuous and useless for realistic setups. Unlike for classical machine learning algorithms, this measure of complexity is unable to explain the generalization properties of neural networks.

6 Algorithm-Dependent Bounds

As previously shown, bounding the generalization for the entire hypothesis class $\mathcal{H}$ fails, whether it is dependant or independent of the data distribution, since it is a bound on the worst case scenario. But there remains a faint hope that by restricting the bounds on only those hypotheses picked up by the learning algorithm, one can tighten these bounds.

6.1 Taking Into Account SGD

Taking into account the learning algorithm was first suggested by empirical studies [27, 18] who hypothesize the existence of an implicit regularization mechanism.
A number of different research directions have spawned in response to these findings. The first analytical result given by [6], which showed that for two layer neural networks with the second layer fixed, under very specific conditions on the activation function being the leaky ReLU $\phi(x) = \max\{\alpha x, x\}$ where $0 < \alpha < 1$, the data distribution being linearly separable by a vector $\mathbf{w}^* \in \mathbb{R}^d$ and using the mean hinge loss, Stochastic gradient descent (SGD) with step size η, can avoid overfitting even in the over-parameterized regime.
Specifically, they show that SGD converges to a global optimum while making at most:

$$M = \frac{\|\mathbf{w}^*\|^2}{\alpha^2} + \mathcal{O}\left(\frac{\|\mathbf{w}^*\|^2}{\min\{\eta, \sqrt{\eta}\}}\right) \tag{14}$$

non-zero update steps, see [6], corrolary 5.2.
They then show that the model learned by SGD will have a generalization error of $\mathcal{O}\left(\frac{M \log m}{m}\right)$ see [6], theorem 4. Thus for fixed $\|\mathbf{w}^*\|$ and η we obtain a guarantee that is independent of the network size. Their results imply that, given a sufficiently large number of training samples, SGD converges to a global minimum with good generalization behaviour avoiding global minima which overfit the training set.

Remark 5. *The above bound is optimal for $\eta \mapsto \infty$, which is unique to the restrictif setting considered.*

For similarly restricted deep neural networks, [24] indicate that SGD converges to the solution with minimal weight norm.

Following these findings, a huge variety of novel and refined, *algorithm-dependent* generalization bounds for deep networks have been developed. As summarized in appendix B, most bounds are either numerically large, grow with the size of the network or hold on a modified version of the network.

6.2 Norm Based Bounds

As we stated in section 4, a fundamental requirement for a generalization bound is that it should decrease with the data-set size m. Many recent works, as presented in table 2, tightened their bounds using norm-based quantities.

The margin based bounds presented are of the form :

$$L_0(f_{\mathbf{w}}) \leq \widehat{L}_\gamma(f_{\mathbf{w}}) + \text{generalization error bound.} \tag{15}$$

Where $L_\gamma(f_{\mathbf{w}})$ is the margin loss as defined in section 3.

[20] derive a generalization bound for feedforward neural networks with ReLU activations in terms of the product of the spectral norms and the Frobenius norm of the weights.

Theorem 6 (Generalization Bound [20])**.** *For any $B, d, h > 0$, let $f_{\mathbf{w}} : \mathcal{X}_{B,n} \to \mathbb{R}^k$ be a d-layer feedforward network with ReLU activations. Then, for any $\delta, \gamma > 0$, with probability $\geq 1 - \delta$ over a training set of size m, for any $\mathbf{w}$, we have:*

$$L_0(f_{\mathbf{w}}) \leq \widehat{L}_\gamma(f_{\mathbf{w}}) + \mathcal{O}\left(\sqrt{\frac{B^2 d^2 h \ln(dh) \Pi_{i=1}^d \|W_i\|_2^2 \sum_{i=1}^d \frac{\|W_i\|_F^2}{\|W_i\|_2^2} + \ln \frac{dm}{\delta}}{\gamma^2 m}}\right).$$

By investigating a new complexity measure for neural networks, [3] proved a margin bound. It's corresponding generalization error bound term in equation 15 (ignoring logarithmic factors) is of the form :

$$\mathcal{O}\left(\frac{Bd\sqrt{h}}{\gamma\sqrt{m}} \prod_{k=1}^{d} \|W_k\|_2 \times \frac{1}{d\sqrt{h}} \left(\sum_{k=1}^{d} \left(\frac{\|W_k - Z_k\|_{2,1}}{\|W_k\|_2}\right)^{2/3}\right)^{3/2}\right)$$

Where $\{W_i\}_{i=1}^d$ are the learned weights and $\{Z_i\}_{i=1}^d$ is the random initialization of the networks parameters.

Remark 7. *Both the stated bounds are very similar and depend on the spectral norm of the weights $\|W_k\|_2$, this quantity is key to showing their dependence on m.*

In the following, we will present the work in [16] showing empirically that these bounds violate criterion 5, figure 3. Indeed, we observe that these norm-based quantities increase with m, yielding vacuous bounds.

Setup. Fully connected networks of depth $d = 5$, width $h = 1024$ trained on MNIST by SGD with learning rate 0.1 and batch size 1. The network is trained until it achieves margin $\gamma^* \geq 10$ for 99% of the training data.

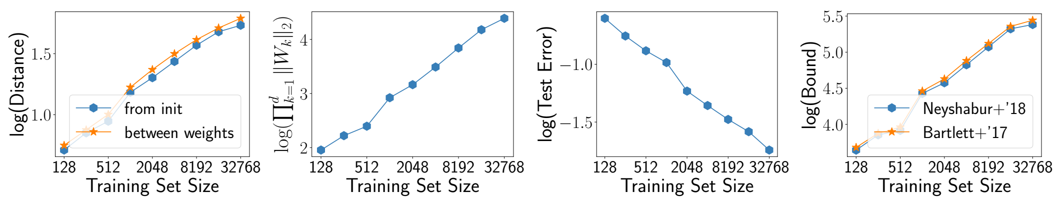


Figure 3: The **first** figure, shows both the ℓ_2 the distance of the network from the initialization and the ℓ_2 distance between the weights learned on two random draws of training data starting from the same initialization. The **second** figure shows the product of spectral norms of the weights matrices. The **third** figure shows the test error. The **last** figure shows the bounds from [20, 3].

