S&DS 659: Mathematics of Deep Learning

Lecture 1: General Introduction

Instructor: Theodor Misiakiewicz

Yale University

Spring 2025

Today

0. Class organization
1. Why and What is Deep Learning theory?
2. Supervised learning, ERM, classical learning paradigm
3. New paradigm: two working hypotheses
4. Plan for the semester
5. Some (pre)history for the theory of NNs

0 Class Organization

Schedule

- Lectures:
  - Time: Wednesdays: 4:00pm – 5:50pm
  - Location: Kline Tower 207 (2nd floor)
- Office Hours:
  - Time: Thursdays: 4:00pm – 5:00pm
  - Location: Kline Tower 1049 (10th floor)
- 12 lectures + 1 in-class presentations (last week of class)

Goal

- **Course description:**
  The goal of this course is to provide an introduction to selected topics in deep learning theory. I will present a number of mathematical models and theoretical concepts that have emerged in recent years to understand neural networks.

- **Disclaimers:**
  - Currently no general theory for how neural networks work, or even a consensus of what is the right approach to study them.
  - Selection of topics + presentation will reflect my interests and biases.
  - First iteration of this class.

- Nonetheless, I will aim for a coherent and systematic presentation.

Prerequisites

- I will not assume specific background in machine learning, let alone neural networks.
- On the other hand, I will assume a degree of mathematical maturity: linear algebra, analysis, and probability theory (at the level of S&DS 241/541).

References

- I will maintain a separate PDF which I will update regularly with references on topics we see in class.

- There are many tutorials on Deep Learning theory. You can check:
  - *Deep learning: A statistical viewpoint*, Bartlett, Montanari, Rakhlin, 2021.
  - *Six Lectures on Linearized Neural Networks*, Misiakiewicz, Montanari, 2023.
  - *Applying statistical learning theory to deep learning*, Gerbelot, et al., 2024.
  - *Deep learning theory lecture notes*, Telgarsky, 2022.
  - *Learning Theory from First Principles*, Bach, 2023.

Assignments

- You will have to scribe one lecture (in latex) during the semester.
  - Link to google doc with class schedule: please add yourself before 01/21.
  - Scribed lecture due the following Monday by email, with both '.tex' and '.pdf' files.
- Written report about a topic related to Deep Learning theory:
  - Work in groups of 2 or 3.
  - Topic + group members by email (02/19), 1 page summary (2/26), 6 pages report in NeurIPS format (4/16)
- In-class presentation during the last lecture of the semester (4/23).
- Separate PDF with rules and ideas of topics in the next few weeks.

Any questions?

1 Why and What is Deep Learning Theory?

Why? (The Obvious)

Some twitter "facts":

- 200 billion USD invested in training next generation of AI in 2024.
  [Some VC]
- Trained on the whole internet: 13 trillion tokens for GPT4.
  [Jensen Huang]
- 100 million monthly users on ChatGPT.
  [Sam]
- Nuclear power plants to power data centers.
  [Google just ordered 2 reactors]
- AI will bring the apocalypse. We should bomb data centers.
  [Eliezer Yudkowsky]
- AI will bring utopia. Riemann hypothesis solved by o5.
  [Average OpenAI twitter fan]

Most recent advances have come from practice:

- Trial-and-error architecture and algorithm innovations (with exceptional intuition)
- Technological advances (most importantly, increase in compute power)

Limited understanding of how they work:

- As they are increasingly deployed throughout the world, we should understand how they operate and their limitations (for safety and sanity).
- Potentially make them more efficient and more broadly applicable.

Why? (The Less Obvious)

- Some challenges are unlikely to be solved by engineering alone:
  - Reliability, privacy, robustness, interpretability...

- Interaction between application and theory typically goes both directions:
  - Understanding deep learning requires new mathematical insights, which will feed back into other areas of mathematics, statistics, optimization, learning theory...
  - Offer new perspectives on statistical learning theory
  - Inspire a number of mathematical problems

  E.g., random matrix theory, optimization in random landscapes, kernel methods, decision trees, non-convex optimization, overfitting...

What is Deep Learning?

- Diverse toolbox of techniques/algorithms that have evolved from the decades-old methodology of neural networks: *"circuits of parametrized nonlinear functions trained by gradient-based methods."*

- **Diverse goals**
  - Function approximation, language model, image generation, PDE solver, playing games, etc.

- **Diverse architectures**
  - Fully connected, convnets, resnets, LSTM, transformers, etc.

