An Exposition on the Universal Law of Robustness

A central tenet of machine learning is the careful balancing of the bias–variance trade–off[1], yet this tenet is at odds with the methods used in standard machine learning practices. The trade–off implies that a model should balance under–fitting and over–fitting, but neural–networks are trained to exactly fit (i.e, interpolate) training data. Classically, models that interpolate should over–fit and not generalize well, but this does not arise in practice as they often obtain high accuracy on *test* data. Recently, a new unified performance curve, dubbed the "double–descent" curve, was discovered that reconciled classical understanding and modern practices. The curve extends the U–shaped bias–variance trade–off curve by showing how increasing model capacity beyond the point of interpolation results in improved performance.

More formally, in machine learning, given training data $(x_i, y_i)_{i=1}^n$, we want to find a function (predictor) $h \in \mathcal{H}$ that minimizes the empirical training risk, $\frac{1}{n}\sum \ell(h(x_i) - y_i)$, where ℓ is the loss function.[2] We also want h to perform well with new (test) data. This is where the challenge lies, balancing the seemingly orthogonal goals of minimizing the empirical risk and minimizing the test risk. Classical wisdom dictates controlling the capacity of the function class $\mathcal{H}$ based on the bias–variance trade–off by balancing under–fitting and over–fitting ([1]).

- $\mathcal{H}$ too small $\implies$ all predictors in $\mathcal{H}$ may under–fit the training data and not generalize well to test data.
- $\mathcal{H}$ too large $\implies$ the empirical risk minimizer may over–fit spurious patterns in the training data and thus not generalize well to test data.

That is, classical thinking is concerned with finding the "sweet spot", but machine learning practitioners do not care to find this balance. This disparity is what led to the discovery of the double–descent curve:

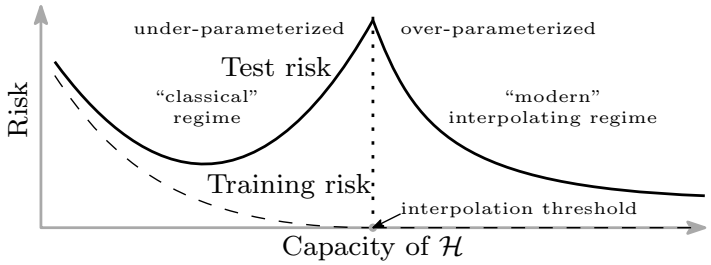


Figure 1: ([1]) **Training risk (dashed line) and test risk (solid line).** The double descent risk curve, which incorporates the U–shaped risk curve (i.e., the "classical" regime) together with the observed behavior from using high capacity function classes (i.e., the "modern" interpolating regime), separated by the interpolation threshold.

[1] The appendix has a description of this.

[2] In literature, $\{y_i\}_{i=1}^n$ are called labels.

Previously, there had been no firm mathematical foundation for why this disparity exists, but in 2021, Sellke *et al* proposed a necessary condition for fitting that leads to this phenomenon. They proved that constructing a smooth function that fits d–dimensional data requires nd parameters. That is, overparametrization by a factor of d is a necessary condition for smooth interpolation.[3] This discovery, further pointed towards the necessity of large model sizes in deep learning **and will be the main focus of this exposition** ([2]).

This paper is divided into three main parts:

1. (**Universal Law of Robustness**) : We will give an intuitive *rendition* of the main ideas of the paper, introducing new definitions/concepts required for complete understanding.
2. (**Open Questions**): Is it possible to consider other non-euclidean norms? Is having a small *global* Lipschitz constant the right way to think about robustness?
3. (**Recent Advances**): We will give a short description of a recent modification of the Universal Law of Robustness and its application to finding matching subnetworks ([3]).

For the rest of this paper we make extensive use of footnotes to provide additional context and details without interrupting the flow of the main text. While not essential, these footnotes offer valuable insights and clarifications that enhance overall understanding. Therefore, we **strongly** encourage reading them.

1 The Universal Law of Robustness

1.1 Building Intuition

Firstly, we define *memorization*:

$$\text{Given data points } (x_i, y_i)_{i=1}^n, \text{ is there a function } h \in \mathcal{H} \text{ s.t. } \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i) - y_i) < \epsilon \tag{1}$$

Said differently, given a family of functions $\mathcal{H}$, for which data sets can we fit the data?[4]

We often denote $\mathcal{H}$, with a superscript p representing the number of parameters required in order to express a function in the family. Consequently, we can view $\mathcal{H}^p$ as a family of families of functions. In the case of neural networks (NN), p can represent the depth or width of a network. A natural question is then: how large should p be so that there exists Lipschitz $h \in \mathcal{H}^p$ such that (1) holds?[5]

We can assume the function class $\mathcal{H}^p$[6]consist of only families of Lipschitz continuous functions and then ask: are there "nice" families of Lipschitz functions such that we can always fit the data with one of them? The aim will be to establish that for family of Lipschitz functions its hard to fit the data with only n "degrees of freedom."[7]

[3]The Universal Law of Robustness does **not** talk about generalization, but the trade–off between size and smoothness on low training error. It posits, that scale enables entirely new behavior. As a **remark**, not all measures of smoothness admit a trade–off (e.g., Sobolov like norms).

[4]For simplicity, from henceforth, we assume a squared loss function.