Test error. As expected, the test error (third plot) decreases with dataset size m.

Norm based quantities. The product of the spectral norms of the network's weights matrices (on which depends bounds from [20, 3]) as well as the ℓ_2 distance of the weights from their initialization (on which depends bounds from [7, 15]) are represented by the blue plots in the first two figures, both increase at a polynomial rate with m. Another norm-based quantity is examined. The ℓ_2 diameter of the smallest ball encompassing all parameters explored by SGD for all possible initialization and all training set draws of $\mathcal{D}^m$. This quantity is lower bounded by the distance from the same initialization of weights matrices learned from two independent draws of training sets from $\mathcal{D}^m$ (orange plot in the first figure). It shows the same behaviour as the other two quantities.

Bounds from [20, 3]. As said earlier the network is trained until it achieves margin $\gamma^* \geq 10$ for 99% of the training data. By plugging $\gamma = 10$, we get that $\widehat{L}_{10}(f_{\mathbf{w}})$ is at most 0.01. The generalisation error bounds are then plotted as shown in the forth figure. They both grow with data-set size. This is because the norm-based quantities used in these bounds, introduced a dependence on m in the numerator.

Other settings of γ were further tested in [16]. Specifically, the median margin on the training set for which $\widehat{L}_{\gamma_{median}}(f_{\mathbf{w}})$ equals 0.5. However, a similar dependence on m is observed (the median margin doesn't grow as fast with m as the norm-based quantities). Other bounds depending on the same norm-based quantities are expected to behave similarly.

Observing how difficult it seems to come up with a generalization bound that meets all the criteria, [16] question the effectiveness of the underlying tool behind the above bounds, *uniform convergence*. They present a setup were any form of algorithm dependent uniform convergence bounds *provably* fails at explaining the generalization properties of deep learning.

7 Provable Failure of Uniform Convergence

We now get into the heart of this course, recall that our goal is to rigorously and thoroughly **rule out** the possibility that any kind of uniform convergence bound, however cleverly applied, can explain generalization in our setting of interest (which we will describe later).

7.1 Tightest Algorithm-Dependent Uniform Convergence Bound

To do this, let us define the tightest possible uniform convergence bound by tightening ϵ_{unif} to *only* those hypotheses that are picked by algorithm $\mathcal{A}$ under $\mathcal{D}$, excluding everything else.

It is helpful to first rephrase definition 2 of ϵ_{unif} to an *equivalent* form:
we can say that $\epsilon_{\text{unif}}(m, \delta)$ is the smallest value for which there exists a set of sample sets $\mathcal{S}_\delta \subseteq (\mathcal{X} \times \{-1, 1\})^m$ for which :

$$\mathbb{P}_{S \sim \mathcal{D}^m}[S \in \mathcal{S}_\delta] \geq 1 - \delta \tag{16}$$

hence

$$\sup_{S \in \mathcal{S}_\delta} \sup_{h \in \mathcal{H}} |\mathcal{L}_{\mathcal{D}}(h) - \hat{\mathcal{L}}_S(h)| \leq \epsilon_{\text{unif}}(m, \delta) \tag{17}$$

We are now ready to define the tightest uniform convergence bound by replacing $\mathcal{H}$ in 17 with only those hypotheses that are explored by the algorithm $\mathcal{A}$ under the datasets belonging to $\mathcal{S}_\delta$.

Definition 8. *The* ***tightest algorithm-dependent uniform convergence bound*** *with respect to loss $\mathcal{L}$ is the smallest value $\epsilon_{\text{unif-alg}}(m, \delta)$ for which there exists a set of sample sets $\mathcal{S}_\delta$ such that :*

$$\mathbb{P}_{S \sim \mathcal{D}^m}[S \in \mathcal{S}_\delta] \geq 1 - \delta$$

and for the space of hypotheses explored by $\mathcal{A}$ on $\mathcal{S}_\delta$ defined as :

$$\mathcal{H}_\delta := \bigcup_{S \in \mathcal{S}_\delta} \{h_S\} \subseteq \mathcal{H}$$

the following holds:

$$\sup_{S \in \mathcal{S}_\delta} \sup_{h \in \mathcal{H}_\delta} \left| \mathcal{L}_{\mathcal{D}}(h) - \hat{\mathcal{L}}_S(h) \right| \leq \epsilon_{\text{unif-alg}}(m, \delta)$$

7.2 Setup

We will now describe the setup for which *uniform convergence* is made to fail :

Distribution $\mathcal{D}$: Each input $(\mathbf{x}_1, \mathbf{x}_2)$ is a $K + D$ dimensional vector where $\mathbf{x}_1 \in \mathbb{R}^K$ and $\mathbf{x}_2 \in \mathbb{R}^D$. $\mathbf{u} \in \mathbb{R}^K$ determines the centers of the classes such that $\|\mathbf{u}\|_2 = 1/\sqrt{m}$. The label y is drawn uniformly from $\{-1, +1\}$, and conditioned on y, we have $\mathbf{x}_1 = 2 \cdot y \cdot \mathbf{u}$ while $\mathbf{x}_2$ is sampled independently from $\mathcal{N}(0, \frac{32}{D} I)$.

Learning algorithm $\mathcal{A}$: We consider a linear classifier with weights $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2)$. The output is computed as $h(\mathbf{x}) = \mathbf{w}_1\mathbf{x}_1 + \mathbf{w}_2\mathbf{x}_2$. Assume the weights are initialized to origin. Given $S = \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m)}, y^{(m)})\}$, $\mathcal{A}$ takes a gradient step of learning rate 1 to maximize $y \cdot h(\mathbf{x})$ for each $(\mathbf{x}, y) \in S$. Regardless of the batch size, the learned weights would satisfy, $\mathbf{w}_1 = 2m\mathbf{u}$ and $\mathbf{w}_2 = \sum_i y^{(i)}\mathbf{x}_2^{(i)}$.