- **Lots and lots of engineering...**
  - Architecture choices, optimization algorithms, regularization techniques, hyperparameter tuning, ...
  - Hardware, infrastructure, libraries, ...

For the purpose of this class (except last 2 lectures): **fully-connected neural nets**

$$f_{\mathsf{NN},L}(\boldsymbol{x};\boldsymbol{\Theta}) = f^{(L)} \circ f^{(L-1)} \circ \cdots \circ f^{(1)}(\boldsymbol{x}),$$

where:

- $\boldsymbol{x} \in \mathbb{R}^d$ is the input vector (e.g., image);
- $f^{(l)}(\boldsymbol{z}) = \sigma(\boldsymbol{W}^{(l)}\boldsymbol{z} + \boldsymbol{b}^{(l)})$, for $l = 1, \ldots, L-1$;
- $f^{(L)}(\boldsymbol{z}) = \boldsymbol{a}_L^\mathsf{T}\boldsymbol{z}$;
- $\boldsymbol{W}^{(l)} \in \mathbb{R}^{N_l \times N_{l-1}}$, $\boldsymbol{b}^{(l)} \in \mathbb{R}^{N_l}$, $\boldsymbol{a}_L \in \mathbb{R}^{N_L}$ are trainable parameters;
- $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function. E.g., $\mathrm{ReLU}(x) = \max(x, 0)$;
- $N_0 = d$ and N_ℓ are the width of the intermediary layers.

The network parameters are $\boldsymbol{\Theta} = \{\boldsymbol{W}^{(l)}, \boldsymbol{b}^{(l)}\}_{l=1}^{L-1} \cup \{\boldsymbol{a}_L\}$.

Goals of Theory?

Leo Breiman (1928–2005): "Reflections After Refereeing Papers for NIPS" (1995)

---

2. USES OF THEORY

- **Comfort**: We knew it worked, but it's nice to have a proof.
- **Insight**: Aha! So that's why it works.
- **Innovation**: At last, a mathematically proven idea that applies to data.
- **Suggestion**: Something like this might work with data.

- Andrea Montanari (Stanford), paraphrased:

  "Everyone is talking about a theory of Deep Learning, as if it obviously exists. It's like asking theoretical physicists to do a theory for washing machines. "

  "...we did not wait to have a proof of existence and uniqueness of Navier-Stokes equations before building airplanes..."

This Class

- Do we already know the "elecrodynamics" of deep learning?
- Several aspects of DL methodologies contradict standard wisdom from classical statistical learning theory. **Their success seems to follow from completely different mathematical principles than previous approaches.**
- This class:
  - **Focus on mathematical principles behind the deep learning revolution.**
  - Highly stylized models with emphasis on mathematical insights.
- Maybe next iteration of this class: LLMs + diffusion models

  (Boaz Barak: "By the end of the course you should be able to read most cutting-edge papers in this field.")

2 The Classical Paradigm of Learning

Supervised Learning

- For concreteness, most of this class: **supervised learning setting**.

- We get iid samples $(y_i, \boldsymbol{x}_i)_{i \le n} \sim_{iid} \mathbb{P}$
  - Covariate/input $\boldsymbol{x} \in \mathbb{R}^d$ (e.g., text, image, etc)
  - Response/label $y \in \mathbb{R}$.

- Fit a predictor/model $\hat{f} : \mathbb{R}^d \to \mathbb{R}$ such that given a new covariate $\boldsymbol{x}_{\text{new}}$, predicts the label y_{new} using $\hat{f}(\boldsymbol{x}_{\text{new}})$.

- Measure the performance of this predictor: loss $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$,

$$\text{Population loss/Risk/Test error: } \mathcal{R}(\hat{f}) = \mathbb{E}[\ell(y_{\text{new}}, \hat{f}(\boldsymbol{x}_{\text{new}}))].$$

E.g., squared loss $\ell(y, \hat{y}) = (y - \hat{y})^2$ ("regression loss")
cross-entropy loss $\ell(y, \hat{y}) = \log(1 + e^{-y\hat{y}})$ ("classification loss")

Empirical Risk Minimization

- **Goal:** Fit a predictor $\hat{f}$ with small $\mathcal{R}(\hat{f})$. In practice, minimize the test error

$$\mathcal{R}(f) = \mathbb{E}[\ell(y, f(\boldsymbol{x}))]$$

over a (parametrized) class of models $\mathcal{F} \subseteq \{f : \mathbb{R}^d \to \mathbb{R}\}$:

E.g., $\mathcal{F} = \{\boldsymbol{x} \mapsto \langle \boldsymbol{w}, \boldsymbol{x} \rangle \ : \ \boldsymbol{w} \in \mathbb{R}^d\}$, or $\mathcal{F} = \{\text{L-layer fully connected NNs}\}$

- Population dist $\mathbb{P}$ unknown.
Instead, natural approach: **Empirical Risk Minimization**

$$\hat{f}_{\text{ERM}} \in \arg\min_{f \in \mathcal{F}} \widehat{\mathcal{R}}_n(f) := \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\boldsymbol{x}_i)).$$

Rationale: we have $\widehat{\mathcal{R}}_n(f) \longrightarrow \mathbb{E}[\widehat{\mathcal{R}}_n(f)] = \mathcal{R}(f)$ and perhaps

$$\arg\min_{f \in \mathcal{F}} \widehat{\mathcal{R}}_n(f) \approx \arg\min_{f \in \mathcal{F}} \mathcal{R}(f).$$

Decomposition Excess Risk

- Algorithm compute approximate minimizer $\hat{f}$ of ERM problem.
- Denote $f_* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f)$.
- "Classical" decomposition of the excess test error:

$$
\begin{aligned}
& \mathcal{R}(\hat{f}) - \inf_f \mathcal{R}(f) \\
= \; & \inf_{f \in \mathcal{F}} \mathcal{R}(f) - \inf_f \mathcal{R}(f) && \Big\} \text{ "Approximation error"} \\
& + \left(\mathcal{R}(\hat{f}) - \widehat{\mathcal{R}}_n(\hat{f})\right) + \left(\widehat{\mathcal{R}}_n(f_*) - \mathcal{R}(f_*)\right) && \Big\} \text{ "Generalization error"} \\
& + \widehat{\mathcal{R}}_n(\hat{f}) - \widehat{\mathcal{R}}_n(\hat{f}_{\mathrm{ERM}}) && \Big\} \text{ "Optimization error"} \\
& + \widehat{\mathcal{R}}_n(\hat{f}_{\mathrm{ERM}}) - \widehat{\mathcal{R}}_n(f_*) && \Big\} \leq 0.
\end{aligned}
$$

Three Competing Objectives

- Statistical prediction must balance three goals:
  - **Approximation:** Class of models $\mathcal{F}$ needs to be expressive enough to approximate the task at hand.
  - **Generalization:** The predictor fitted on training data needs to generalize to new test data.
  - **Computation:** The training algorithm must be computationally efficient to be practical.

- **Often competing goals.** E.g.,
  - Richer class of models can lead to worse generalization due to overfitting.
  - Methods that make optimal use of limited data might be computationally intractable.

Classical approach

Classical way of managing these trade-offs:

(1) **Convexity:** optimization is tractable because it is convex.
Loss function and parametrization of $\mathcal{F}$ are chosen such that $\widehat{\mathcal{R}}_n(f)$ is convex.

e.g., ℓ convex and linearly parametrized $\mathcal{F} = \{\boldsymbol{x} \mapsto \langle \boldsymbol{\theta}, \boldsymbol{\psi}(\boldsymbol{x}) \rangle\}$.

(2) **Uniform Convergence:** predictors generalize to new data because of UC.

Uniform Convergence (I)

- Generalization error: $\widehat{\mathcal{R}}_n(f_*) - \mathcal{R}(f_*)$ and $\widehat{\mathcal{R}}_n(\hat{f}) - \mathcal{R}(\hat{f})$
  - **1st term:** by CLT,

$$\widehat{\mathcal{R}}_n(f_*) - \mathcal{R}(f_*) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f_*(\boldsymbol{x}_i)) - \mathbb{E}[\ell(y, f_*(\boldsymbol{x}))] = O(n^{-1/2}).$$

  - **2nd term:** $\hat{f}$ depends on the data, can't use CLT. Instead:

$$\left|\widehat{\mathcal{R}}_n(\hat{f}) - \mathcal{R}(\hat{f})\right| \leq \sup_{f \in \mathcal{F}} \left|\widehat{\mathcal{R}}_n(f) - \mathcal{R}(f)\right| =: \varepsilon_n(\mathcal{F})$$

- Choose $\mathcal{F}$ s.t. $\varepsilon_n(\mathcal{F}) \ll 1$. That is:

  training error $\approx$ test error for all models $f \in \mathcal{F}$.