[5]Our definition of robustness is an h that has a uniformly bounded Lipschitz constant. **This is important**, the Universal Law of Robustness requires the Lipschitz constant to be bounded everywhere.

[6]We proceed to drop the superscript in future references to the function class.

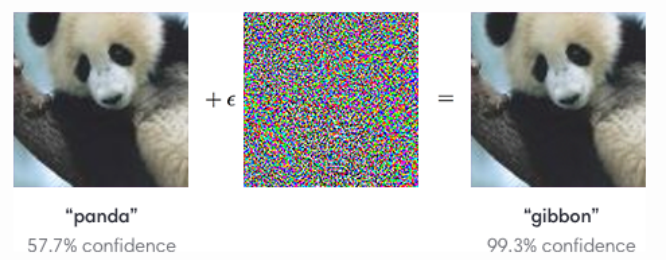
[7]Its "natural" to expect that if $p > \Omega(n)$ then we can fit the data, but if p is only as big as n you may not be able to memorize with a lipschitz function.

1.1.1 Digression: What does a good Lipschitz constant have to do with robustness?

Definition 1. *Let (X, d_X) and (Y, d_Y) be metric spaces. A function $h : X \to Y$ is called L–Lipschitz if there exists a $L \in \mathbb{R}$ such that that,*

$$d_Y(h(x), h(y)) \leq L d_X(x, y) \quad \forall x, y \in X.$$

In this exposition, we use on euclidean norm as our distance metric. A Lipschitz constant tracks *smoothness* since it is a measure of how much the output of a function can change relative to a change in its input. This has a close tie to robustness. Take the figure below:



With a small amount of carefully crafted noise to the image, the network can be tricked into thinking that the image of a panda is of a gibbon. That is, the network's output function h gives the result $h(x)$="panda" for the original image x, but $h(x + \epsilon\hat{x})$="gibbon" for the slightly noisy image $x + \epsilon\hat{x}$. This indicates that the output of the function h changes drastically near the input x. One way to prevent such sensitivity to small perturbations is to require that the function h has a small Lipschitz constant. By limiting the Lipschitz constant, we ensure that the function h is smooth and cannot exhibit large fluctuations near any input point, thus making the neural network more *robust*.

1.1.2 Back to the Main Argument

As previously stated, we hope to establish that for families of Lipschitz functions it is "hard" to fit the data using only n degrees of freedom. Before presenting the main theorem, let us given an example, where we can easily solve for p.

- **2–Layer NN**: Fix $k \in \mathbb{N}$, (activation function) $\sigma : \mathbb{R} \to \mathbb{R}$, $h : \mathbb{R}^d \to \mathbb{R}$. Let us consider functions of the form,

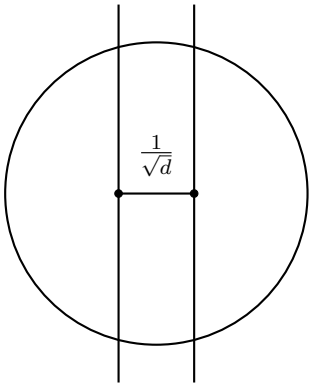
  $$h(x) = \sum_{i=1}^{k} \alpha_i \sigma(x \cdot w_i + b_i)$$

  Here, each component of the sum is a neuron (so we have k neurons), and $\alpha_i, b_i \in \mathbb{R}$. Then $p = k(d + 2)$. Each component requires $d + 2$ (d from w_i and one for each α_i, b_i) and there are k of them.

Now, let us give two extreme examples to illustrate the connection between the parameter count p and the Lipschitz constant.

- $p \approx n = d$: Suppose $x_i \sim U(S^{d-1})$ and $y_i \sim U(\{\pm 1\})$ independent from x_i. Then the Lipschitz constant is $\Omega(\sqrt{d})$

  Proof. Take the figure below and assume the left vertical line represents the subspace of all x_i's s.t. $y_i = 1$, while the right vertical line represents the subspace of all x_i's s.t. $y_i = -1$



Since $||x_i|| = 1$ and there are d of them, the center of mass of the x_i's have norm $\frac{1}{\sqrt{d}}$:

Proof.

$$\begin{aligned}\langle \frac{1}{d}\sum_{i=1}^{d} x_i, \frac{1}{d}\sum_{i=1}^{d} x_i\rangle &= \frac{1}{d^2}\langle \sum_{i=1}^{d} x_i, \sum_{i=1}^{d} x_i\rangle \\ &= \frac{1}{d^2}\sum_{i=1}^{d}\sum_{j=1}^{d}\langle x_i, x_j\rangle \\ &\overset{1}{=} \frac{1}{d^2}\cdot d = \frac{1}{d}\end{aligned}$$

1 $\sum_{i=1}^{d}\sum_{j=1}^{d}\langle x_i, x_j\rangle = d$ because if $i = j$ then we have 1, otherwise 0 because the correlations are 0. This is because over the entire sphere for every x_i there exists an equally likely x_j pointing in the opposite direction making the average contribution to the pairwise inner product 0 as the number of vectors increases.