Remark 9. *We first show how uniform convergence could fail for linear classifiers trained using GD because linear models are arguably the simplest of classifiers and it is more natural to expect uniform convergence to yield poorer bounds in more complicated classifiers hence making this setup the most interesting.*

For the above setup, [16] give the following theorem.

Theorem 10. *In the setup above, for any $\epsilon, \delta > 0$ and $\delta < 1/4$, let D be sufficiently large that it satisfies*

$$D \geq \frac{1}{c_1} \ln \frac{6m}{\delta}, \tag{18}$$

$$D \geq m \left(\frac{4c_4c_3}{c_2^2}\right)^2 \ln \frac{6m}{\delta}, \tag{19}$$

$$D \geq m \left(\frac{4c_4c_3}{c_2^2}\right)^2 \cdot 2\ln \frac{2}{\epsilon}, \tag{20}$$

then we have that for all $\gamma \geq 0$, for the $\mathcal{L}^{(\gamma)}$ loss,

$$\epsilon_{\textit{unif-alg}}(m, \delta) \geq 1 - \epsilon_{\textit{gen}}(m, \delta)$$

Specifically, for $\gamma \in [0, 1]$, $\epsilon_{\textit{gen}}(m, \delta) \leq \epsilon$, and so $\epsilon_{\textit{unif-alg}}(m, \delta) \geq 1 - \epsilon$

With constants, $c_1 = 1/32$, $c_2 = 1/2$ and $c_3 = 3/2$ and $c_4 = \sqrt{2}$

The theorem above states that despite the good generalisation properties of the algorithm $\mathcal{A}$, the tightest uniform convergence bound is loose. In other words even if ϵ_{gen} is smaller than ϵ, $\epsilon_{\text{unif-alg}}$ is at least greater than $1 - \epsilon$.

Interpretation : Theorem 10 rules out any possible pruning of the hypothesis class $\mathcal{H}$ to yield good uniform convergence guarantees, even after fixing $\mathbf{w}_1$ to the learned value ($2m\mathbf{u}$) and for any possible $1 - \delta$ truncation of the Gaussian $\mathbf{w}_2$, the resulting pruned class of weights – despite all of them having a test error less than ϵ – would give only nearly vacuous uniform convergence bounds as $\epsilon_{\text{unif-alg}}(m, \delta) \geq 1 - \epsilon$.

7.3 Proof

Useful lemmas are relegated to appendix A.
To prove theorem 10, we first upper bound the generalization error in Lemma 1, and we lower bound the uniform convergence bound in Lemma 2.

We first prove that the above algorithm generalizes well with respect to the losses corresponding to $\gamma \in [0, 1]$.

Lemma 1. *In the setup in subsection 7.2, when $\gamma \in [0, 1]$, for $\mathcal{L}^{(\gamma)}$, $\epsilon_{\textit{gen}}(m, \delta) \leq \epsilon$.*

Proof. First for the training data, we argue that both $\mathbf{w}_1$ and a small part of the noise vector $\mathbf{w}_2$ align along the correct direction, while the remaining part of the high-dimensional noise vector are orthogonal to the input; this leads to correct classification of the training set.

Since the trainning is done using gradient descent of step 1, the parameters learned by our algorithm satisfy $\mathbf{w}_1 = 2m \cdot \mathbf{u}$ and $\mathbf{w}_2 = \sum y^{(i)}\mathbf{x}_2^{(i)} \sim \mathcal{N}(0, \frac{8m}{c_2^2 D})$.

We have from Corollary 1 that with probability $1-\frac{\delta}{3m}$ over the draws of $\mathbf{x}_2^{(i)}$, as long as $\frac{\delta}{3m} \geq 2e^{-c_1 D}$ (which is given to hold by Equation 18),

$$c_2 \leq \frac{1}{2\sqrt{2}} c_2 \|\mathbf{x}_2^{(i)}\| \leq c_3. \tag{21}$$

Next, for a given $\mathbf{x}^{(i)}$, we have from Corollary 2, with probability $1 - \frac{\delta}{3m}$ over the draws of $\sum_{j\neq i} y^{(j)}\mathbf{x}_2^{(j)}$,

$$|\mathbf{x}_2^{(i)} \cdot \sum_{j\neq i} y^{(j)}\mathbf{x}_2^{(j)}| \leq c_4 \|\mathbf{x}_2^{(i)}\| \frac{2\sqrt{2}\cdot\sqrt{m}}{c_2\sqrt{D}}\sqrt{\ln\frac{6m}{\delta}}. \tag{22}$$

Then, with probability $1 - \frac{2}{3}\delta$ over the draws of the training dataset we have for all i,

$$\begin{aligned}
y^{(i)}h(\mathbf{x}^{(i)}) &= y^{(i)}\mathbf{w}_1 \cdot \mathbf{x}_1^{(i)} + y^{(i)} \cdot y^{(i)}\|\mathbf{x}_2^{(i)}\|^2 + y^{(i)} \cdot \mathbf{x}_2^{(i)} \cdot \sum_{j\neq i} y^{(j)}\mathbf{x}_2^{(j)} \\
&= 4 + \underbrace{\|\mathbf{x}_2^{(i)}\|^2}_{\text{apply Equation 21}} + \underbrace{y^{(i)}\mathbf{x}_2^{(i)} \cdot \sum_{j\neq i} y^{(j)}\mathbf{x}_2^{(j)}}_{\text{apply Equation 22}} \\
&\geq 4 + 4\cdot 2 - c_4 \frac{2\sqrt{2}c_3}{c_2} \cdot \underbrace{\frac{2\sqrt{2}\cdot\sqrt{m}}{c_2\sqrt{D}}\sqrt{\ln\frac{6m}{\delta}}}_{\text{apply Equation 19}} \\
&\geq 4 + 8 - 2 = 10 > 1.
\end{aligned} \tag{23}$$

Thus, for all $\gamma \in [0, 1]$, the $\mathcal{L}^{(\gamma)}$ loss of this classifier on the training data-set S is zero.