- $\varepsilon_n(\mathcal{F})$ increases with the complexity of the class $\mathcal{F}$, but vanishes as $n \to \infty$.

  Trade-off between approximation and generalization error.

Uniform Convergence (II)

- Instead of choosing the complexity of $\mathcal{F}$ in advance: can choose it adaptively.
- Consider a rich class $\mathcal{F}$ and a "complexity measure" $\Phi : \mathcal{F} \to \mathbb{R}_{\geq 0}$

$$\mathcal{F}_B := \{f \in \mathcal{F} \ : \ \Phi(f) \leq B\} \quad \text{and vary } B.$$

Alternatively: **Structural Risk Minimization (SRM):**

$$\arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\boldsymbol{x}_i)) + \lambda \Phi(f).$$

$\lambda \to 0$: low approx error and large generalization error (overfitting)

$\lambda \to \infty$: large approx error and small generalization error (underfitting)

- $\mathcal{F} = \{f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle : \boldsymbol{w} \in \mathbb{R}^d\}$: can take $\Phi(f) = ||\boldsymbol{w}||_2^2$ or $\Phi(f) = ||\boldsymbol{w}||_1$

$$\varepsilon_n(\mathcal{F}_B) \leq C \frac{B}{\sqrt{n}}.$$

Three Pillars of Classical Approach

(A bit caricatural, for dramatic effect!)

(I) **Empirical Risk Minimization.**

(II) **Uniform Convergence.**

- Model class $\mathcal{F}$ chosen (or explicitly regularized) such that $\widehat{\mathcal{R}}_n(\hat{f}) \approx \mathcal{R}(\hat{f})$.

(III) **Convexity.**

- Loss function and parametrization of $\mathcal{F}$ chosen such that $\widehat{\mathcal{R}}_n(f)$ is convex.

Very influential view: the role of the statistician is to craft fct classes with small $\varepsilon_n(\mathcal{F})$ (or appropriate complexity measure) + tractable (convex) ERM formulation.

E.g., Spline methods, SVM, lasso, kernel ridge regression.

["The Nature of Statistical Learning Theory", Vapnik 1999]

Modern Approach

- These do not seem to hold for Deep Learning.

- **Convexity obviously does not hold:** $\ell(y, \hat{y}) = (y - \hat{y})^2$

$$\widehat{\mathcal{R}}_n(\boldsymbol{\Theta}) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \boldsymbol{a}_L \circ \sigma \circ \boldsymbol{W}^{(L-1)} \circ \cdots \circ \sigma \circ \boldsymbol{W}^{(1)}(\boldsymbol{x}_i) \right)^2$$

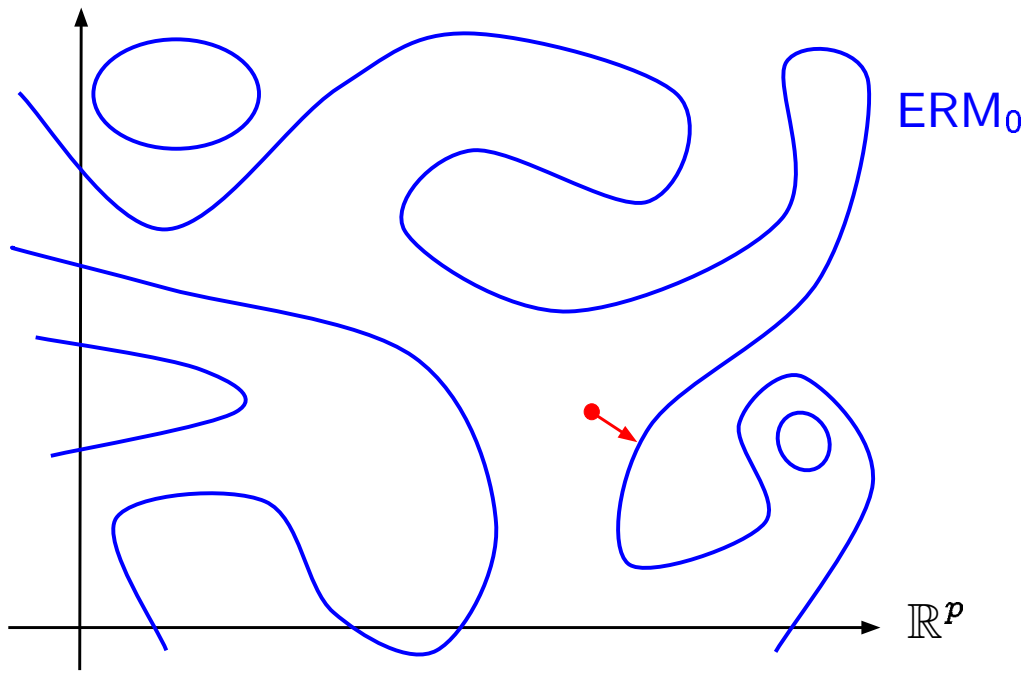
  - Highly non-convex optimization problem with many bad local minima.
  - Despite that, SGD reliably converge to global minima when the #parameters is large enough.