Now, $\frac{1}{d}$ was the norm squared, hence we have $\frac{1}{\sqrt{d}}$ □

This tells us the mass is contained within a $\frac{1}{\sqrt{d}}$ width. Now, we must find a constant L s.t, $|h(x_i) - h(x_j)| \leq L|x_i - x_j|$, but this is simply

$$\begin{aligned}|1 - (-1)| &\leq L \cdot \frac{1}{\sqrt{d}} \\ 2 &\leq L \cdot \frac{1}{\sqrt{d}} \\ 2\sqrt{d} &\leq L\end{aligned}$$

Hence $L = \Omega(\sqrt{d})$ □

As a remark, $\sqrt{d}$ is a bad Lipschitz constant as it is highly sensitive. If one moves by distance $\frac{1}{\sqrt{d}}$, then the function value changes by 1. In literature, what is presented above is called the **Baum construction** and one can show it is actually $\Omega(d)$ Lipschitz.[8]

[8]Intuitively, if you move each of the d neurons (dimensions) a distance of $\frac{1}{\sqrt{d}}$, then in total you've moved only distance one ($\sqrt{d(\frac{1}{\sqrt{d^2}})}$) but you've activated d neurons so the function value is d.

- $p = nd$: Under the same assumptions, $x_i \sim U(S^{d-1})$ and $y_i \sim U(\{\pm 1\})$ independent from x_i, we have a Lipschitz constant of $O(1)$.

  Proof. The intuition here is that in high dimensions, the points will be well isolated with high probability. That is, if one draws a ball of radius $\frac{1}{r_i}$ around a point x_i, this ball will be empty. Consequently, interpolating these data points smoothly requires superposing r_i-Lip bump functions[9] around each data point. Hence, with $n(d+1)$ parameters (d for the center and 1 for the height for each bump function) one can get perfect fit with a $O(1)$ Lipschitz constant. □

In 2020, it was conjectured one needs $\text{Lip}(h) \geq \Omega(\sqrt{\frac{nd}{p}})$ for memorization in 2-layer NN ([4]). That is, there exists an interpolation between the two aforementioned cases.[10]

1.2 Framing the Theorem

Definition 2. *A measure μ on $\mathbb{R}^d$ is c–concentrated if for all φ 1–Lip,*

$$\mu(\{|\varphi(x) - \mathbb{E}_\mu \varphi| > t\}) \leq 2e^{-\frac{t^2}{c^2}}$$

In particular, c is the going to be the variance of 1–Lip functions, all while retaining a subgaussian tail. An important example, $\mu = U(S^{d-1})$ is $\frac{1}{\sqrt{d}}$–concentrated.

Definition 3. *$\mathcal{H} = \{h_\omega : \omega \in \Omega \subseteq \mathbb{R}^p\}$ is a J–Lip representation if for all $x \in B$,*

$$|h_{\omega_1}(x) - h_{\omega_2}(x)| \leq J|\omega_1 - \omega_2|$$

Here, B represents a bounded set.

Theorem 1 ([2])**.** *Let $(x_i, y_i)_{i=1}^n$ iid sampled from some measure $\mathcal{D}$ on $\mathbb{R}^d \times [-1, 1]$ s.t. the marginal of the measure on the x coordinate, i.e., $\mathcal{D}_x$ is c–concentrated. Moreover,*

- *Let $\mathcal{H}$ admit a J–Lip representation with p degrees of freedom.*
- *Assume for all x, $Var[y_i|X_i = X] \geq \sigma^2$*

[11] *Then, w.h.p for every $h \in \mathcal{H}$ s.t. $\frac{1}{n}\sum(h(x_i) - y_i)^2 < \sigma^2 - \epsilon \implies Lip(h) \geq \tilde{\Omega}(\frac{\epsilon}{\sigma}\sqrt{\frac{nd}{p}})$*

A few remarks on the theorem:

1. In real world data, labels (the y's) are not always noisy, but, mathematically, we have to assume labels are not deterministic otherwise one would be able to fit anything, and there would be no need to learn. Consequently, this "theory noise level" we inject is meant to capture the difficult part of learning. Proving a universal law of robustness on noiseless data is an open problem.

[9] A bump function is a technical term for a function that goes smoothly to 0 in finite space and is C^∞.

[10] This BLN conjecture is still open, as the Universal of Law of Robustness paper proves a special case of it, when one has polynomially bounded weights. However, the polynomial weight assumption is necessary when considering neural networks with ≥ 2 hidden layers. Consequently, the BLN conjecture, is of more mathematical interest than practical one.

[11] To further tie this theorem back to our original musings about how large p has to be: this tells us if p is very low, you would need a higher Lipschitz constant, which is less smooth and thus not as robust. Consequently, if you want to find an h that fits the data well and has a good notion of robustness you need a higher p. That is, any h with training error below the noise level σ^2 has smoothness $1/\text{Lip}(h)$ increasing with the number of parameters, p.

2. The theorem, and this will be made more clear in the proof, makes a very strong assumption that x_i are distributed from something that is highly concentrated, such as $U(S^{d-1})$ and bounded. What happens if $x_i \sim \mathcal{N}(0,1)$?

