What are the ingredients making Deep Learning work:

Data:

- real data lives on a lower dimensional manifold,
- much of the variation we observe are invariance which should be in the net null space; e.g. translation, rotation, dilation,
- real data is compressible, what is the role of sparsity,

Architecture:

- how does the data inform the type of architecture to be used,
- what are the ingredients of modern deep nets, and why,
- to what, if any degree, are networks robust to adversaries,

Ability to train:

- how to train large numbers of networks parameters, θ,
- methods to train them must be scalable and are not run to global convergence,
- what are the impacts of choices in the optimisation method, such as batch size as an implicit regulariser.

Theories for Deep Learning:

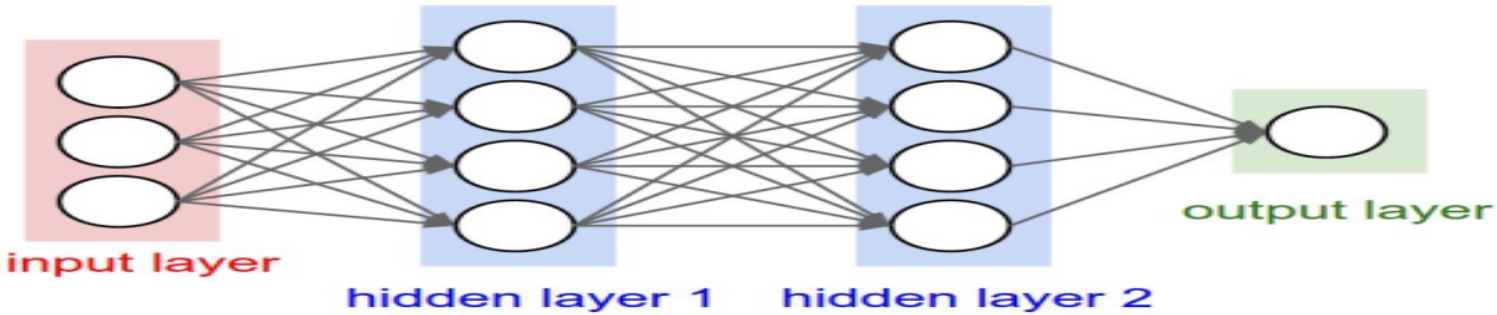
- Expressivity of deep networks and the depth vs. width tradeoff?
- Convergence properties of stochastic gradient descent and other alternatives?
- What can we say about a single layer?
- Are there models of data for which we can guarantee activations are correct?
- What is the topology of the net parameters, θ?
- Are there variants of deep networks that don't require learning, easier to analyse?
- Random matrix theory as a model if few network weights are changed.

Example of a Feedforward network / multilayer perceptron

A feedforward net is a nonlinear function composed of repeated affine transformation followed by a nonlinear action:

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

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



This Feedforward is an example of a network "architecture."
https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Architecture/feedforward.html

Deep learning requires a deep net, loss function, and data

The network "Weights" $W^{(i)}$ (and shifts $b^{(i)}$) are learnt to fit a task for a particular data set; the collection of all network weights are normally summarised as a variable θ.

A "labeled" data set is a collection of input, $x(j) = h_1(j)$, and desired output $y(j)$, pairs $\{(x(j), y(j))\}_{j=1}^{m}$.

The net is trained by minimising a loss function $L(x(i), y(i); \theta)$ summed over all data pairs; that is

$$\min_{\theta} \sum_{i=1}^{m} L(x(i), y(i); \theta).$$

and the resulting learned net is, $H(\cdot; \theta)$.

Outline for today: views on expressivity of feedforward nets

- Network architectures are able to approximate any function (Cybenko (89') and Hornik (90')).
- There are functions for which deep networks are able to construct with polynomially many parameters that require exponentially many parameters for a shallow network to approximate. (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')
- Many recent extensions to give approximation rates, such as work by Bölcskei et al. (18'), Devore et al. (19') and many more...

