Mathematics of Deep Learning

Approximation guarantees

Lessons: Kevin Scaman

Dauphine UNIVERSITÉ PARIS | PSL

Class overview

Function approximation

Objective

- The aim is to describe all functions that can be approximated by a given neural network architecture, i.e. for $\varepsilon > 0$, $\exists\theta$ s.t.

$$d(f, g_\theta) \leqslant \varepsilon$$

where d is a distance over functions.

Examples

- Uniform approximation: $d(f, g_\theta) = \|f - g_\theta\|_\infty = \max_{x\in\mathcal{X}} |f(x) - g(x)|$.
- L_p approximation: $d(f, g_\theta) = \|f - g_\theta\|_p = (\int_{x\in\mathcal{X}} |f(x) - g(x)|^p d\mu(x))^{1/p}$.
- Exact learning: $d(f, g_\theta) = \mathbb{1}\{\exists x \in \mathcal{X}, f(x) \neq g_\theta(x)\}$.

Exact learning

Exact recovery of a function with a neural network

Exact Learning

- Any piecewise linear function can be created using ReLU networks.
- For other activation functions, we cannot say much...
- ...however, if the activation function is not fixed, then **we can recreate any continuous function**!

Kolmogorov-Arnold-Sprecher theorems

- **Answer to Hilbert's 13^{th} problem:** are there continuous functions of several variables that are not finite compositions of continuous functions of a lesser number of variables?

Theorem (Kolmogorov, 1957)

Any continuous function $f(x_1, \ldots, x_n)$ defined on $[0,1]^n, n \geqslant 2$, can be written in the form

$$f(x_1, \ldots, x_n) = \sum_{j=1}^{2n+1} \chi_j \left(\sum_{i=1}^{n} \psi_{ij}(x_i) \right)$$

where $\chi_j, \psi_{ij} : \mathbb{R} \to \mathbb{R}$ are continuous functions of one variable and ψ_{ij} are monotone functions which are not dependent on f.

Kolmogorov-Arnold-Sprecher theorems

Theorem (Sprecher, 1964)

For each integer $n \geqslant 2$, there exists a real, monotone increasing function ψ, $\psi([0,1]) = [0,1]$, dependent on n and having the following property: for each preassigned number $\delta > 0$,there is a rational number $\varepsilon \in (0, \delta)$, such that every real continuous function $f(x_1, \ldots, x_n)$, defined on $[0,1]^n$, can be written in the form

$$f(x_1, \ldots, x_n) = \sum_{j=1}^{2n+1} \chi\left(\sum_{i=1}^{n} \lambda^i \psi\left(x_i + \varepsilon(j-1)\right) + j - 1 \right)$$

where χ is real and continuous and λ is a constant independent of f.

Kolmogorov-Arnold-Sprecher theorems

Application to neural networks

- For any continuous function $f : \mathbb{R}^d \to \mathbb{R}$ and $K \subset \mathbb{R}^d$ compact, there is a 3-layer MLP that recreates **exactly** the function on K.
- The MLP has $(d+1)(2d+1)$ neurons and $(2d+1)(3d^2+d+1)$ parameters.

Limitations

- The activation function χ **depends on the function to approximate** f.
- The function ψ is **very irregular** (despite being continuous), e.g. not Lipschitz.
- This result is of limited use in practice...but has useful extensions for **geometric deep learning**!

DeepSets

Theorem (Zaheer et.al., 2018)

A function $f : [0,1]^n \to \mathbb{R}$ is continuous and permutation invariant if and only if it can be decomposed in the form

$$f(x_1, \ldots, x_n) = \chi\left(\sum_{i=1}^{n} \psi(x_i)\right)$$

where $\chi : \mathbb{R}^{n+1} \to \mathbb{R}$ and $\psi : \mathbb{R} \to \mathbb{R}^{n+1}$ are continuous functions.