Before proceeding to the proof of Theorem 1, lets first build intuition as to why it might be true with a toy problem: Assume the x_i's and y_i's are independent. Now, suppose y_i is pure noise, that is, $y_i \sim \text{Ber}(\frac{1}{2})$ on ± 1. Take $A = \{|\frac{1}{n}\sum_i y_i| < 0.1\}$, then $\mathbb{P}(A) \geq 1 - 2e^{-\Omega(n)}$ by Hoeffdings inequality. Now, under A, fix a 1–Lip $h \in \mathcal{H}$. We can pose the question: what is the probability h exactly fits our data?[12]

Proof. We know $\mathcal{D}_x$ is $\frac{c}{\sqrt{d}}$–concentrated and h is 1–Lip. Now, h is $\frac{1}{\sqrt{d}}$ concentrated so h cannot be close to both -1 and 1 on large sets (since most points of h are close to the expectation). Either the set where h is close to -1 is small or the set where h is close to 1 is small. That is, it is one of

$$\mathcal{D}_x(\{h(x) \geq 1 - \epsilon\}) \leq e^{-\Omega(d)} \tag{2}$$

$$\mathcal{D}_x(\{h(x) \leq -1 + \epsilon\}) \leq e^{-\Omega(d)} \tag{3}$$

Now, conditioning on A (so about half of the points are -1 and the other 1), then for all h the probability that all the points fall under the proper label, that is, $\mathbb{P}(h \text{ fits the data}) \leq e^{-nd}$.[13] Consequently, taking a union bound over function class $\mathcal{H}$ of size N of 1–Lip funtions, tells us that the probability there exists an $h \in \mathcal{H}$ fitting the data is at most: $Ne^{-nd} = e^{\log(N)-nd}$. Hence, under a discretized argument, "for a smoothly parametrized family with p (bounded) parameters one expects $log(N) = \tilde{O}(p)$ ([2])." □

1.3 The Proof

Before procreeding to the proof, we introduce a few definitions and lemmas:

Definition 4. *The log moment generation function φ of a random variable X is*

$$\varphi(\lambda) := log\ \mathbb{E}[e^{\lambda(X - \mathbb{E}(X)}]$$

Definition 5. *A r.v. X is σ–subgaussian if its log moment generation function $\varphi(\lambda) \leq \frac{\lambda^2\sigma^2}{2}$*

Lemma 1. *If X is σ–subgaussian then $\mathbb{P}(|X - \mu| \geq t) \leq 2e^{-t^2/2\sigma^2}$*

Lemma 2. *if $\{W_i\}_{i=1}^n$ iid c–subgaussian then $\frac{1}{n}\sum W_i$ is $\frac{c}{\sqrt{n}}$–subgaussian.*

Proof. WLOG, let $\mathbb{E}[W_1] = 0$, then

$$\begin{aligned}
\mathbb{E}[e^{\lambda/n(W_1+W_2+\cdots+W_n)}] &= \mathbb{E}[e^{\lambda/nW_i}] \cdots \mathbb{E}[e^{\lambda/nW_n}] \\
&\leq e^{(\frac{\lambda^2}{n^2}c^2)/2} \cdots e^{(\frac{\lambda^2}{n^2}c^2)/2} \\
&= e^{(\lambda^2\frac{c^2}{n})/2}
\end{aligned}$$

Consequently, $\frac{1}{n}\sum W_i$ is $\frac{c}{\sqrt{n}}$–subgaussian □

[12] We condition under A to do away with the possibility of y_i's not being well-balanced. As a remark, the probability of this unlikely event is not amplified by a union bound over $h \in \mathcal{H}$.

[13] More precisely, it is $e^{\frac{-nd}{\text{Lip}(h)^2}}$ if we take $\mathcal{D}_x$ to be $\frac{c}{\sqrt{d}}$–**isoperimetry** because then h can be any bounded L–Lipschitz function.

Now, we proceed with the proof of Theorem 1:

Proof. Let $y_i = g(x_i) + z_i$, such that $\mathbb{E}[z_i|x_i] = 0$. Here $g(x_i)$ is the signal and is deterministic while z_i is pure noise. Recall, $Var(z_i) \geq \sigma^2$. Consider the following two events[14]:

$$A := \{\frac{1}{n}\sum z_i^2 \geq \sigma^2 - \frac{\epsilon}{10}\} \tag{4}$$

$$B := \{\frac{1}{n}\sum z_i g(x_i) \geq -\frac{\epsilon}{10}\} \tag{5}$$

Then, by Hoeffdings, $\mathbb{P}(A \cap B) \geq 1 - e^{\Omega(n\epsilon^2)}$. Now, for all $h \in \mathcal{H}$, define

$$C_h := \{\frac{1}{n}\sum h(x_i) z_i \leq \frac{\epsilon}{10}\} \tag{6}$$

We claim, if event $A \cap B \cap C_h$ occurs then it must be the case that we are not too correlated, i.e., $\frac{1}{n}\sum (h(x_i) - y_i)^2 \geq \sigma^2 - \epsilon$.

