Exponential expresivity with depth.

Theories of Deep Learning: C6.5, Lecture / Video 3

Prof. Jared Tanner
Mathematical Institute
University of Oxford

Mathematical Institute

Oxford Mathematics

DNNs as function approximators

Functions act as classifiers and other machine learning tasks

Classification of inputs $x \in \mathbb{R}^n$ to c classes denoted by $\{e_i\}_{i=1}^{c}$, is modelled by a function $H(x)$ for which $H(x) = e_i$ for all x in class i where $e_i(\ell) = 1$ for $i = \ell$ and 0 otherwise.

Approximation Theory concerns the ability to approximate functions from a given representation; see Approximation of Function (C6.3).

Some of the most well studied examples include approximation of a function $f(x)$ over $x \in [-1, 1]$ with some smoothness, say three times differentiable, by polynomials of degree at most k or trigonometric exponentials.

Here our focus is on the ability to approximate functions $H(x; \theta)$ given by a deep network architecture; for $x \in \mathbb{R}^n$.

What functions can a DNN approximate arbitrarily well? What is the advantage of depth?

- Network architectures are able to approximate any function (Cybenko (89') and Hornik (90')).
- There are functions which DNNs are able to construct with polynomially many parameters, that require exponentially many parameters for a shallow network to represent. (Telgarsky 15').
- Deep networks can approximate nonlinear functions on compact sets to ϵ uniform accuracy with depth and width scaling like $\log(1/\epsilon)$. (Yarotsky 16')

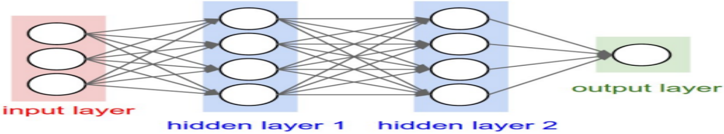
Example of a fully connected DNN:

Two layer fully connected neural net

Repeated affine transformation followed by a nonlinear action:

$$h_{i+1} = \phi_i\left(W^{(i)}h_i + b^{(i)}\right) \qquad \text{for} \quad i = 1, \ldots, N-1$$

where $W^{(i)} \in \mathbb{R}^{n_{i+1}\times n_i}$ and $b^{(i)} \in \mathbb{R}^{n_{i+1}}$ and $\phi(\cdot)$ is a nonlinear activation such as ReLU, $\phi(z) := max(0, z) = z_+$.
The input is h_1, the output is h_N, and h_i for intermediate $i = 2, \cdot, N-1$ are referred to as "hidden" layers.
The number of layers N is the depth, $N \gg 1$ is called "deep."



https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Architecture/feedforward.html

Superposition of sigmoidal functions (Cybenko 89')

DNNs with sigmoidal activations are dense in $C_n([0,1])$

Consider the feedforward network with one hidden layer:

input $h_1 = x \in \mathbb{R}^n$
hidden layer $h_2 = \phi\left(W^{(1)} h_1 + b^{(1)}\right) \in \mathbb{R}^m$
output $H(x,\theta) = \alpha^T h_2 = \sum_{i=1}^m \alpha_i \phi(w_i^T x + b_i)$
with $\phi(t) \in [0,1]$, say $\phi(t) = 1/(1+e^{-t})$.

Theorem (Cybenbko 89')

Let $\phi(t)$ be a continuous monotone function with $\lim_{t\to-\infty} \phi(t) = 0$ and $\lim_{t\to\infty} \phi(t) = 1$, then the set of functions of the form $H(x;\theta) = \sum_{i=1}^m \alpha_i \phi(w_i^T x + b_i)$ is dense in $C_n([0,1])$.

That is, one (or more) layer fully connected nets are sufficient to approximate any continuous function, provided m is large enough.
https://link.springer.com/article/10.1007/BF02551274

Approximation of multilayer feedforward nets (Hornik 90')

DNNs with continuous bounded activations are dense in $C_n([0,1])$

Consider the feedforward network with one hidden layer:
input $h_1 = x \in \mathbb{R}^n$
hidden layer $h_2 = \phi\left(W^{(1)}h_1 + b^{(1)}\right) \in \mathbb{R}^m$
output $H(x,\theta) = \alpha^T h_2 = \sum_{i=1}^m \alpha_i \phi(w_i^T x + b_i)$
with $\phi(t) \in [0,1]$ non-constant.

Theorem (Hornik 90')

Let $\phi(t)$ be unbounded then $H(x;\theta) = \sum_{i=1}^m \alpha_i \phi(w_i^T x + b_i)$ is dense in $L^p(\mu)$ for all finite measures μ and $1 \leq p < \infty$. Moreover, if $\phi(t)$ is continuous and bounded, then $H(x;\theta) = \sum_{i=1}^m \alpha_i \phi(w_i^T x + b_i)$ is dense in $C_n([0,1])$.