- **Uniform convergence does not hold:** often $\widehat{\mathcal{R}}_n(\hat{f}) \ll \mathcal{R}(\hat{f})$.
  - No apparent attempt to control the complexity of the model. In fact, often trained until they interpolate training data $\widehat{\mathcal{R}}_n(\hat{f}) = 0$.
  - These models are complex enough to interpolate pure noise.
  - Despite that, solutions found by SGD generalize well on test data.

The Interpolation Manifold

Interpolation manifold:

$$\mathsf{ERM}_0 = \{\boldsymbol{\Theta} \in \mathbb{R}^p \ : \ \widehat{\mathcal{R}}_n(\boldsymbol{\Theta}) = 0\}.$$



Solution $\widehat{\boldsymbol{\Theta}}$ found by SGD will depend on initialization and algorithm (batch size, ...)

Two Mysteries

To summarize, DL success builds on two surprising empirical discoveries:

- Despite a highly non-convex optimization problem, local optimization approaches (gradient-based methods) succeed at finding global minimum.

- Despite being trained until interpolation—with no explicit regularization controlling their statistical complexity—, the solutions found in practice generalize well on test data.

These observations suggest that DL success is based on a radical different way of managing the approximation/generalization/computation trade-offs.

3 New Paradigm: Two Working Hypotheses

Overparametrization

Hypothesis 1:

Tractability via overparametrization.

Idea: The ERM optimization problem simplifies dramatically with enough overparametrization.

Tractability is achieved not by convexity but through overparametrization.

- **Intuitive?**

  $$\text{Finding } \boldsymbol{\Theta} \in \mathbb{R}^p \text{ such that } y_i = f(\boldsymbol{x}_i; \boldsymbol{\Theta}) \text{ for all } i \leq n$$

  is more and more underconstrained as $p \to \infty$ (sufficiently overparametrized).

- **Counterintuitive from a classical learning perspective:**

  The models become more and more complex and shouldn't generalize well.

- **Empirically well demonstrated.**

  However it is **established rigorously only in a few settings:** lazy regime (NTK) and mean-field regime (in some cases).

Implicit bias

Hypothesis 2:

Generalization via implicit/algorithmic regularization.

Idea: ERM_0 contains many models with widely different generalization performance. Any optimization algo breaks the equivalence between these interpolating solutions and induces a "bias" towards certain models (e.g., with good generalization).

Bias depends on network architecture, details of the algo, and data distribution.

Generalization happens not via explicit model complexity regularization but via an implicit regularization from the learning algorithm.

- This is very different from classical wisdom:
  - We should trade-off model complexity with fit to the data.
  - Good generalization can only be achieved by underparametrized or sufficiently regularized models.

  In particular, a model that interpolates noisy data cannot generalize well. **Overfitting is bad and should be avoided.**

- Here instead, the algorithm selects an interpolating solution that generalizes well: **overfitting is benign!**

- **Implicit regularization:** only well understood for a few algorithms (e.g., mirror descent)

  Benign overfitting: only well understood in a few settings (e.g., linear or kernel regression).

Summary

Working hypotheses:

- Overparametrization makes optimization tractable.

- Training algorithms select solutions that generalize well, despite interpolating noisy data (benign overfitting).

4 Plan for the Semester

Plan (I)

- **Lecture 2: Generalization and Uniform Convergence (1/22)**
- **Lecture 3: Implicit/Algorithmic Bias (1/29)**
- **Lecture 4: Benign Overfitting/Double Descent (2/5)**
- **Lecture 5: Lazy Regime and NTK (2/12)**
- **Lecture 6: Kernel Methods (2/19)**
- **Lecture 7: Mean-Field Description (2/26)**

Plan (II)

- **Lecture 8: Kernels vs Feature Selection vs Feature Learning (3/5)**
- **Spring recess 3/12 and 3/19**
- **Lecture 9: Power and Limitations of Differentiable Learning (3/26)**
- **Lecture 10: High-dimensional Landscapes and Dynamics (4/2)**
- **Lecture 11: Transformers, Attention, and In-Context Learning (4/9)**
- **Lecture 12: Edge-of-Stability, Neural Scaling Laws, Emergence, and Beyond (4/16)**
- **Lecture 13: in-class presentations + pizza (4/23)**

5 The Early Days of Neural Network Theory

Universal Approximation

- One of the early success of theory: the proof that **neural networks are universal approximator**.