We now show that for most test data, the classification is correct, and hence the test and generalization error are both small.

From Corollary 1, with probability $1 - \frac{\delta}{3}$ over the draws of the training data, we also have that, as long as $\frac{\delta}{3m} \geq 2e^{-c_1 D}$ (which is given to hold by Equation 18),

$$c_2\sqrt{m} \leq \frac{1}{2\sqrt{2}} c_2 \|\sum y^{(i)}\mathbf{x}_2^{(i)}\| \leq c_3\sqrt{m}. \tag{24}$$

Next, conditioned on the draw of S and the learned classifier, for any $\epsilon' > 0$, with probability $1 - \epsilon'$ over the draws of a test data point, $(\mathbf{z}, y)$, we have from Corollary 2 that

$$|\mathbf{z}_2 \cdot \sum y^{(i)}\mathbf{x}_2^{(i)}| \leq c_4 \|\sum y^{(i)}\mathbf{x}_2^{(i)}\| \cdot \frac{2\sqrt{2}}{c_2\sqrt{D}} \cdot \ln\frac{1}{\epsilon'}. \tag{25}$$

Using this, we have that with probability $1 - 2\exp\left(-\frac{1}{2}\left(\frac{c_2^2}{4c_4c_3}\sqrt{\frac{D}{m}}\right)^2\right)$ over the draws of a test data point, $(\mathbf{z}, y)$,

$$\begin{aligned}
yh(\mathbf{x}) &= y\mathbf{w}_1 \cdot \mathbf{z}_1 + \underbrace{y \cdot \mathbf{z}_2 \cdot \sum_j y^{(j)}\mathbf{x}_2^{(j)}}_{\text{apply Equation 25}} \\
&\geq 4 - c_4 \underbrace{\|\sum y^{(i)}\mathbf{x}_2^{(i)}\|}_{\text{apply Equation 24}} \cdot \frac{2\sqrt{2}}{c_2\sqrt{D}} \frac{c_2^2}{4c_4c_3}\sqrt{\frac{D}{m}} \\
&\geq 4 - 2 \geq 2.
\end{aligned} \tag{26}$$

Thus, we have that for $\gamma \in [0, 1]$, the $\mathcal{L}^{(\gamma)}$ loss of the classifier on the distribution $\mathcal{D}$ is $2\exp\left(-\frac{1}{2}\left(\frac{c_2^2}{4c_4c_3}\sqrt{\frac{D}{m}}\right)^2\right)$ which is at most ϵ as assumed in Equation 20.
In other words, the absolute difference between the distribution loss and the train loss is at most ϵ and this holds for at least $1 - \delta$ draws of the samples S. Then, by the definition of ϵ_{gen} we have the result. □

The main idea behind the proof of the lower bound is to show that for any choice of $\mathcal{S}_\delta$ as required by the definition of $\epsilon_{\text{unif-alg}}$, we can always find an $S_\star$ and its noise-negated version $S'_\star$ both of which belong to $\mathcal{S}_\delta$. Furthermore, we can show that $h_{S_\star}$ has small test error but high empirical error on $S'_\star$, and that this leads to a nearly vacuous uniform convergence bound.

Lemma 2. *In the setup in subsection 7.2, for any $\epsilon > 0$ and for any $\delta \leq 1/4$, and for the same lower bounds on D, and for any $\gamma \geq 0$, we have that*

$$\epsilon_{\textit{unif-alg}}(m, \delta) \geq 1 - \epsilon_{\textit{gen}}(m, \delta)$$

for the $\mathcal{L}^{(\gamma)}$ loss.

Proof. For any S, let S' denote the set of noise-negated samples.

$$S' = \{((\mathbf{x}_1, -\mathbf{x}_2), y) \mid ((\mathbf{x}_1, \mathbf{x}_2), y) \in S\} \tag{27}$$

We first show with high probability $1 - 2\delta/3$ over the draws of S, that the classifier learned on S, misclassifies S' completely. The proof for this is nearly identical to our proof for why the training loss is zero, except for certain sign changes. For any $\mathbf{x}^{(i)}_{\text{neg}} = (\mathbf{x}^{(i)}_1, -\mathbf{x}^{(i)}_2)$, we have

$$\begin{aligned}
y^{(i)} h(\mathbf{x}^{(i)}_{\text{neg}}) &= y^{(i)} \mathbf{w}_1 \cdot \mathbf{x}^{(i)}_1 - y^{(i)} \cdot y^{(i)} \|\mathbf{x}^{(i)}_2\|^2 - y^{(i)} \cdot \mathbf{x}^{(i)}_2 \cdot \sum_{j \neq i} y^{(j)} \mathbf{x}^{(j)}_2 \\
&= 4 - \underbrace{\|\mathbf{x}^{(i)}_2\|^2}_{\text{apply Equation 21}} \underbrace{- y^{(i)} \mathbf{x}^{(i)}_2 \cdot \sum_{j \neq i} y^{(j)} \mathbf{x}^{(j)}_2}_{\text{apply Equation 22}} \\
&\leq 4 - 4 \cdot 2 + c_4 \frac{2\sqrt{2} c_3}{c_2} \cdot \underbrace{\frac{2\sqrt{2} \cdot \sqrt{m}}{c_2 \sqrt{D}} \ln \frac{3m}{\delta}}_{\text{apply Equation 19}} \\
&\leq 4 - 8 + 2 = -2 < 0.
\end{aligned}$$

Since the learned hypothesis misclassifies all of S', it has loss of 1 on S', thus :

$$\mathbb{P}_{S \sim \mathcal{D}^m} \left[\hat{\mathcal{L}}_{S'}(h_S) = 1 \right] \geq 1 - 2\delta/3 \tag{28}$$