Proof. Let $G = \frac{1}{\sqrt{n}}(g(x_1), ..., g(x_n)), Z = \frac{1}{\sqrt{n}}(z_1, ..., z_n)$, and $H = \frac{1}{\sqrt{n}}(h(x_1), ..., h(x_n))$. Then, we want to lower bound $||H - G - Z||^2$. Notice under $A \cap B \cap C_h$,

$$||H - G - Z||^2 = |Z|^2 + 2\langle Z, G\rangle - 2\langle Z, H\rangle + |H - G|^2 \geq \sigma^2 - \epsilon$$

□

If we can establish that for all 1–Lip h, $\mathbb{P}(C_h) \geq 1 - e^{-\Omega(nd)}$ then Theorem 1 follows. This is because, under event $A \cap B \cap C_h$ we are not too correlated and thus cannot approximate. Consequently, we will be able to establish w.h.p. there does not exists a 1–Lip h that approximates (fits) our data well, i.e., where $\frac{1}{n}\sum (h(x_i) - y_i)^2 < \sigma^2 - \epsilon$.

Proof. Let $r_i = z_i(h(x_i) - \mathbb{E}_{\mathcal{D}_x}[h])$ and consider $\frac{1}{n}\sum r_i$. We know x_i are distributed in a measure that is $\frac{1}{\sqrt{d}}$ concentrated, that is, the probability of x_i being far from the mean decreases exponentially with the distance, scaled by $1/\sqrt{d}$. Now, since h is 1–Lip the deviation of $h(x_i)$ from its mean is bounded by the deviation of x_i from its mean. So, the probability of $|h(x_i) - \mathbb{E}[h]|$ being large also decreases exponentially, scaled by $1/\sqrt{d}$. Consequently, $h(x_i) - \mathbb{E}_{\mathcal{D}_x}[h]$ is $\frac{1}{\sqrt{d}}$–subgaussian. Since the $z_i \in [-1, 1]$, r_i is also $\frac{1}{\sqrt{d}}$–subgaussian.

By Lemma 2, $\frac{1}{n}\sum r_i$ is $\frac{1}{\sqrt{nd}}$–subgaussian. By Lemma 1, this tells us that the probability $\frac{1}{n}\sum r_i$ is at least ϵ is $\approx e^{-\epsilon^2 nd}$. Moreover, $|\sum z_i \mathbb{E}[h]| \leq |\sum z_i|$. Therefore, if we have[15]:

$$D := \{|\frac{1}{n}\sum z_i \mathbb{E}[h]| < \frac{\epsilon}{10}\} \tag{7}$$

then $\mathbb{P}(D) \geq 1 - 2e^{-\Omega(n\epsilon^2)}$ by Hoeffdings.[16] Consequently, $\mathbb{P}(C_h) \geq 1 - e^{-\Omega(nd)}$ under D.[17] □

□

[14]Notice, neither depends on h.
[15]The $\mathbb{E}[h]$ is bounded because we can always round it to ± 1 since $y_i \in [-1, 1]$.
[16]Notice, this bound is independent of h.
[17]Or, $\mathbb{P}(C_h^c - D) \leq e^{-\Omega(nd)}$.

1.4 Closing Remarks

We are not quite done. We have shown, one cannot finding a 1–Lip function h to fit the data. In fact, this is true for any L–lip function, if we use the concept of c–isoperimetry, rather than c–concentrated,

Definition 6. *A probability measure μ on $\mathbb{R}^d$ satisfies c-isoperimetry if for any bounded L–Lipschitz φ,*

$$\mu(\{|\varphi(x) - \mathbb{E}_\mu \varphi| > t\}) \leq 2e^{-\frac{dt^2}{2cL^2}}$$

Now, this does not directly imply any h in our class that fits the training data well below the noise level must also have $Lip(h) \geq \tilde{\Omega}(\frac{\epsilon}{\sigma}\sqrt{\frac{nd}{p}})$.

In the paper, a proof of Theorem 3 is provided using the ϵ–net technique (to deal with an infinite $\mathcal{H}$), then is applied to deep neural nets to get our desired Lipschitz constant bound. Proving Theorem 3 requires a lot of runway; the paper has many previous theorems/lemma leading up to it and is not particularly enlightening. The heart of the paper, which is essentially shown above, is proving under certain conditions, there is no L–lip h that approximates data well.

2 Open Questions [18]

2.0.1 Norms

As mentioned in a previous footnote, not all measures of smoothness admit a trade–off; examples include natural alternatives such as the Sobolov norms or Barron norms on two layer networks ([5]). That is, for those measures of smoothness, it is possible to build networks with as few parameters as information-theoretically possible while also being as smooth as possible (with respect to that norm). Let's expand on why we cannot find a universal law of robustness (in the ways it's constructed in the paper) using Sobolov norms:

- The Universal Law of Robustness requires the Lipschitz constant to be bounded everywhere. We can view the Lipschitz constant as an upper bound on the magnitude of the gradient, so this boundedness constraint is akin to needing a small gradient everywhere. Consequently, If we require the ℓ_2 norm of the gradient to be small, it means that the gradient will be small at every point. This is in contrast to Sobolov norms that only lets us express that a function has a small gradient *almost* everywhere.[19]

The paper focuses on Euclidean norm, specifically ℓ_2, but the proof does not depend on this norm. Everything follows as long as one assumes isoperimetry in the norm of interests. Now, finding a norm in which a trade–off exists and finding an interesting case in which isoperimetry provably holds for that norm is no easy task.[20] An interesting open problem is to check if a universal law of robustness holds for something like the wasserstein norm or ℓ_∞. The latter norm is of particular interest as that is more commonly used in adversarial robustness literature.