Much of the result includes showing $L(\phi) = \int_{I_n} \phi(x) d\mu(x) = 0$ for $\phi(x)$ in the specified class implies $\mu(x) = 0$.

https://www.sciencedirect.com/science/article/pii/089360809190009T

Representational benefits of depth (Telgarsky 15')

Two layer ReLU network: sawtooth basis function

Telegarsky (2015) considered a specific construction of a function from a deep network which requires an shallow network to have exponential width.

Let $\phi(x) = ReLU(x) = max(x, 0)$ and consider the two layer net:

$$h_2(x) = 2\phi(x) - 4\phi(x - 1/2) = \begin{cases} 0 & x < 0 \\ 2x & x \in [0, 1/2] \\ 2 - 2x & x > 1/2 \end{cases}$$

and $h_3(x) = \phi(h_2(x))$ set to zero the negative portion for $x > 1$.

`https://arxiv.org/abs/1509.08101`

Representational benefits of depth (Telgarsky 15')

Composition gives exponential growth in complexity

For $\phi(x) = \max(x, 0)$ let $f(x) = h_3(x) = \phi(2\phi(x) - 4\phi(x - 1/2))$ and iterate this 2-layer network k times to obtain a $2k$-layer network $f^k(x) = f(f(\cdots(f(x)\cdots))$ with the property that it is piecewise linear with change in slope at $x_i = i2^{-k}$ for $i = 0, 1, \ldots, 2^k$ and moreover takes on the values $f^k(x_i) = 0$ for i even and $f^k(x_i) = 1$ for i odd.

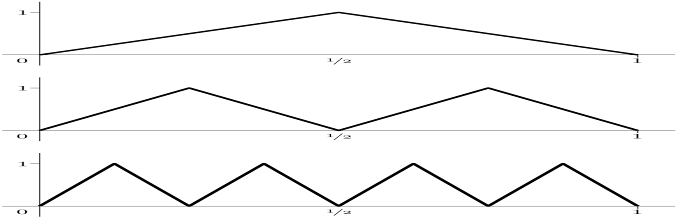


Figure 2: f_m, f_m^2, and f_m^3.

Representational benefits of depth (Telgarsky 15')

Composition gives exponential growth in complexity: width vs. depth

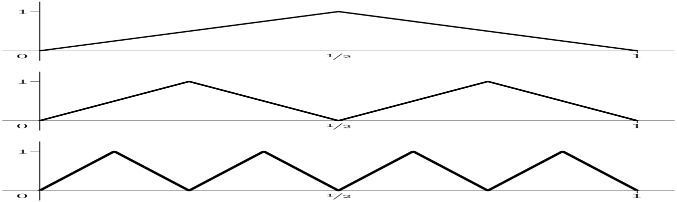


Figure 2: f_m, f_m^2, and f_m^3.

In contrast, a two-layer network with the same $\phi(x)$ of the form $\phi\left(\sum_{j=1}^{m}\alpha_j\phi(w_jx-b_j)\right)$ requires $m=2^k$ to exactly express $f^k(x)$.

The deep network can be thought of as having $6k$ parameters, whereas the two-layer network requires $3\cdot 2^k+1$ parameters; exponentially more. `https://arxiv.org/abs/1509.08101`

Representational benefits of depth (Telgarsky 15')

Classification error rates

Define the function class $F(\phi; m, \ell)$ be the space of functions composed of ℓ layer fully connected m width feed forward nets with nonlinear activation function ϕ. Let $\mathcal{R}(f) := n^{-1} \sum_{i=1}^{n} \chi[f(x_i) \neq y_i]$ count the number of incorrect labels of the data set $\{(x_i, y_i)\}_{i=1}^{n}$.

Theorem (Telgarsky 15')

Consider positive integers k, ℓ, m with $m \leq 2^{(k-3)/\ell-1}$, then there exists a collection of $n = 2^k$ points $\{(x_i, y_i)\}_{i=1}^{n}$ with $x_i \in [0, 1]$ and $y_i \in \{0, 1\}$ such that

$$\min_{f \in F(\phi;2,2k)} \mathcal{R}(f) = 0 \quad \text{and} \quad \min_{g \in F(\phi;m,\ell)} \mathcal{R}(g) \geq \frac{1}{6}.$$

https://arxiv.org/abs/1509.08101

Representational benefits of depth (Yarotsky 16')