Now recall that, by definition, to compute $\epsilon_{\text{unif-alg}}$, one has to pick a sample set space $\mathcal{S}_\delta$ of mass $1 - \delta$ i.e., $\mathbb{P}_{S \sim \mathcal{S}^m}[S \in \mathcal{S}_\delta] \geq 1 - \delta$. We first argue that for *any* choice of $\mathcal{S}_\delta$, there must exist a 'bad' $S_\star$ such that :

1. $S_\star \in \mathcal{S}_\delta$
2. $S'_\star \in \mathcal{S}_\delta$
3. $h_{S_\star}$ has test error less than $\epsilon_{\text{gen}}(m, \delta)$
4. $h_{S_\star}$ completely misclassifies $S'_\star$

We show the existence of such an $S_\star$, by arguing that over the draws of S, there is non-zero probability of picking an S that satisfies all the above conditions. Specifically, we have by the union bound that

$$\begin{aligned}
&\mathbb{P}_{S \sim \mathcal{D}^m} \left[S \in \mathcal{S}_\delta, S' \in \mathcal{S}_\delta, \mathcal{L}_{\mathcal{D}}(h_S) \leq \epsilon_{\text{gen}}(m, \delta), \hat{\mathcal{L}}_{S'}(h_S) = 1 \right] \\
&\geq 1 - \mathbb{P}_{S \sim \mathcal{D}^m} [S \notin \mathcal{S}_\delta] - \mathbb{P}_{S \sim \mathcal{D}^m} [S' \notin \mathcal{S}_\delta] \\
&- \mathbb{P}_{S \sim \mathcal{D}^m} [\mathcal{L}_{\mathcal{D}}(h_S) > \epsilon_{\text{gen}}(m, \delta)] - \mathbb{P}_{S \sim \mathcal{D}^m} \left[\hat{\mathcal{L}}_{S'}(h_S) \neq 1 \right].
\end{aligned} \tag{29}$$

We have,

$$\begin{aligned}
&\mathbb{P}_{S \sim \mathcal{D}^m} [S \notin \mathcal{S}_\delta] \leq \delta, && \text{by definition of } \mathcal{S}_\delta \\
&\mathbb{P}_{S \sim \mathcal{D}^m} [\mathcal{L}_{\mathcal{D}}(h_S) > \epsilon_{\text{gen}}(m, \delta)] \leq \delta && \text{by definition of } \epsilon_{\text{gen}} \\
&\mathbb{P}_{S \sim \mathcal{D}^m} \left[\hat{\mathcal{L}}_{S'}(h_S) \neq 1 \right] \leq 2\delta/3, && \text{given by 28} \\
&\mathbb{P}_{S \sim \mathcal{D}^m} [S' \notin \mathcal{S}_\delta] \leq \delta, && \text{given by } S' \sim \mathcal{D}^m (\text{ isotropic Gaussian noise})
\end{aligned}$$

Recall that δ is assumed $< 1/4$, we thus have,

$$\mathbb{P}_{S \sim \mathcal{D}^m} \left[S \in \mathcal{S}_\delta, S' \in \mathcal{S}_\delta, \mathcal{L}_{\mathcal{D}}(h_S) \leq \epsilon_{\text{gen}}(m, \delta), \hat{\mathcal{L}}_{S'}(h_S) = 1 \right] > 1 - 4\delta > 0 \tag{31}$$

This implies that for any given choice of $\mathcal{S}_\delta$, there exists $S_\star$ that satisfies our requirement. Then, from the definition of $\epsilon_{\text{unif-alg}}(m, \delta)$, we essentially have that,

$$\begin{aligned}\epsilon_{\text{unif-alg}}(m, \delta) &= \sup_{S \in \mathcal{S}_\delta} \sup_{h \in \mathcal{H}_\delta} |\mathcal{L}_\mathcal{D}(h) - \hat{\mathcal{L}}_S(h)| \\ &\geq |\mathcal{L}_\mathcal{D}(h_{S_\star}) - \hat{\mathcal{L}}_{S'_\star}(h_{S_\star})| = |\epsilon - 1| = 1 - \epsilon.\end{aligned}$$

□

7.4 Nearly Vacuous Bounds for Any $\gamma > 0$

From theorem 10, it is obvious that for $\gamma \leq 1$, the approach in 7.2 would yield vacuous bounds. We now establish that this is the case even for $\gamma > 1$.

The following inequality is used to derive a bound on the 0-1 error :

$$\mathcal{L}^{(0)}_\mathcal{D}(h_S) \underbrace{\leq}_{\text{holds over all draws of } S} \mathcal{L}^{(\gamma)}_\mathcal{D}(h_S) \underbrace{\leq}_{\text{holds with at least} 1-\delta \text{ over draws of } S} \hat{\mathcal{L}}^{(\gamma)}_S(h_S) + \epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) \quad (32)$$

Where $\epsilon^{(\gamma)}_{\text{unif-alg}}, \epsilon^{(\gamma)}_{\text{gen}}$ denote the uniform convergence and generalization error for $\mathcal{L}^{(\gamma)}$ loss.

To establish that uniform convergence is nearly vacuous in any setting of γ, we must show that the right hand side of the above bound is nearly vacuous for any choice of $\gamma \geq 0$.

Proposition 11. *Given that for all $\gamma \geq 0$, $\epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) \geq 1 - \epsilon^{(\gamma)}_{\text{gen}}(m, \delta)$ then, we then have that for all $\gamma \geq 0$,*

$$\mathbb{P}_{S \sim \mathcal{D}^m}\left[\hat{\mathcal{L}}^{(\gamma)}_S(h_S) + \epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) \geq \frac{1}{2}\right] > \delta$$

or in other words, the guarantee from the right hand side of Equation 32 is nearly vacuous.