[18] My initial plan was to poke at the role of depth in the Universal Law of Robustness. That is, given depth D neural networks, showing why finding a theoretic construction that shows that $\text{Lip}(h) \geq \tilde{\Omega}\left(\sqrt{\frac{nd}{Dp}}\right)$ is tight (at some large depth D) is a non-trivial task. Unfortunately, to do this well would require introducing too many new ideas, and would make this exposition long.

[19] Sobolov norm like $\mathbb{E}^\mu |\nabla h(x)|^2$.

[20] In what cases is it reasonable to think of this isoperimetry property defined in terms of concentrations of Lipschitz functions as realistic?

2.0.2 Robustness

The paper, [2], connects a low number of parameters, p, to a higher Lipschitz constant bound, and then connects that to low robustness (or high adversarial generalization error).[21] However, this latter connection is tenuous especially since we are looking at a global Lipschitz constant.

From a theoretical perspective, it may not be the case that this global constant is fundamentally related to the adversarial generalization error. That relationship may depend on more fine–grained quantities such as the *distribution* of the Lipschitz constant over the data domain. Studying more fine–grained properties is an open problem, but requires finding other norms if we still want some form of a universal law.

From a practical perspective, there are two points of contention,

- One of the most effective adversarially robust algorithms is randomized smoothing, which does not rely on a Lipschitz continuity assumption ([6]).
- Provably safe defenses aim to provide formal guarantees about a model's robustness to adversarial perturbations within a certain ϵ–ball around an input and rely on the concept of Lipschitz continuity to establish bounds on the model's output variation. However, instead of using a global Lipschitz constant, which may be overly conservative and limit the model's expressiveness, many verifiers use the local Lipschitz constant.[22] Even if a model has a high global Lipschitz constant, it can still have a low local Lipschitz constant in regions close to the actual data points. This means that the model can be relatively smooth and stable around the data manifold. State–of–the–art robust models recognize this distinction and often explicitly optimize for local Lipschitz constants rather than global smoothness ([7]).

In general, connecting the papers findings more precisely to robust test error is a big open problem.

3 Recent Advances

There have been a couple papers pushing/expanding the theoretical foundations of this universal law of robustness ([7], [3]). In this section, we give a short description of [3], and the way in which they connect the Universal Law of Robustness to pruning.

3.1 Background

There has been increasing interest in neural network pruning as a means to reduce the cost and size of training while maintaining performance. Pruning is done by *masking away* a certain fraction of the weights (setting them to zero), so they can be ignored during training or inference which reduces the number of operations and thus the cost required to achieve good performance. One can prune at any moment during the life-cycle (at initialization, during training or after training).

It was empirically shown that there exist "lottery tickets": sparse subnetworks that can be trained to the accuracy of the full dense model at or near initialization ([8]).[23] Moreover, there exists algo-

[21] Adversarial generalization error measures how well the model performs on a test set that has been adversarially perturbed. Consequently, with low robustness comes high adversarial generalization error.

[22] This allows for a more precise characterization of the model's behavior around specific data points.

[23] [8] sought to answer the question: If we can prune models after training, can we train smaller models? The study of finding these smaller models that can be trained to the accuracy of the full model is called matching subnetworks.

rithms to find these matching subnetworks called *iterative magnitude pruning* (IMP) with weight rewinding. While interesting in principle, the IMP technique necessitates training the entire model on the dataset multiple times to discover these "lottery tickets." This approach contradicts the primary objective of finding highly sparse yet trainable subnetworks.

With this contention, a recent line of of research has emerged, charged with finding these "lottery tickets" fast (i.e., without training the full model). While there remains large efforts to develop algorithms to prune at initialization, most of them remain unsuccessful at finding lottery tickets in general settings without training.

If a technique for finding matching subnetwork of a task were available, one could prune the network at the outset and then efficiently train the resulting sparse subnetwork to achieve comparable performance on the task, potentially rendering the training of large, dense models unnecessary. Finding an algorithm capable of this is unlikely as it would contradict the large theoretical literature that espouses the necessity of overparametrization ([1],[2]). Consequently, any theory that intends to align with past literature on the benefits of overparametrization and also formalize the intractability of pruning at initialization must explain why lottery tickets can exist, but not be found efficiently (without training the full network on the data). This paper seeks to accomplish exactly that.

3.2 Contributions

They present a modified version of the Universal Law or Robustness that replaces parameter count, p, with an effective parameter count, p_{eff},

Theorem 2 ([3])**.** *Assume the same conditions as in Theorem 3 and that $\mathcal{H}$ has the additional structure of masks, so that each hypothesis $h \in \mathcal{H}$ has parameters $(\mathbf{m}, \mathbf{w})$ satisfying $\mathbf{m}_i = 0 \implies \mathbf{w}_i = 0$ for all $i \in [p]$. Then, with high probability over sampling of the data, one has for any learning algorithm W taking in data $\mathcal{D}$ and outputting function $h^W \in \mathcal{H}$:*

$$\frac{1}{n}\sum_{i=1}^{n}(h^W(x_i) - y_i)^2 \leq \sigma^2 - \epsilon \implies \text{Lip}(h) \geq \widetilde{\Omega}\left(\epsilon\sqrt{\frac{nd}{p_{\mathit{eff}}}}\right),$$

where $p_{\mathit{eff}} = \tilde{\Theta}\left(I(\mathbf{m}^W; \mathcal{D}) + \mathbb{E}[\|\mathbf{m}\|_1]\right)$.