DeepSets

Theorem (Zaheer et.al., 2018)

A function $f : [0,1]^n \to \mathbb{R}$ is continuous and permutation invariant if and only if it can be decomposed in the form

$$f(x_1, \ldots, x_n) = \chi\left(\sum_{i=1}^{n} \psi(x_i)\right)$$

where $\chi : \mathbb{R}^{n+1} \to \mathbb{R}$ and $\psi : \mathbb{R} \to \mathbb{R}^{n+1}$ are continuous functions.

Extensions

- If $d \geqslant 1$, $f : [0,1]^n \to \mathbb{R}^d$ can also be decomposed using $d(n+1)$ inner dimensions.
- If $K \subset \mathbb{R}^d$ is compact, then $f : K^n \to \mathbb{R}$ can also be decomposed using more inner dimensions.

DeepSets (proof sketch)

- We use $\psi(x) = [1, x, \dots, x^{n+1}]$.
- $E(x) = \sum_{i=1}^{n} \psi(x_i)$ is a polynomial.
- The function E is bijective and bi-continuous.
- We take $\chi = f \circ E^{-1}$.

Link with polynomial approximation

Stone-Weierstrass theorem and applications

Universality

Definition (universality)

Let $d \geqslant 1$. A subset of continuous functions $\mathcal{F} \subset \mathcal{C}(\mathbb{R}^d)$ is called *universal* if, for any compact $K \subset \mathbb{R}^d$, $\mathcal{F}$ is uniformly dense in $\mathcal{C}(\mathbb{R}^d)$. In other words, for any continuous function $g \in \mathcal{C}(\mathbb{R}^d)$ and $\varepsilon > 0$, there exists $f \in \mathcal{F}$ such that

$$\forall x \in K, \qquad |f(x) - g(x)| \leqslant \varepsilon$$

- For example, polynomials are uniformly dense in $\mathcal{C}(\mathbb{R})$.
- This result easily extends to vector-valued outputs.

Stone-Weierstrass theorem

Theorem (Stone-Weierstrass, simple version)

Suppose f is a continuous real-valued function defined on the real interval $[a, b]$. For every $\varepsilon > 0$, there exists a polynomial p such that for all x in $[a, b]$, we have $|f(x) - p(x)| \leqslant \varepsilon$.

- In other words, polynomials are universal for $\mathcal{C}(\mathbb{R})$.

Stone-Weierstrass theorem

Theorem (Stone-Weierstrass, simple version)

Suppose f is a continuous real-valued function defined on the real interval $[a, b]$. For every $\varepsilon > 0$, there exists a polynomial p such that for all x in $[a, b]$, we have $|f(x) - p(x)| \leqslant \varepsilon$.

- In other words, polynomials are universal for $\mathcal{C}(\mathbb{R})$.

Proof.

- If OK on $[0, 1]$, then OK on $[a, b]$.
- Uniform continuity of f on $[0, 1]$: $\forall \varepsilon > 0, \exists \delta > 0$ s.t. $|x - y| \leqslant \delta \implies |f(x) - f(y)| \leqslant \varepsilon$.
- Let $x \in [0, 1]$ and $K \sim \text{Bin}(n, x)$. Then $K/n \to x$ a.s. (by the LLN) and $\mathbb{E}(f(K/n)) = \sum_{k=0}^{n} f(k/n)\binom{n}{k}x^k(1-x)^{n-k} = P_{n,f}(x)$ is a (Bernstein) polynomial.
- $|P_{n,f}(x) - f(x)| \leqslant \mathbb{E}(|f(K/n) - f(x)|) \leqslant \varepsilon + 2\|f\|_\infty \mathbb{P}(|K/n - x| > \delta) \leqslant \varepsilon + \frac{\|f\|_\infty}{2n\delta^2}$.

□

Side comment: convergence rate

Definition (Lipschitz regularity)

A function $f : \mathcal{X} \to \mathcal{Y}$ is L-Lipschitz iff, $\forall x, y \in \mathcal{X}$, $\|f(x) - f(y)\| \leqslant L\|x - y\|$.