ReLU nets can approximate x^2 exponentially well

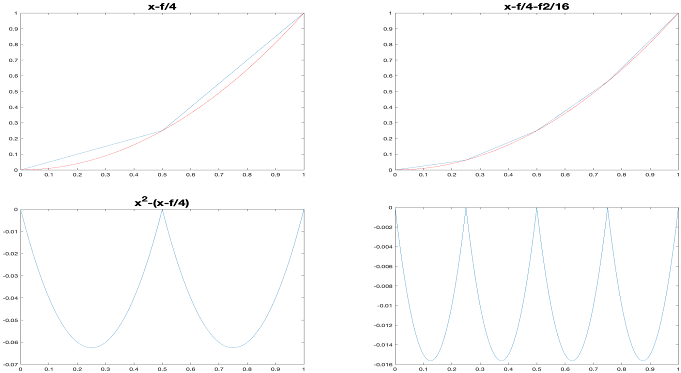
Returning to the saw-tooth function composted of $\phi(x) = \max(x, 0)$ let $f(x) = h_3(x) = \phi(2\phi(x) - 4\phi(x - 1/2))$ and iterate this 2-layer network m times to obtain a $2m$-layer network $f^m(x) = f(f(\cdots(f(x)\cdots))$ with $6m$ weights.
Let $h_m(x)$ denote the piecewise linear interpolation of $h(x) = x^2$ at 2^{m+1} equispaced points, then

$$h_m(x) = x - \sum_{s=1}^{m} 2^{-2s} f^s(x)$$

and $\max_{x\in[0,1]}|x^2 - h_m(x)| = 2^{-2(m+1)}$. Consequently, x^2 can be approximated on $[0,1]$ to uniform accuracy ϵ by a ReLU network having depth $\frac{1}{2}\log_2(1/\epsilon)$ and 6 weights per pair of layers.
https://arxiv.org/pdf/1610.01145.pdf

Representational benefits of depth (Yarotsky 16')

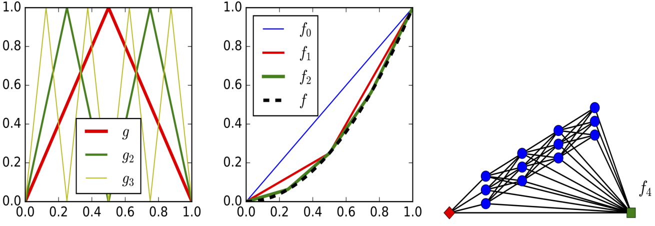
ReLU nets can approximate x^2 exponentially well: plots 1



Yarotsky (16') approximation of x^2 with ReLU DNN.
`https://arxiv.org/pdf/1610.01145.pdf`

Representational benefits of depth (Yarotsky 16')

ReLU nets can approximate x^2 exponentially well: plots 2



(a) (b) (c)

Figure 2: Fast approximation of the function $f(x) = x^2$ from Proposition 2: (a) the "tooth" function g and the iterated "sawtooth" functions g_2, g_3; (b) the approximating functions f_m; (c) the network architecture for f_4.

Telgarsky (15') and Yarotsky (16') follow from exponential nature of composition of the same function, self similarity.
`https://arxiv.org/pdf/1610.01145.pdf`

Representational benefits of depth (Yarotsky 16')

ReLU nets can approximate x^2 exponentially well: plots

High order approximation can be shown by extending that a DNN with depth and number of weights proportional to $\ln(1/\epsilon)$ can approximate any quadratic function within ϵ to polynomials of arbitrary degree. This follows by noting the relationship

$$xy = \frac{1}{2}\left((x+y)^2 - x^2 - y^2\right)$$

which demonstrates that the ability to square a number allows general multiplication. For example, letting $H(x;\theta)$ denote a network approximating x^2, then the above relation can be applied to compute $x^3 = xH(x)$ by letting $y = H(x)$. Similarly polynomials of arbitrary degree can be approximated within ϵ by a DNN with depth and number of weights proportional to $\ln(1/\epsilon)$.
https://arxiv.org/pdf/1610.01145.pdf

Representational benefits of depth (Yarotsky 16')

ReLU nets can approximate Sobolev spaces

The Sobolev norm is similar to that of functions with $n-1$ derivatives that are Lipschitz continuous $C^{n-1}([0,1]^d)$ excluding sets of measure zero.

$$\|f\|_{W^{n,\infty}}([0,1]^d) = \max_{|s|\leq n} \text{esssupp}_{x\in[0,1]^d} |D^s f(x)|.$$

Define the unit ball of functions in $W^{n,\infty}([0,1]^d)$ as

$$F_{n,d} = \left\{ f \in W^{n,\infty}([0,1]^d) : \|f\|_{W^{n,\infty}}([0,1]^d) \leq 1 \right\}.$$

Theorem (Yarotsky 16')

For any d, n and $\epsilon \in (0,1)$, there is a ReLU network with depth at most $c(1+\ln(1/\epsilon))$ and at most $c\epsilon^{-d/n}(1+\log(1/\epsilon))$ weights (width $\mathcal{O}(\epsilon^{-d/n})$), for c a function of d, n, that can approximate any function from $F_{d,n}$ within absolute error ϵ.