This new parameter couples the number of parameters with the mutual information of the sparsity mask with the dataset.[24] More specifically, it includes the number of unmasked parameters $\mathbb{E}[\|\mathbf{m}\|_1]$ as well as the mutual information between the sparsity pattern $\mathbf{m}^W$ and the data.[25] This coupling reveals a new way in which information and parameters can be traded off.[26]

With this new re–framing they show that subnetworks derived from pruning algorithms that train on the data, such as lottery tickets, **are not really sparse in $p_{\mathbf{eff}}$**, whereas those derived from pruning at initialization are. Essentially, a learning algorithm pruning at initialization with little

[24] p_{eff} cannot be larger than p up to logarithmic factors incurred from discretization.

[25] The latter term reflects the notion that a sparsity mask learned from data should be interpreted as a set of binary parameters, thus contributing to the network's total parameter count.

[26] Similar to [2], they show that for a learned function to fit below the noise level, it must correlate with the noise in the data, but they go beyond this by showing this correlation must be large, and thus the mutual information must be large with high probability.

dependence on data will result in a subnetwork that has low p_{eff} and thus poor robustness, since it is truly sparse. On the other hand, pruning algorithms that iteratively use properties of the data to find a mask may not be truly sparse in p_{eff}; they trade off the unmasked parameter count of the network for mutual information, so that the subnetwork produced has p_{eff} much larger than a truly sparse network. The paper conjectures this disparity is exactly why lottery tickets exists, but cannot be found fast.

4 Closing Remarks

The Universal Law of Robustness fully contextualizes the trade–off between size and smoothness on low training error. It tells us, any $h \in \mathcal{H}$ with training error below the noise level σ^2 has smoothness $1/\text{Lip}(h)$ increasing with the number of parameters, p. Morever, this was under a broad parametrized function class, $\mathcal{H}$, so this law holds for practical function classes (e.g., resnet or transformers). The work remains groundbreakng despite the many open problems, including but not limited to:

- Can we construct a universal law of robustness for noiseless data?
- Can we find a universal law of robustness under different norms (e.g., ℓ_∞)?
- Can we better connect the Universal Law of Robustness to robust test error?

Furthermore, using the Universal Law of Robustness as a foundation, a new paper emerged giving a solution the question of why "lottery tickets" exist but cannot be found efficiently ([3]); another groundbreaking result, that may have otherwise never been found. We are excited to see how this new line of research continues evolving.

References

[1] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine learning practice and the bias-variance trade-off," Proceedings of the National Academy of Sciences, vol. 116, pp. 15849–15854, Aug. 2019. arXiv:1812.11118 [cs, stat].

[2] S. Bubeck and M. Sellke, "A Universal Law of Robustness via Isoperimetry," Dec. 2022. arXiv:2105.12806 [cs, stat].

[3] T. Kumar, K. Luo, and M. Sellke, "No Free Prune: Information-Theoretic Barriers to Pruning at Initialization," Feb. 2024. arXiv:2402.01089 [cs, stat].

[4] S. Bubeck, Y. Li, and D. Nagaraj, "A law of robustness for two-layers neural networks," Nov. 2020. arXiv:2009.14444 [cs, stat].

[5] S. Bubeck, R. Eldan, Y. T. Lee, and D. Mikulincer, "Network size and weights size for memorization with two-layers neural networks," Nov. 2020. arXiv:2006.02855 [cs, stat].

[6] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter, "Certified Adversarial Robustness via Randomized Smoothing," June 2019. arXiv:1902.02918 [cs, stat].

[7] C. Qin, J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth, and P. Kohli, "Adversarial Robustness through Local Linearization,"

[8] J. Frankle and M. Carbin, "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks," Mar. 2019. arXiv:1803.03635 [cs].

Appendices

A Bias–Variance Trade–off

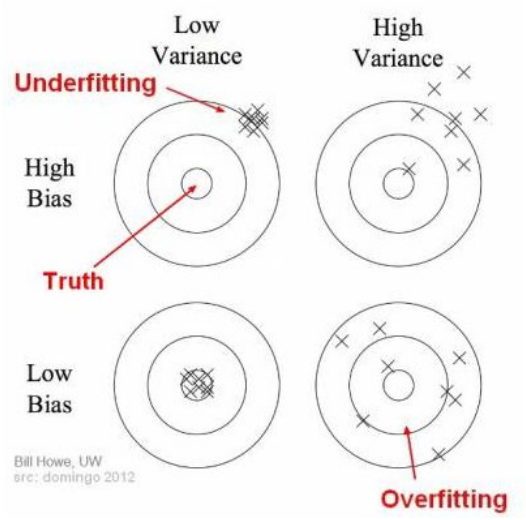
Assume there is an value θ we seek to estimate. Then,

$$\text{Bias}(\hat{\theta}, \theta) = \mathbb{E}[\hat{\theta}] - \theta$$

Here, the expectation is with respect to the underlying distribution that $\hat{\theta}$ is from. If $\text{Bias}(\hat{\theta}, \theta) = 0$, it is called unbiased. Now variance is,

$$\text{Var}(\hat{\theta}) = \mathbb{E}[\theta - \mathbb{E}[\hat{\theta}]]^2$$

The goal is to have an estimator with low bias and variance. The image below is helpful,



If our model is simple with few parameters then it may have high bias and low variance (i.e., under–fits but generalizes well). On the other hand, having too large of a number of parameters can lead to high variance and low bias (over–fits but struggles to generalize). The goal then becomes to find a balance.