Remark 12. *We established the relation $\epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) \geq 1 - \epsilon^{(\gamma)}_{\text{gen}}(m, \delta)$ to be true in all of our setups.*

Proof. Assume on the contrary that for some choice of γ, we have,

$$\mathbb{P}_{S \sim \mathcal{D}^m}\left[\hat{\mathcal{L}}^{(\gamma)}_S(h_S) + \epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) < \frac{1}{2}\right] \geq 1 - \delta$$

This means that $\epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) < 1/2$. Furthermore, this also means that with probability at least $1 - \delta$ over the draws of S, $\hat{\mathcal{L}}^{(\gamma)}_S(h_S) < 1/2$ and $\mathcal{L}^{(\gamma)}_\mathcal{D}(h_S) < 1/2$ (which follows from the second inequality in Equation 32).

As a result, we have,

$$\mathbb{P}_{S \sim \mathcal{D}^m}\left[\mathcal{L}^{(\gamma)}_\mathcal{D}(h_S) - \hat{\mathcal{L}}^{(\gamma)}_S(h_S) < 1/2\right] \geq 1 - \delta$$

In other words, $\epsilon^{(\gamma)}_{\text{gen}}(m, \delta) < 1/2$. Since we are given that $\epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) \geq 1 - \epsilon^{(\gamma)}_{\text{gen}}(m, \delta)$, by our upper bound on the generalization error, we have $\epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) \geq 1/2$, which is a contradiction to our earlier inference that $\epsilon^{(\gamma)}_{\text{unif-alg}}(m, \delta) < 1/2$. Hence, our assumption is wrong.

□

7.5 Failure of Hypothesis-Dependent Uniform Convergence Bounds

Failure of the above setups are extended by [16] to hypothesis-dependent bounds where the capacity term depends only on the algorithm's output (learned hypothesis or the learned weights) with no other data dependence elsewhere in the bound. The form of such bounds may be : with high probability over draws of training set $\tilde{S}$, for any hypothesis h with weights $\mathbf{w}$,

$$\mathcal{L}_\mathcal{D}(h) - \hat{\mathcal{L}}_{\tilde{S}}(h) \leq \frac{\|\mathbf{w}\|_2}{\sqrt{m}}.$$

In order to better capture this, we redefine $\epsilon_{\text{unif-alg}}$ so that it depends on the learned hypothesis. Similarly to definition 8, we define the $\mathcal{S}_\delta$ for which $\mathbb{P}_{\tilde{S} \sim \mathcal{D}^m}[\tilde{S} \notin \mathcal{S}_\delta] \leq \delta$. $\epsilon_{\text{unif-alg}}(h_{\tilde{S}}, m, \delta)$ is then an upper bound on the generalization gap of $h_{\tilde{S}}$ as follows :

$$\epsilon_{\text{unif-alg}}(h_{\tilde{S}}, m, \delta) := \sup_{\tilde{S} \in \mathcal{S}_\delta} |\mathcal{L}_D(h_{\tilde{S}}) - \hat{\mathcal{L}}_S(h_{\tilde{S}})|. \quad (33)$$

This tightest hypothesis-dependent uniform convergence bound is the difference between the expected error and the empirical error of the hypothesis $h_{\tilde{S}}$ computed over sets of S_δ.
As stated in proof of theorem 10 in subsection 7.3. We have that for at least $1 - O(\delta)$ draws of the sample set $\tilde{S}$ there must exist a 'bad' dataset $\tilde{S}'$ such that :

1. $\tilde{S} \in \mathcal{S}_\delta$
2. $\tilde{S}' \in \mathcal{S}_\delta$
3. $h_{\tilde{S}}$ has test error less than $\epsilon_{\text{gen}}(m, \delta)$
4. $h_{\tilde{S}}$ completely misclassifies $\tilde{S}'$ i.e. $\hat{\mathcal{L}}_{\tilde{S}'}(h_{\tilde{S}}) = 1$

For all such $\tilde{S}$, by setting $S = \tilde{S}'$ in the definition of $\epsilon_{\text{unif-alg}}(h_{\tilde{S}}, m, \delta)$ in equation 33, we get that :

$$\epsilon_{\text{unif-alg}}(h_{\tilde{S}}, m, \delta) \geq |\mathcal{L}_D(h_{\tilde{S}}) - \hat{\mathcal{L}}_{\tilde{S}'}(h_{\tilde{S}})| \geq 1 - \epsilon_{\text{gen}}(m, \delta)$$

Hence, with probability at least $1 - O(\delta)$ over the draws of the training sets, hypothesis-dependent generalization bounds fail to explain generalization of the corresponding hypothesis.

7.6 ReLU Neural Network

This section illustrates that the effects we modeled theoretically in the linear classifier *are* indeed reflected in typical training settings.
We now design a non-linearly separable task (with no "noisy" dimensions as in 10) where a sufficiently wide ReLU network trained in the standard manner leads to failure of uniform convergence.

Remark 13. *For this argument, we will rely on a classifier trained* empirically*, in contrast to the linear setup in 10 where we rely on an analytically derived expression for the learned classifier. We also refer the reader to [16], Appendix F, where a closed form expression is derived for a neural network with exponential activation functions.*

Distribution. We consider two classes distributed uniformly over two origin-centered hyperspheres with radius 1 and 1.1 respectively. We vary the number of training examples from $4k$ to $65k$ (thus ranging through typical dataset sizes like that of MNIST). The dimensionality is set to 1000. Observe that compared to the linear example, this data distribution is more realistic in two ways. First, there is no specific dimensions in the data that are noisy and second, the data dimensionality is a constant less than m.

Learning algorithm. A two-layer, $d = 2$, ReLU network with width, $h = 100k$ to minimize cross entropy loss using SGD with learning rate 0.1 and batch size 64. the network is trained until 99% of the data is classified by a margin of $\gamma = 10$.

