Theoretical Principles of Deep Learning

Class II: Approximation with Neural nets

Hédi Hadiji

Université Paris-Saclay - CentraleSupelec

hedi.hadiji@l2s.centralesupelec.fr

December, 2024

Table of Contents

Last time

Last time:

- Supervised learning
- Neural nets
- The Perceptron: optimization and generalization on **(linearly) separable** data

Today: Approximation of neural nets.
Or 'Is there any hope to follow data with arbitrary patterns?'

Reading Material:

- Matus Telgarsky's notes.
- Universal approximation bounds for superpositions of a sigmoidal function, Barron 1993.

Table of Contents

Recall: Definitions of Neural Nets

Feedforward neural networks

For dimensions p, q, r, a **layer** is a function $\mathbb{R}^p \to \mathbb{R}^r$

$$\Phi_{\sigma,A,b} : x \mapsto \sigma\big(Ax + b\big)$$

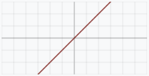
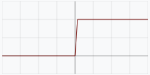
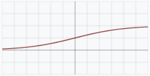
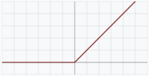
where $\sigma : \mathbb{R}^q \to \mathbb{R}^r$ is a simple non-linear function, A is an $q \times p$ matrix and $b \in \mathbb{R}^q$ is a vector.

A **neural network** is a function of the form

$$h : x \mapsto \Phi_{\sigma_L,A_L,b_L} \circ \cdots \circ \Phi_{\sigma_0,A_0,b_0}(x)\,.$$

Terminology: since σ_L is often the identity, L is the number of *hidden layers* aka *activation layers* (while there are $L + 1$ layer functions composed together.)

Activation functions

Identity		x
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$
Logistic, sigmoid, or soft step		$\sigma(x) \doteq \dfrac{1}{1+e^{-x}}$
Hyperbolic tangent (tanh)		$\tanh(x) \doteq \dfrac{e^x - e^{-x}}{e^x + e^{-x}}$
Rectified linear unit (ReLU)[8]		$(x)^+ \doteq \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max(0, x) = x\mathbf{1}_{x>0}$

Training neural nets in Supervised Learning

Given an n-sample of features and responses. Fix a structure for the net and compute an (approximate) Empirical Risk Minimizer

$$\arg\min_{\text{weights}} \frac{1}{n}\sum_{i=1}^{n} \ell\big(h(x_i), y_i\big) .$$

Typically using a variant of gradient descent on the weights.

Mystery: Why/how/when does it work?

One hidden layer neural networks

Focus of today: **Shallow nets** (2-layer, one hidden layer)

$$h(x) = \sum_{i=1}^{m} c_i \, \sigma(a_i^\top x + b_i)$$

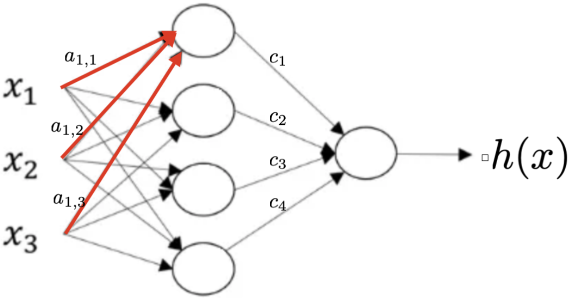


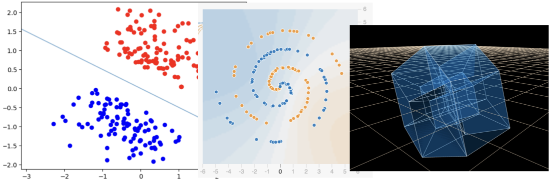
Table of Contents

Practice: Attempt to minimize Empirical Loss for a fixed architecture

- Practice: train network as long as possible, hope that loss goes as low as possible. Best case you get to 0.
- Ignore how it is computed. And whether it generalizes..

Today's question: Approximation

With high-dimensional data generated by a complicated function. Is there any hope that a neural net will get good training performance?



Setting: Approximation

Norm is euclidean.

Assumption: Bounded features

For all $x \in \mathcal{X}$, we have $\|x\| \leqslant 1$.

Goal: Approximation

Given a continuous function $f : \mathcal{X} \to \mathbb{R}$, can we find a neural network that is close to f?