More specifically: let's say we have data set $S = (x_i, y_i)_{i=1}^n$ where each sample (x_i, y_i) is drawn iid from some underlying distribution D. We want to use S to learn a function $h_S : \mathcal{X} \to \mathcal{Y}$ (here $\mathcal{Y} \subset \mathbb{R}$). We want to know the expected prediction error of this learning algorithm. That is, we want to know $\mathbb{E}_{x,y,S}[(y - h_S(x))^2]$.

Let $\bar{y} = \bar{h}(x) = \mathbb{E}_S[h_S(x)]$. Then for a fixed x,

$$\begin{aligned}
\mathbb{E}_{y,S}[(y - h_S(x))^2] &= \mathbb{E}_{y,S}[(y - \bar{y} + \bar{y} - h_S(x))^2] \\
&= \mathbb{E}_y[(y - \bar{y})^2] + \mathbb{E}_S[(\bar{y} - h_S(x))^2] + 2\mathbb{E}_{y,S}[(y - \bar{y})(\bar{y} - h_S(x))] \\
&= \mathbb{E}_y[(y - \bar{y})^2] + \mathbb{E}_S[(\bar{y} - h_S(x))^2] \\
&= \mathbb{E}_y[(y - \bar{y})^2] + \mathbb{E}_S[(\bar{h}(x) - h_S(x))^2 + (\bar{y} - \bar{h}_S(x))^2]
\end{aligned}$$

Consequently, for fixed x

$$\mathbb{E}_{y,S}[(y - h_S(x))^2] = \text{Var}_{y|x}(y) + \text{Var}_S(h_S(x)) + \text{bias}(h_S(x))^2$$

$\implies$

$$\mathbb{E}_{x,y,S}[(y - h_S(x))^2] = \mathbb{E}_x[\mathrm{Var}_{y|x}(y) + \mathrm{Var}_S(h_S(x)) + \mathrm{bias}(h_S(x))^2]$$

This is the bias–variance trade–off (a trade-off between accuracy and complexity): we want to choose the right model which requires balancing between reducing the second and third terms in order to get the lowest mean squared error.

B Main Theorem From [2]

Theorem 3. *[2] Let $\mathcal{H}$ be a class of functions from $\mathbb{R}^d \to \mathbb{R}$ and let $(x_i, y_i)_{i\in[n]}$ be i.i.d. input-output pairs in $\mathbb{R}^d \times [-1, 1]$. Fix $\epsilon, \delta \in (0, 1)$. Assume that:*

1. *The function class can be written as $\mathcal{H} = \{h_{\boldsymbol{w}}, \boldsymbol{w} \in \mathcal{W}\}$ with $\mathcal{W} \subset \mathbb{R}^p$, $\mathrm{diam}(\mathcal{W}) \leq W$ and for any $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathcal{W}$,*
$$||h_{\boldsymbol{w}_1} - h_{\boldsymbol{w}_2}||_\infty \leq J||\boldsymbol{w}_1 - \boldsymbol{w}_2||.$$
2. *The distribution μ of the covariates x_i can be written as $\mu = \sum_{\ell=1}^k \alpha_\ell \mu_\ell$, where each μ_ℓ satisfies c-isoperimetry, $\alpha_\ell \geq 0, \sum_{\ell=1}^k \alpha_\ell = 1$, and k is such that*
$$10^4 k \log(8k/\delta) \leq n\epsilon^2. \tag{8}$$
3. *The expected conditional variance of the output is strictly positive, denoted $\sigma^2 \equiv \mathbb{E}^\mu[Var[y|x]] > 0$.*
4. *The dimension d is large compared to ϵ:*
$$d \geq C_1 \left(\frac{cL^2\sigma^2}{\epsilon^2}\right). \tag{9}$$

Then, with probability at least $1-\delta$ with respect to the sampling of the data, one has simultaneously for all $h \in \mathcal{H}$:

$$\frac{1}{n}\sum_{i=1}^n (h(x_i) - y_i)^2 \leq \sigma^2 - \epsilon \;\Rightarrow\; \mathrm{Lip}(h) \geq \frac{\epsilon}{\sigma\sqrt{C_2 c}} \times \sqrt{\frac{nd}{p\log(1 + 60WJ\epsilon^{-1}) + \log(4/\delta)}}. \tag{10}$$

Moreover if $\mathcal{W}$ consists only of s-sparse vectors with $||w||_0 \leq s$, then the above inequality improves to

$$\frac{1}{n}\sum_{i=1}^n (h(x_i) - y_i)^2 \leq \sigma^2 - \epsilon \;\Rightarrow\; \mathrm{Lip}(h) \geq \frac{\epsilon}{\sigma\sqrt{C_2 c}} \sqrt{\frac{nd}{s\log\left(p(1 + 60WJ\epsilon^{-1})\right) + \log(4/\delta)}}. \tag{11}$$