As shown in Figure 4, in this setup, the 0-1 error (i.e., $\mathcal{L}^{(0)}$) as approximated by the test set, decreases at the rate of $O(m^{-0.5})$.

To prove failure of uniform convergence, a dataset S' is constructed by projecting every data point onto the other hypershpere and then flipping the labels. Observe that $S' \sim \mathcal{D}^m$ because the distributions are uniform over the hyperspheres.

As observed in Figure 4, S' is completely misclassified by the learned network. Having established these facts, the rest of the argument follows similarly to the previous setting, implying failure of uniform convergence as in Theorem 10.

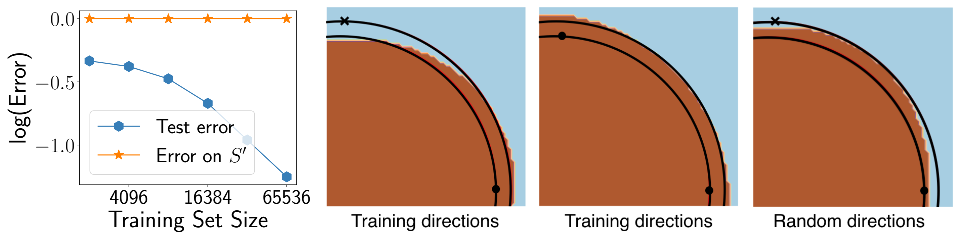


Figure 4: In the **first** plot, the 0-1 error on a test data set and on the 'bad' data set S' is plotted. In the **second** and **third** plots, the decision boundary rightly classifies the training points on the two hyperspheres, as can be seen, the decision boundary is skewed. On the **fourth** plot, two random test data points are represented along with the decision boundary located in between the hyperspheres (also confirmed by the low test error). Experiments by [16].

The decision boundary is skewed around the training data. In Figure 4, we visualize how the learned boundaries are skewed around the training data in a way that the projected points of S' are misclassified. This demonstrates how the boundary learned by the ReLU network is sufficiently complex that it hurts uniform convergence while not affecting the generalization error, at least in this setting.

Deep learning conjecture. In their work, [16] conjecture that in over-parameterized deep learning, SGD finds a fit that is simple at a macroscopic level (leading to good generalization) but also many microscopic fluctuations (hurting uniform convergence).

8 Take Home Message

Before concluding this course, let's quickly recap the main points:

- Algorithm-independent bounds are inherently limited.
- Norm-based bounds grow with the data set size.
- In multiple settings, **any** uniform convergence bound that only depends on the algorithm fails.

9 Conclusion

In this course, we explained how conventional methods fall short of solving generalization in deep learning and showed how recent research has been pursuing the wrong direction. More importantly, we present the findings of an influential work [16] that casts doubt on the most widely used tool in learning theory, uniform convergence. This course gives a general idea on the challenges that must be overcome to achieve the long term objective of explaining the behavior of deep learning models.

References

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. "Learning and Generalization in Overparameterized Neural Networks, Going Beyond Two Layers". In: *CoRR* abs/1811.04918 (2018). arXiv: 1811.04918. URL: http://arxiv.org/abs/1811.04918.

[2] Sanjeev Arora et al. "Stronger generalization bounds for deep nets via a compression approach". In: *CoRR* abs/1802.05296 (2018). arXiv: 1802.05296. URL: http://arxiv.org/abs/1802.05296.

[3] Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. "Spectrally-normalized margin bounds for neural networks". In: *CoRR* abs/1706.08498 (2017). arXiv: 1706.08498. URL: http://arxiv.org/abs/1706.08498.

[4] Peter L. Bartlett and Shahar Mendelson. "Rademacher and Gaussian Complexities: Risk Bounds and Structural Results". In: *Computational Learning Theory*. Ed. by David Helmbold and Bob Williamson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 224–240. ISBN: 978-3-540-44581-4.

[5] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[6] Alon Brutzkus et al. "SGD Learns Over-parameterized Networks that Provably Generalize on Linearly Separable Data". In: *CoRR* abs/1710.10174 (2017). arXiv: 1710.10174. URL: http://arxiv.org/abs/1710.10174.

[7] Gintare Dziugaite and Daniel Roy. "Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data". In: (Mar. 2017).

[8] Noah Golowich, Alexander Rakhlin, and Ohad Shamir. *Size-Independent Sample Complexity of Neural Networks*. 2019. arXiv: 1712.06541 [cs.LG].

[9] Nick Harvey, Christopher Liaw, and Abbas Mehrabian. "Nearly-tight VC-dimension bounds for piecewise linear neural networks". In: *CoRR* abs/1703.02930 (2017). arXiv: 1703.02930. URL: http://arxiv.org/abs/1703.02930.

[10] V. Koltchinskii and D. Panchenko. "Empirical Margin Distributions and Bounding the Generalization Error of Combined Classifiers". In: *Ann. Statist.* 30.1 (Feb. 2002), pp. 1–50. DOI: 10.1214/aos/1015362183. URL: https://doi.org/10.1214/aos/1015362183.

[11] Vladimir Koltchinskii. "Local Rademacher complexities and oracle inequalities in risk minimization". In: *Ann. Statist.* 34.6 (Dec. 2006), pp. 2593–2656. DOI: 10.1214/009053606000001019. URL: https://doi.org/10.1214/009053606000001019.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012, pp. 1097–1105. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[13] Yuanzhi Li and Yingyu Liang. "Learning Overparameterized Neural Networks via Stochastic Gradient Descent on Structured Data". In: *CoRR* abs/1808.01204 (2018). arXiv: 1808.01204. URL: http://arxiv.org/abs/1808.01204.

[14] Vaishnavh Nagarajan and J. Zico Kolter. "Deterministic PAC-Bayesian generalization bounds for deep networks via generalizing noise-resilience". In: *CoRR* abs/1905.13344 (2019). arXiv: 1905.13344. URL: http://arxiv.org/abs/1905.13344.