Expressivity of deep net: what functions can it approximate

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 $x \in \mathbb{R}^n$. In particular:

- What functions can a deep net approximate arbitrarily well?
- What is the advantage of depth?

Superposition of sigmoidal functions (Cybenko 89[1]): Pt. 1

Consider the feedforward network with one hidden layer:

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

Theorem (Cybenbko 89')

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

[1] https://link.springer.com/article/10.1007/BF02551274

Superposition of sigmoidal functions (Cybenko 89[2]): Pt. 2

Theorem (Cybenbko 89')

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

Proof: Let S be the set of functions that can be expressed by $H(x;\theta)=\sum_{i=1}^{m}\alpha_i\sigma(w_i^Tx+b_i)$; our aim is to show S is the space $C_n[0,1]$ (n-dimensional continuous functions in $[0,1]^n$). Assume S is not all of $C_n[0,1]$ and let $\bar{S}$ be its closure. Hahn-Banach Theorem and Riesz Representation Theorem respectively tell us that there is a bounded linear functional $L\neq 0$ such that $L(S)=L(\bar{S})=0$ and L is of the form $L(h)=\int_{I_n}h(x)d\mu(x)$ for some measure $\mu(x)$ and for all $h\in C_n([0,1])$. In particular $\sigma(t)\in\bar{S}$ so $L(\sigma)=0$, but this implies $\mu(x)=0$ for our choice of $\sigma(\cdot)$ which implies $S=\bar{S}$; that is $H(x;\theta)$ is dense in $C_n([0,1])$.

[2] https://link.springer.com/article/10.1007/BF02551274

Approximation of multilayer feedforward nets (Hornik 90[3])

Consider the feedforward network with one hidden layer:

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

Theorem (Hornik 90')

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

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

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

Representational benefits of depth (Telgarsky 15[4]) Pt. 1

The results of Cybenko and Hornik show that the network structure with nonlinear activations is sufficient to generate any function for our tasks; however, they do not provide function approximation rates.

Telegarsky (2015) considered a specific construction of a function from a deep network which requires an shallow network of exponential depth.
Let $\sigma(x) = ReLu(x) = max(x, 0)$ and consider the two layer net:

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

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

[4] https://arxiv.org/abs/1509.08101

Representational benefits of depth (Telgarsky 15[5]) Pt. 2

For $\sigma(x) = \max(x, 0)$ let $f(x) = h_3(x) = \sigma(2\sigma(x) - 4\sigma(x - 1/2))$ 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.

In contrast, a two-layer network with the same $\sigma(x)$ of the form $\sigma\left(\sum_{j=1}^{m} \alpha_j \sigma(w_j x - b_j)\right)$ 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.

[5] https://arxiv.org/abs/1509.08101

Representational benefits of depth (Telgarsky 15[6]) Pt. 3

A consequence of the aforementioned construction of $f_k(x)$ by Telgarsky can be quantified in terms of classification rates as functions of depth and width.
Define the function class $F(\sigma; m, \ell)$ be the space of functions composed of ℓ layer fully connected m width feed forward nets with nonlinear activation function σ. 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

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

[6] https://arxiv.org/abs/1509.08101

Representational benefits of depth (Yarotsky 16[7]) Pt. 1

Returning to the saw-tooth function composted of $\sigma(x) = \max(x, 0)$ let $f(x) = h_3(x) = \sigma(2\sigma(x) - 4\sigma(x - 1/2))$ and iterate this 2-layer network k times to obtain a $2k$-layer network $f_k(x) = f(f(\cdots(f(x)\cdots))$.
Let $h_m(x)$ denote the piecewise linear interpolation of $h(x) = x^2$ at 2^{m+1} equispaced points, then the difference when increasing m by one is

$$h_{m-1}(x) - h_m(x) = 2^{-2m} f_m(x)$$

and consequently $h_m(x) = x - \sum_{s=1}^{m} 2^{-2s} f_s(x)$.
Consequently, $h(x) = x^2$ can be approximated on $[0, 1]$ to uniform accuracy ϵ by a ReLU network having depth and number of weights proportional to $\ln(1/\epsilon)$.