Even better if

- Small net
- With small weights
- With our favorite activation function

Naive approximation in 1-d

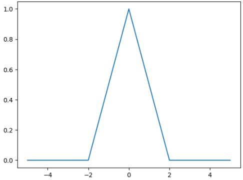
Theorem (Naive approximation)

Let $f : [0, 1] \to \mathbb{R}$ be a 1-Lipschitz function and $\varepsilon > 0$.
There exists a two-layer ReLU neural net h with $\Theta(1/\varepsilon)$ nodes such that

$$\|f - h\|_{L^1} \leqslant \varepsilon \,.$$

Idea: Discretize space into small intervals and localize approximations

$$\frac{\mathrm{ReLU}(x+2) - 2\,\mathrm{ReLU}(x) + \mathrm{ReLU}(x-2)}{2}$$



Remark: large slopes imply large weights

Naive approximation and the Curse of Dimensionality

Theorem (Naive approximation)

Let $f : [0,1]^d \to \mathbb{R}$ be a 1-Lipschitz function in $\|\cdot\|_\infty$ and $\varepsilon > 0$.
There exists a three-layer ReLU neural net h with $\Theta(1/\varepsilon^d)$ nodes such that

$$\|f - h\|_{L^1} \leqslant \varepsilon\,.$$

Proof idea: Discretize space into small cubes and localize approximations.

Exercise: Write a bump function with three layers.

Issue: high d requires many cubes, thus many nodes. E.g. Cifar-10 images have dim $32 \times 32 \times 3 = 3072$, Imagenet $> 500\,000$
+ Construction is not adaptive: does not exploit large flat surfaces.

Generality of approximation

Definition

An activation is **sigmoidal** if it is continuous,

$$\lim_{x \to -\infty} \sigma(x) = 0 \quad \text{and} \quad \lim_{x \to +\infty} \sigma(x) = 1$$

Theorem (Universal approximation)

For any sigmoidal activation function σ, any continuous function f and any ε, there exists a two-layer neural net h with activation σ such that

$$\|h - f\|_{L^\infty} \leqslant \varepsilon\,.$$

Proof sketch:

- Approximate the $\cos$ function by a net (width $\Theta(1/\varepsilon)$ is achievable)
- Stone-Weierstrass to algebra generated by $x \mapsto \cos(ax + b)$

Universal approximation: so what?

Theorem (Universal approximation)

For any sigmoidal activation function σ, any continuous function f and any ε, ***there exists*** *a two-layer neural net h with activation σ such that*

$$\|h - f\|_{L^\infty} \leqslant \varepsilon\,.$$

But this is not constructive enough. What matters for real life is whether the number of nodes and the magnitude of the weights stay reasonable.

(Could check proofs of Stone-Weierstrass to get an explicit construction, but typically this yields at least an exponentially bad dependence on d.)

Table of Contents

Barron smoothness: breaking the curse of dimensionality

Barron's result: (ca. 1993) neural nets can avoid the curse of dimensionality if we consider **a specific notion of regularity.**

Reminder: Fourier transform and inverse

Let f be a continuous function from compact $\mathcal{X}$ to $\mathbb{R}$. The Fourier transform of f is the function $\mathbb{R}^d \to \mathbb{C}$

$$\widehat{f}(w) = \int_{\mathcal{X}} e^{-2i\pi w\cdot x} f(x)\mathrm{d}x\,.$$

If $\widehat{f}(w) \in L_1(\mathbb{R}^d)$, then for any $x \in \mathcal{X}$,

$$f(x) = \int_{\mathbb{R}^d} e^{2i\pi w\cdot x} \widehat{f}(w)\mathrm{d}x\,.$$

(Remember $\mathcal{X}$ is a compact subset of $\mathbb{R}^d$, so a continuous function $\mathcal{X} \to \mathbb{R}$ is bounded.)

Barron smoothness

Assumption: Barron smoothness

We consider functions $f : \mathcal{X} \to \mathbb{R}$ such that

$$C(f) = \int_{\mathbb{R}^d} \|w\|_2 \left|\widehat{f}(w)\right| \mathrm{d}w < +\infty \,.$$

Note: this is an assumption on the rate of decay of $\widehat{f}$ at infinity. Therefore it controls the regularity of f.

Conventions may vary depending on sources (e.g. many define the Barron norm with $1/(2\pi)$.)