Adding more regularity

- With a slight modification of the proof, we can see that, if f is L-Lipschitz, then

$$|P_{n,f}(x) - f(x)| \leqslant \frac{L}{2\sqrt{n}}$$

- Gives a quantitative trade-off between **quality of the approximation** (i.e. small aprox. error) and **model complexity** (i.e. order of the polynomial).

Stone-Weierstrass theorem

Definition (point separation)

A set $\mathcal{F}$ of functions defined on $\mathcal{X}$ is said to separate points if, for every two different points x and y in $\mathcal{X}$ there exists a function $f \in \mathcal{F}$ such that $f(x) \neq f(y)$.

Theorem (Stone-Weierstrass, general version)

Suppose $\mathcal{X}$ is a compact Hausdorff space and $\mathcal{F}$ is a **subalgebra** of $\mathcal{C}(\mathcal{X}, \mathbb{R})$ which contains a non-zero constant function. Then $\mathcal{F}$ is dense in $\mathcal{C}(\mathcal{X}, \mathbb{R})$ if and only if it separates points.

Stone-Weierstrass theorem

Definition (point separation)

A set $\mathcal{F}$ of functions defined on $\mathcal{X}$ is said to separate points if, for every two different points x and y in $\mathcal{X}$ there exists a function $f \in \mathcal{F}$ such that $f(x) \neq f(y)$.

Theorem (Stone-Weierstrass, general version)

Suppose $\mathcal{X}$ is a compact Hausdorff space and $\mathcal{F}$ is a **subalgebra** of $\mathcal{C}(\mathcal{X}, \mathbb{R})$ which contains a non-zero constant function. Then $\mathcal{F}$ is dense in $\mathcal{C}(\mathcal{X}, \mathbb{R})$ if and only if it separates points.

Remarks

- Point separation is a necessary condition for universality.
- Allows to extend polynomial approximation to $\mathcal{C}(\mathbb{R}^d, \mathbb{R})$.
- Provides another proof for universality of deep set.

Universality theorems

Approximation guarantees of MLPs

Universality of 2-layer MLPs

Definition (sigmoidal function)

A function $\sigma : \mathbb{R} \to [0,1]$ is *sigmoidal* if $\lim_{x\to-\infty} \sigma(x) = 0$ and $\lim_{x\to+\infty} \sigma(x) = 1$.

Theorem (Cybenko, 1989)

Let σ be an arbitrary continuous sigmoidal function. Then the finite sums of the form

$$f(x) = \sum_{j=1}^{N} c_j \sigma(w_j^\top x + b_j)$$

for $N \geqslant 1$, $c_j, b_j \in \mathbb{R}$, and $w_j \in \mathbb{R}^d$ is dense in $\mathcal{C}([0,1]^d)$.

In other words, 2-layer MLPs are universal approximators of continuous functions.

Universality of 2-layer MLPs

Cybenko's universality theorem does not work for ReLUs (as well as most modern activation functions)!

Theorem (Pinker, 1999)

Finite sums of the form $\sum_{j=1}^{N} c_j \sigma(w_j^\top x + b_j)$ for $N \geqslant 1$, $c_j, b_j \in \mathbb{R}$, and $w_j \in \mathbb{R}^d$ are dense in $\mathcal{C}([0,1]^d)$ if and only if σ is not a polynomial.

Universality of 2-layer MLPs

Cybenko's universality theorem does not work for ReLUs (as well as most modern activation functions)!

Theorem (Pinker, 1999)

Finite sums of the form $\sum_{j=1}^{N} c_j \sigma(w_j^\top x + b_j)$ for $N \geqslant 1$, $c_j, b_j \in \mathbb{R}$, and $w_j \in \mathbb{R}^d$ are dense in $\mathcal{C}([0,1]^d)$ if and only if σ is not a polynomial.