Universal Approximation

Model class $\mathcal{F}$ of NNs is rich enough to approximate any (reasonable) function arbitrarily well.

Here is a simple example of such a result by Cybenko:

Theorem [Cybenko, 1989]

Assume $\mathbb{E}[f_*(\boldsymbol{x})^2] < \infty$ and sigmoidal function $\sigma : \mathbb{R} \to \mathbb{R}$, i.e.,

$$\lim_{z \to +\infty} \sigma(z) = 1, \qquad \lim_{z \to -\infty} \sigma(z) = 0.$$

Then for any $\varepsilon > 0$, there exists $N = N(\varepsilon)$ such that

$$\inf_{\{a_i, b_i, \boldsymbol{w}_i\}_{i \in [N]}} \mathbb{E}\left[\left(f_*(\boldsymbol{x}) - \frac{1}{N} \sum_{i \in [N]} a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i)\right)^2\right] \leq \varepsilon.$$

- Consider $\rho \in \mathcal{P}(\mathbb{R}^{d+2})$ a probability distribution over $(a, b, \boldsymbol{w})$ and denote

$$\hat{f}(\boldsymbol{x}; \rho) = \int a\sigma(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)\rho(\mathrm{d}a, \mathrm{d}b, \mathrm{d}\boldsymbol{w}) = \int \sigma(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)\mu(\mathrm{d}b, \mathrm{d}\boldsymbol{w}),$$

  where $\mu = \int a\rho(\mathrm{d}a, b, \boldsymbol{w})$ signed measure.
  Taking the empirical distribution $\hat{\rho}_N = \frac{1}{N}\sum_{i\in[N]} \delta_{a_i, b_i, \boldsymbol{w}_i}$:

$$\hat{f}(\boldsymbol{x}; \hat{\rho}_N) = \frac{1}{N}\sum_{i\in[N]} a_i\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i).$$

- In the special case $b = 0$ and $\sigma(z) = e^{iz}$, then

$$\hat{f}(\boldsymbol{x}; \mu) = \int e^{i\langle \boldsymbol{w}, \boldsymbol{x} \rangle}\mu(\mathrm{d}\boldsymbol{w})$$

  (the Fourier transform of μ).
  From Fourier analysis: any squared integrable function can be represented in this way. Cybenko's result can be seen as a generalization to sigmoidal σ.

Proof (Cybenko, 1989)

- Let $\mathsf{P}(\boldsymbol{x})$ denote the distribution of $\boldsymbol{x}$ and $\mathcal{S}$ be the linear space

$$\mathcal{S} = \left\{ \sum_{i \in [N]} a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i) \ : \ N \in \mathbb{N}, a_i, b_i \in \mathbb{R}, \boldsymbol{w}_i \in \mathbb{R}^d \right\}.$$

  Denote $\bar{\mathcal{S}}$ its closure in $L^2(\mathsf{P})$. Let's prove $\bar{\mathcal{S}} = L^2(\mathsf{P})$.

- Assume by contradiction that there exists $f \in L^2(\mathsf{P}) \setminus \bar{\mathcal{S}}$. Then there exists $g \in L^2(\mathsf{P}) \setminus \bar{\mathcal{S}}$ orthogonal to $\bar{\mathcal{S}}$. In particular,

$$\int g(\boldsymbol{x}) \sigma(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b) \mathsf{P}(\mathrm{d}\boldsymbol{x}) = 0, \qquad \forall w, b.$$

- Take $\boldsymbol{w} = \alpha \boldsymbol{v}$ and $b = -\alpha c$. Take $\alpha \to \infty$ above and we get

$$\int g(\boldsymbol{x}) \mathbb{1}[\langle \boldsymbol{v}, \boldsymbol{x} \rangle \geq c] \mathsf{P}(\mathrm{d}\boldsymbol{x}) = 0.$$

- The integral of g over any half-space is 0. This implies $g(\boldsymbol{x}) = 0$ (for P-a.e. $\boldsymbol{x}$).