Representational benefits of depth (Yarotsky 16')

Sketch of the proof 1 of 3: localization

Localize an arbitrary function in $\mathbb{R}^d$ into $(N+1)^d$ local continuous regions using local (compactly supported) functions $\phi_m(x)$ which sum to 1. E.g. let

$$\text{with } \psi(x) = \begin{cases} 1 & |x| < 1 \\ 2-|x| & 1 \leq |x| \leq 2 \\ 0 & |x| > 2 \end{cases}$$

and note that $\sum_{m=0}^{N} \psi\left(3N(x_k - m/N)\right) = 1$ for $x_k \in [0,1]$. Multiplying $f(\cdot)$ by each shift $\psi\left(3N(x_k - m/N)\right)$ for $m = 0, \cdots, N$ localizes the x_k variable over and can be done via a one-dimensional convolutional layer with one filter that doesn't require trainable parameters. This can then be repeated over d times to localize each of the d variables into $(N+1)^d$ partitions.

Representational benefits of depth (Yarotsky 16')

Sketch of the proof 2 of 3

Taylor series of $f(\cdot)$ about $\{(x_k - m/N)\}_{m=0}^{N}$ to degree n in each dimension x_k $k = 1, \ldots, d$ is

$$P_{k,n}(f)(x) := \sum_{s=0}^{n} \frac{\partial^s f(x)}{s! \partial x_k} (x_k - m/N)^s$$

and the composite over all dimensions is $P_n(f)(x) := \Pi_{k=1}^{d} P_{k,n}(f)(x)$.
The resulting error approximating $f(x)$ about $\{(x_k - m/N)\}_{m=0}^{N}$ is bounded by at most 2^d local terms (as any location x interacts with at most 2 local dilated $\phi(3N(x_k - m/N)$ with each term bounded using the standard Taylor series truncation bound

$$\frac{d^n}{n! N^n} \max_{|s| \leq n} \text{esssupp}_{x \in [0,1]^d} |D^s f(x)|.$$

Representational benefits of depth (Yarotsky 16')

Sketch of the proof 3 of 3: combining terms

Treating $\|f\|_{W^{n,\infty}}([0,1]^d) := \max_{|s|\leq n}\text{esssupp}_{x\in[0,1]^d}|D^s f(x)|$ as bounded independent of n (not really true) gives a total bound on the local error of $2^d d^n/N^n n!$ which is bounded by ϵ if $N \geq \left(n!\epsilon/2^d d^n\right)^{-1/n}$.

It then remain to construct a network that can approximate the local Taylor series with the claimed width and depth. The partition has $d^n(N+1)^d$ terms of the form $\phi_m(x)(x-m/N)^n$, each of which can be approximated efficiently using the aforementioned ReLU networks using order $\log(2^d d^n/\epsilon)$ depth for a total of $d^n(N+1)^d\log(2^d d^n/\epsilon)$ weights.

Recalling the number of partitions $N \geq \left(n!\epsilon/2^d d^n\right)^{-1/n}$ and Stirling's Inequality that $n! \sim (n/e)^n\sqrt{2\pi n}$, gives the claimed depth and width.

Representational benefits of depth (Yarotsky 16')

Near optimality

- Yarotsky's result shows a neural network with ReLU activation can approximate any n-smooth function in $d-$dimensions using at most order $\epsilon^{-d/n}(1+\log(1/\epsilon))$ trainable parameters. https://arxiv.org/pdf/1610.01145.pdf
- DeVore et al. proved the minimal number of trainable parameters for any method is of order $\epsilon^{-d/n}$, https://link.springer.com/article/10.1007/BF01171759
- Boulle et al. consider nonlinear activations that are rational functions of the form of a cubic over a quadratic, giving $\epsilon^{-d/n}(1+\log\log((1/\epsilon)))$ parameters. https://arxiv.org/abs/2004.01902
- Goverse et al. used two nonlinear activations (e.g. ReLU and x^2) to achieve optimal order $\epsilon^{-d/n}$ https://arxiv.org/abs/2301.13091

Optimal function approximation ability of deep networks

DNNs can achieve optimal rates for function classes

There is a growing literature on the ability to express high dimensional data using deep networks, to name a few:

- Approximation space for univariate functions; Daubechies, DeVore, Foucart, Hanin, and Petrova (19') https://arxiv.org/pdf/1905.02199.pdf
- That neural networks achieve the same approximation rate as methods such as wavelets, ridgelets, curvelets, shearlets, $\alpha-$molecules; Bölcskei, Grohs, Kutyniok, and Petersen (18') https://www.mins.ee.ethz.ch/pubs/files/deep-approx-18.pdf

The exponential complexity generated by depth allows these remarkable approximation rates. Note however, one needs to be able to train the network parameters to achieve these rates.