Limitations

- Does not provide a quantitative measure of approximation error.
- No dependence on architecture hyper-parameters (number of layers, etc...)

Example of a quantitive results

Lemma

For any L-Lipschitz function $f : [0,1] \to \mathbb{R}$, there exists a ReLU network g_θ of depth 2 and width n such that, we have $\|f - g_\theta\|_\infty \leqslant L/n$.

Example of a quantitive results

Lemma

For any L-Lipschitz function $f : [0,1] \to \mathbb{R}$, there exists a ReLU network g_θ of depth 2 and width n such that, we have $\|f - g_\theta\|_\infty \leqslant L/n$.

- For higher input dimension, we usually have $\varepsilon = \Theta(n^{-1/d})$.
- Thus, $n = \Theta(\varepsilon^{-d})$ neurons are needed to approximate a function to precision ε.
- Trade-off between width and approximation error.

The power of depth ($L = 3$)

There are 3-layer MLPs of width poly(d), which cannot be arbitrarily well approximated by 2-layer networks, unless their width is $\Omega(\exp(d))$.

The power of depth ($L = 3$)

There are 3-layer MLPs of width poly(d), which cannot be arbitrarily well approximated by 2-layer networks, unless their width is $\Omega(\exp(d))$.

Theorem (Eldan & Shamir, 2016)

Under (reasonable) assumptions on activation functions σ, there exists a measure μ and a function $g : \mathbb{R}^d \to \mathbb{R}$ that can be expressed by a 3-layer MLP of width $Cd^{19/4}$ such that any function f expressible by a 2-layer MLP of width ce^{cd} verifies:

$$\int_x (f(x) - g(x))^2 d\mu(x) \geqslant c$$

where $c, C > 0$ are universal constants.

- For ReLU networks, the function g is poly(d)-Lipschitz.

The power of depth $(L > 3)$

For any $k \geqslant 1$, there are $\Theta(k^3)$-layer MLPs of width $\Theta(k^3)$, which cannot be arbitrarily well approximated by $O(k)$-layer networks, unless their width is $\Omega(\exp(k))$.

The power of depth $(L > 3)$

For any $k \geqslant 1$, there are $\Theta(k^3)$-layer MLPs of width $\Theta(k^3)$, which cannot be arbitrarily well approximated by $O(k)$-layer networks, unless their width is $\Omega(\exp(k))$.

Theorem (Telgarsky, 2015)

Let any integer $k \geqslant 1$ and any dimension $d \geqslant 1$ be given. There exists $f : \mathbb{R}^d \to \mathbb{R}$ computed by a ReLU network in $2k^3 + 8$ layers and $3k^3 + 12$ neurons so that, for any ReLU network f of depth less than k and less than 2^k neurons, we have

$$\int_{x\in[0,1]} |f(x) - g(x)|dx \geqslant 1/64$$

- The proof relies on the sawtooth function with 2^k teeths seen in TD.

Universality of fixed-width MLPs (Park et.al., 2021)

Theorem (Park et.al., 2021)

ReLU networks of width $w = \max\{d^{(1)}, \ldots, d^{(L-1)}\}$ are dense in $\mathcal{C}([0,1], \mathbb{R})$ iff $w \geqslant 3$.