[15] Vaishnavh Nagarajan and J. Zico Kolter. *Generalization in Deep Networks: The Role of Distance from Initialization*. 2019. arXiv: 1901.01672 [cs.LG].

[16] Vaishnavh Nagarajan and J. Zico Kolter. "Uniform convergence may be unable to explain generalization in deep learning". In: *CoRR* abs/1902.04742 (2019). arXiv: 1902.04742. URL: http://arxiv.org/abs/1902.04742.

[17] Brady Neal et al. "A Modern Take on the Bias-Variance Tradeoff in Neural Networks". In: *CoRR* abs/1810.08591 (2018). arXiv: 1810.08591. URL: http://arxiv.org/abs/1810.08591.

[18] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. *In Search of the Real Inductive Bias: On the Role of Implicit Regularization in Deep Learning*. 2015. arXiv: 1412.6614 [cs.LG].

[19] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. "Norm-Based Capacity Control in Neural Networks". In: *CoRR* abs/1503.00036 (2015). arXiv: 1503.00036. URL: http://arxiv.org/abs/1503.00036.

[20] Behnam Neyshabur et al. “A PAC-Bayesian Approach to Spectrally-Normalized Margin Bounds for Neural Networks”. In: *CoRR* abs/1707.09564 (2017). arXiv: 1707.09564. URL: http://arxiv.org/abs/1707.09564.

[21] Behnam Neyshabur et al. “Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks”. In: *CoRR* abs/1805.12076 (2018). arXiv: 1805.12076. URL: http://arxiv.org/abs/1805.12076.

[22] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. DOI: 10.1017/CBO9781107298019.

[23] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[24] Daniel Soudry et al. “The Implicit Bias of Gradient Descent on Separable Data”. In: *Journal of Machine Learning Research* 19.70 (2018), pp. 1–57. URL: http://jmlr.org/papers/v19/18-188.html.

[25] V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition*. Moscow: Nauka, 1974.

[26] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019. DOI: 10.1017/9781108627771.

[27] Chiyuan Zhang et al. *Understanding deep learning requires rethinking generalization*. 2017. arXiv: 1611.03530 [cs.LG].

[28] Wenda Zhou et al. “Non-Vacuous Generalization Bounds at the ImageNet Scale: A PAC-Bayesian Compression Approach”. In: *arXiv e-prints*, arXiv:1804.05862 (Apr. 2018), arXiv:1804.05862. arXiv: 1804.05862 [stat.ML].

A Useful Lemmas

In this section, we state some standard results we will use in our proofs. We first define some constants: $c_1 = 1/2048$, $c_2 = \sqrt{15/16}$ and $c_3 = \sqrt{17/16}$ and $c_4 = \sqrt{2}$.

First, we state a tail bound for sub-exponential random variables [26].

Lemma 3. *For a sub-exponential random variable X with parameters (ν, b) and mean μ, for all $t > 0$,*

$$\mathbb{P}\left[|X - \mu| \geq t\right] \leq 2\exp\left(-\frac{1}{2}\min\left(\frac{t}{b}, \frac{t^2}{\nu^2}\right)\right).$$

As a corollary, we have the following bound on the sum of squared normal variables:

Corollary 1. *For $z_1, z_2, \ldots, z_D \sim \mathcal{N}(0, 1)$, we have that*

$$\mathbb{P}\left[\frac{1}{D}\sum_{j=1}^{D} z_j^2 \in [c_2^2, c_3^2]\right] \leq 2\exp(-c_1 D).$$

We now state the Hoeffding bound for sub-Gaussian random variable.

Lemma 4. *Let $z_1, z_2, \ldots, z_D$ be independently drawn sub-Gaussian variables with mean 0 and sub-gaussian parameter σ_i. Then,*

$$\mathbb{P}\left[\left|\sum_{d=1}^{D} z_d\right| \geq t\right] \leq 2\exp(-t^2/2\sum_{d=1}^{D}\sigma_d^2).$$

Again, we restate it as follows:

Corollary 2. *For any $\mathbf{u} = (u_1, u_2, \ldots, u_d) \in \mathbb{R}^D$, for $z_1, z_2, \ldots, z_D \sim \mathcal{N}(0, 1)$,*

$$\mathbb{P}\left[\left|\sum_{d=1}^{D} u_d z_d\right| \geq \|\mathbf{u}\|_2 \cdot c_4\sqrt{\ln\frac{2}{\delta}}\right] \leq \delta.$$

B Survey on Algorithm-Dependent Bounds

Bound	Norm dependencies	Parameter-count dependencies	Numerical value	Holds on original network?
[9]	-	depth $\times$ width	Large	Yes
[3] [20]	Product of spectral norms dist. from init. (not necessarily ℓ_2)	poly(width) exp(depth)	Large	Yes
[19] [8]	Product of Frobenius norms ℓ_2 dist. from init.	$\sqrt{\text{width}}^{\text{depth}}$	Very large	Yes
[14]	Jacobian norms ℓ_2 dist. from init. Inverse pre-activations	poly(width) poly(depth)	Inverse pre-activations can be very large	Yes
[21] for two-layer networks	Spectral norm (1st layer) ℓ_2 Dist. from init (1st layer) Frobenius norm (2nd layer)	$\sqrt{\text{width}}$	Small	Yes
[2]	Jacobian norms dist. from init.	poly(width) poly(depth)	Small	No. Holds on compressed network
[7]	dist. from init. Noise-resilience of network	-	Non-vacuous on MNIST	No. Holds on an optimized, stochastic network
[28]	Heuristic compressibility & noise-resilience of network	-	Non-vacuous on ImageNet	No. Holds on an optimized, stochastic, heuristically compressed, network
[1]	$L_{2,4}$ norm (1st layer) Frobenius norm (2nd layer)	-	Small for carefully scaled init. and learning rate	Yes
[13]	-	-	Small for carefully scaled batch size and learning rate	Yes

Table 2: Summary of generalization bounds for ReLU networks.