Barron's theorem

Assumption: Barron smoothness

We consider functions $f : \mathcal{X} \to \mathbb{R}$ such that

$$C(f) = \int_{\mathbb{R}^d} \|w\|_2 \big|\widehat{f}(w)\big| \mathrm{d}w < +\infty \,.$$

Theorem (Barron's theorem ['93])

Let f be a continuous function. For any $\varepsilon > 0$, there exists a two-layer neural net of width less than

$$k \leqslant \frac{8\,\mathrm{Vol}(\mathcal{X})}{\varepsilon^2}(8\pi C(f))^2 \quad \textit{such that} \quad \|f - h\|_{L^2} \leqslant \varepsilon \,.$$

Proof (Telgarsky): Two important and useful ingredients.

- Infinite-width representation via Fourier
- Approximate Carathéodory

Infinite-width representation

Proposition (Infinite-width representation)

Under the previous assumptions, for any $x \in \mathcal{X}$

$$f(x) - f(0) = \int_{w \in \mathbb{R}^d} \int_{b \in \mathbb{R}} \mathbb{1}\{w \cdot x - b \geqslant 0\} \mu(w, b) \mathrm{d}b \mathrm{d}w \,.$$

where, denoting by $\theta(w)$ is an argument of $\widehat{f}(w)$,

$$\mu(w, b) = \Big(-2\pi \sin(2\pi b + \theta(w)) |\widehat{f}(w)| + 2\pi \sin(-2\pi b + \theta(-w)) |\widehat{f}(-w)| \Big) \mathbb{1}\{0 \leqslant b \leqslant \|w\|\} \,.$$

Proof: Board

This is an **exact** representation of f as an infinite-width two-layer neural network with step function activations, i.e.,

$$\frac{1}{n} \sum_{i=1}^{n} c_i \sigma(x^\top a_i + b_i) \leftrightarrow \int c(u) \sigma(x^\top a(u) + b(u)) \mathrm{d}u$$

From infinite width to finite width

Idea: consider an infinite-width neural net of the form

$$g(x) = \int \sigma(w \cdot x) \mathrm{d}\mu(w)$$

where μ is a probability measure. In other words, g is a convex combination of the functions $\{x \mapsto \sigma(w \cdot x) : w \in \mathbb{R}^d\}$.

We want to approximate g by a neural net with finite width.

Remember *Carathéodory's theorem*?

Approximate Carathéodory

Theorem (Approximate Carathéodory)

If $y^\star$ is in the closed convex envelope of a compact set Y of diameter B in a real Hilbert space, then for any $\varepsilon > 0$, there exists $y_1, \ldots, y_k$ such that

$$\left\| y^\star - \frac{1}{k} \sum_{i=1}^{k} y_i \right\|^2 \leqslant \frac{B^2}{k} \,.$$

Beautiful proof: Empirical method of Maurey.

With $k \geqslant 1/\varepsilon^2$ points, we get squared error $\leqslant \varepsilon^2$.

Approximate Carathéodory applied to infinite-width nets

Consider the Hilbert space $L^2(\mathcal{X})$ and $Y = \{x \mapsto \sigma(w \cdot x); w \in \mathbb{R}^d\}$,

$$g(x) = \int \sigma(w \cdot x) \mathrm{d}\mu(w).$$

For any k, there exists $w_1, \ldots, w_k$ such that

$$\int \left(g(x) - \frac{1}{k} \sum_{i=1}^{k} \sigma(w_i \cdot x) \right)^2 \mathrm{d}x \leqslant \frac{B^2}{k},$$

where

$$B \geqslant \sup_{w \in \mathbb{R}^d} \int_{x \in \mathcal{X}} \sigma(w \cdot x)^2 \mathrm{d}x .$$

Concluding the proof of Barron's theorem

Board + notes

Further comments

On approximation

- Nice to get rid of dimension... but we should check the Barron smoothness. (Exercise: what is $C(f)$ when f is a gaussian function?) see Barron's paper for a lot of examples.
- What about **deep nets**? There exists small 3-layers nets that cannot be approximated by small 2-layer nets. This is a benefit of depth.

Beyond approximation

- Generalization: possible to get error bounds on the least-square neural network. But computing this least-squares neural net is computationally hard.
- SGD does not find the weights in the Barron approximation (to my knowledge).

Table of Contents

Conclusion and Next time

Today: Approximation of neural networks

- Shallow neural nets do not suffer from the curse of dimensionality in approximation: the quality of approximation increases linearly with the number of nodes, and (kind of) independently of the dimension.
- A function can be always be seen as a an infinitely wide shallow neural network.

Next time: **optimization**

Conclusion and Next time

Break