Approximation Guarantees

- Previous result didn't bound the number of neurons N.
- How big N should be to approximate a reasonable target f_*?

Here is a classical result by our very own Andrew Barron:

Theorem [Barron, 1993]

Assume x supported on $\mathrm{B}(0, r)$, sigmoidal σ, and $f_* : \mathbb{R}^d \to \mathbb{R}$ with $f_*(0) = 0$ and Fourier transform F, i.e.,

$$f_*(x) = \int e^{i\langle \omega, x \rangle} F(\omega) \mathrm{d}\omega.$$

Then for any $N \geq 1$, there exists weights such that

$$\mathbb{E}\left[\left(f_*(x) - \sum_{i \in [N]} a_i \sigma(\langle w_i, x \rangle + b_i)\right)^2\right] \leq \frac{(2rC_{f_*})^2}{N}$$

with $\sum_{i \in [N]} |a_i| \leq 2rC_{f_*}$, where the "Barron norm" of f_* is defined by

$$C_{f_*} := \int ||\omega||_2 |F(\omega)| \mathrm{d}\omega.$$

- Compared to Cybenko's result, it gives a quantitative approximation guarantee.
- In particular, show that approx. error is $O(1/N)$ for functions $C_f = O(1)$.
- In contrast, for fixed basis models such as polynomial, spline, trigonometric expansion, the approx. error scales as $O(1/N^{2/d})$ (needs $\asymp e^d$ units).

  NNs do not suffer from the curse of dimensionality compared to classical methods for the class of functions with $C_f = O(1)$.

- Very influential paper: it shows that *NNs are not just universal approximators but efficient approximators for many (practically relevant) functions.*

Partial proof outline (Following Telgarsky, 2022)

- For simplicity: threshold activation $\sigma(x) = \mathbb{1}[x \geq 0]$.
- Write f_* as ∞-width NN using Fourier inversion formula: $F(\boldsymbol{\omega}) = |F(\boldsymbol{\omega})| e^{2\pi i \theta(\boldsymbol{\omega})}$

$$
\begin{aligned}
f_*(\boldsymbol{x}) - f_*(0) = & -2\pi \int \int_0^{\|\boldsymbol{\omega}\|_2} \sigma(\langle \boldsymbol{\omega}, \boldsymbol{x} \rangle - b) \Big[\sin(2\pi b + 2\pi\theta(\boldsymbol{\omega})) |F(\boldsymbol{\omega})| \Big] \mathrm{d}b \mathrm{d}\boldsymbol{\omega} \\
& + 2\pi \int \int_{-\|\boldsymbol{\omega}\|_2}^0 \sigma(-\langle \boldsymbol{\omega}, \boldsymbol{x} \rangle + b) \Big[\sin(2\pi b + 2\pi\theta(\boldsymbol{\omega})) |F(\boldsymbol{\omega})| \Big] \mathrm{d}b \mathrm{d}\boldsymbol{\omega}
\end{aligned}
$$

- Sample from the sign measure to get finite-width approximation.

  Using **Maurey/Pisier lemma (1980)**: for $g(\boldsymbol{x}) = \int \varphi(\boldsymbol{x}; \boldsymbol{u}) \mu(\mathrm{d}\boldsymbol{u})$ and sampling $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$ iid from μ, then

$$
\mathbb{E}_{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k} \left[\left\| g - \frac{1}{k} \sum_{i=1}^k \varphi(\cdot; \boldsymbol{u}_i) \right\|_{L^2}^2 \right] \leq \frac{\mathbb{E}\left[\|\varphi(\cdot; \boldsymbol{u}_1)\|_{L^2}^2 \right]}{k}.
$$

Learning Guarantees

- Previous results show that good networks exist (i.e., with low approx error), but does not guarantee that we can learn them efficiently from data.

- We can give a learning guarantee by combining the approximation guarantee of Barron (1993) and "Uniform Convergence" over functions with bounded number of neurons.