Reference	Function class	Activation ρ	Upper / lower bounds
Lu et al. (2017)	$L^1(\mathbb{R}^{d_x}, \mathbb{R})$ $L^1(\mathcal{K}, \mathbb{R})$	ReLU ReLU	$d_x + 1 \leq w_{\min} \leq d_x + 4$ $w_{\min} \geq d_x$
Hanin and Sellke (2017)	$C(\mathcal{K}, \mathbb{R}^{d_y})$	ReLU	$d_x + 1 \leq w_{\min} \leq d_x + d_y$
Johnson (2019)	$C(\mathcal{K}, \mathbb{R})$	uniformly conti.†	$w_{\min} \geq d_x + 1$
Kidger and Lyons (2020)	$C(\mathcal{K}, \mathbb{R}^{d_y})$ $C(\mathcal{K}, \mathbb{R}^{d_y})$ $L^p(\mathbb{R}^{d_x}, \mathbb{R}^{d_y})$	conti. nonpoly‡ nonaffine poly ReLU	$w_{\min} \leq d_x + d_y + 1$ $w_{\min} \leq d_x + d_y + 2$ $w_{\min} \leq d_x + d_y + 1$
Ours (Theorem 1) **Ours** (Theorem 2) **Ours** (Theorem 3) **Ours** (Theorem 4)	$L^p(\mathbb{R}^{d_x}, \mathbb{R}^{d_y})$ $C([0,1], \mathbb{R}^2)$ $C(\mathcal{K}, \mathbb{R}^{d_y})$ $L^p(\mathcal{K}, \mathbb{R}^{d_y})$	ReLU ReLU ReLU+Step conti. nonpoly‡	$w_{\min} = \max\{d_x + 1, d_y\}$ $w_{\min} = 3 > \max\{d_x + 1, d_y\}$ $w_{\min} = \max\{d_x + 1, d_y\}$ $w_{\min} \leq \max\{d_x + 2, d_y + 1\}$

† requires that ρ is uniformly approximated by a sequence of one-to-one functions.
‡ requires that ρ is continuously differentiable at some z with $\rho'(z) \neq 0$.

Turing completeness of RNNs

Beyond function approximation

Learning the algorithms behind the function

Beyond function approximation

- Can a neural network learn products, i.e. $f : \mathbb{R}^2 \to \mathbb{R}$ s.t. $f(x, y) = xy$?

Learning the algorithms behind the function

Beyond function approximation

- Can a neural network learn products, i.e. $f : \mathbb{R}^2 \to \mathbb{R}$ s.t. $f(x, y) = xy$?
- Universality implies that the answer is **yes** on any bounded subset $K \subset \mathbb{R}^2$.

Learning the algorithms behind the function

Beyond function approximation

- Can a neural network learn products, i.e. $f : \mathbb{R}^2 \to \mathbb{R}$ s.t. $f(x, y) = xy$?
- Universality implies that the answer is **yes** on any bounded subset $K \subset \mathbb{R}^2$.
- But can we learn this concept beyond our training set? On the whole space $\mathbb{R}^2$?

Learning the algorithms behind the function

Beyond function approximation

- Can a neural network learn products, i.e. $f : \mathbb{R}^2 \to \mathbb{R}$ s.t. $f(x, y) = xy$?
- Universality implies that the answer is **yes** on any bounded subset $K \subset \mathbb{R}^2$.
- But can we learn this concept beyond our training set? On the whole space $\mathbb{R}^2$?
- We need to limit the *complexity* of the function that we try to learn on the whole space.
- Possible approach: functions that be computed by algorithms of bounded length.

Learning the algorithms behind the function

RNNs as Turing machines (Siegelmann & Sontag, 1995)

- **RNNs can simulate any given Turing machine.**
- Idea: consider an RNNs such that

$$x_i(t+1) = \sigma\left(\sum_j a_{ij} x_j(t) + \sum_j b_{ij} u_j(t) + c_i\right)$$

  where $u_i(t)$ is the input and $x_i(t)$ is the internal state and $\sigma(x) = \min\{0, \max\{x, 1\}\}$.
- By choosing the paremeters a, b, c, we can recreate the behavior of any given Turing machine.
- In particular, with 886 neurons one can recreate a universal Turing machine.

Recap

- Exact learning is possible, provided that the activation function is not fixed.
- 2-layer MLPs are **universal approximators** of continuous functions.
- However, they usually require an **exponential width** w.r.t. input dimension.
- Increasing depth can allow more flexibility.
- Some functions are impossible to approx. with shallow NNs of **polynomial width**.
- RNNs are **Turing-complete**.