Theorem [Barron, 1994]

Consider the setting of the previous theorem. We fit a target function f_* with Barron norm C_{f_*} from n samples $(f_*(\boldsymbol{x}_i), \boldsymbol{x}_i)_{i \leq n}$ using the ERM estimator

$$\hat{f} = \arg\min \left\{ \frac{1}{n} \sum_{i \in [n]} (f_*(\boldsymbol{x}_i) - f(\boldsymbol{x}_i))^2 \;\middle|\; f = \sum_{j \in [N]} a_j \sigma(\langle \boldsymbol{w}_j, \cdot \rangle + b_j) \right\}. \qquad (\star)$$

Then

$$\mathcal{R}(\hat{f}) = \mathbb{E}[(f_*(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}))^2] = O\left(\frac{(2rC_{f_*})^2}{N} + \frac{Nd}{n} \log(n) \right).$$

- Taking $N = [rC_{f_*}] \sqrt{\frac{n}{d \log(n)}}$, we get

$$\mathcal{R}(\hat{f}) = O\left([rC_{f_*}] \sqrt{\frac{d \log(n)}{n}} \right).$$

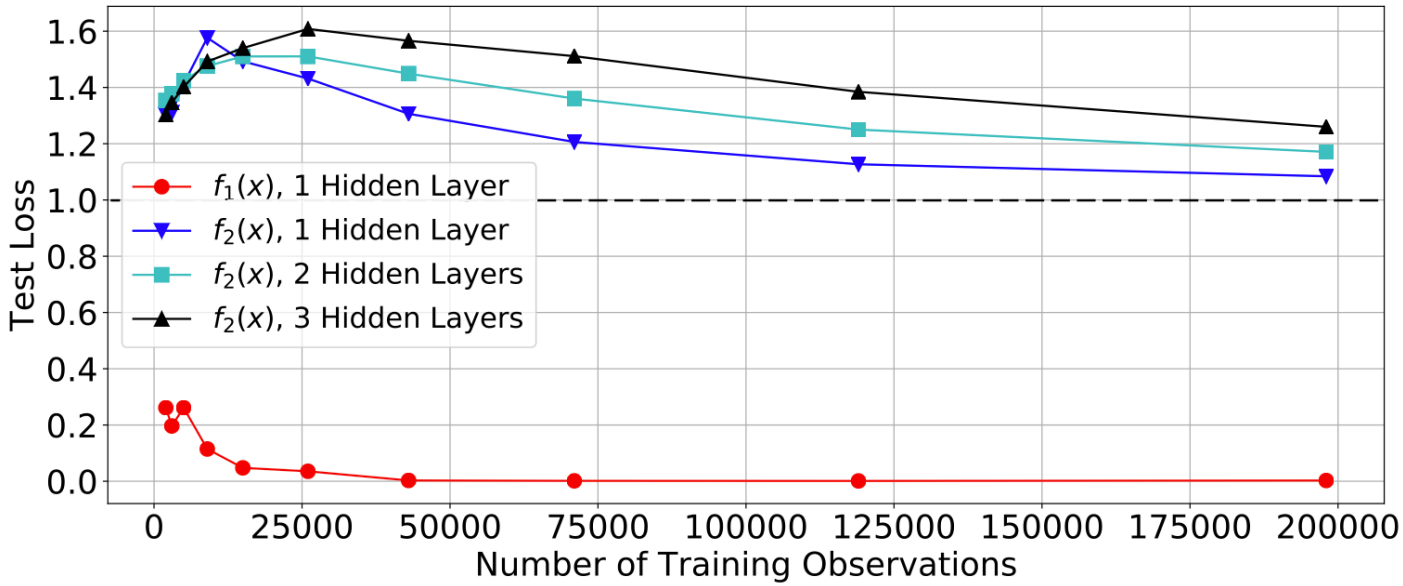
- Proof next week.

Limitations

- Above learning guarantee is for $\hat{f}$ a solution of the ERM problem $(\star)$.
  This is a highly non-convex problem which we expect simplify only in the overparametrized regime $N \gg n$ where the bound is vacuous.

- In practice, we run SGD on the ERM problem to construct $\hat{f}$.

How informative is this learning guarantee for SGD-trained neural networks?



Two functions with same approximation and statistical complexities. However one is learned efficiently by SGD and the other not.

- The previous learning guarantee completely misses the computational aspect (that neural networks are trained by gradient-based methods).

- Still interesting bound: this is a "learning guarantee if we had unbounded computational resources".

- In contrast, modern statistics emphasizes **computational bottlenecks**:

  Computational efficiency is a more stringent requirement than statistical efficiency.

- See Ilias Zadik's course S&DS 688: Computational-to-statistical gaps
  - Tensor PCA, matrix factorization, sparse regression, etc.

Nathan Srebro (TTIC):

"Statistics is easy, computation is hard."

"The mystery of Deep Learning success is a computational mystery."

The above viewpoint will be quite influential in how I organize this course.

See you in person next Wednesday!