[7] https://arxiv.org/pdf/1610.01145.pdf

Representational benefits of depth (Yarotsky 16[8]) Pt. 1

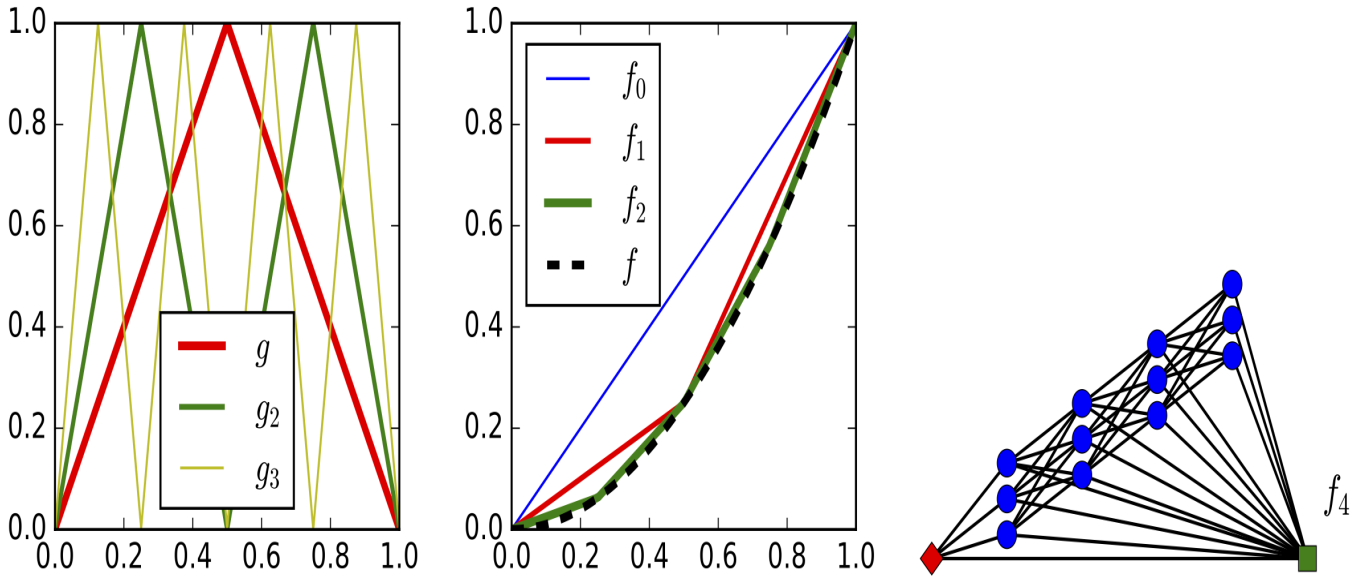


Figure 2: Fast approximation of the function $f(x) = x^2$ from Proposition 2: (a) the "tooth" function g and the iterated "sawtooth" functions g_2, g_3; (b) the approximating functions f_m; (c) the network architecture for f_4.

Composition of the same function, self-similarity, is the key property used to show the results of Telgarsky (15') and Yarotsky (16').

[8] https://arxiv.org/pdf/1610.01145.pdf

Representational benefits of depth (Yarotsky 16[9]) Pt. 2

Yarotsky used the ability to approximate x^2 as the foundation to approximate functions in Sobolev spaces. The Sobolev norm

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

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. 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 $c\epsilon^{-d/n}(1+\log(1/\epsilon))$, for c a function of d, n, that can approximate any function from $F_{d,n}$ within absolute error ϵ.

[9] https://arxiv.org/pdf/1610.01145.pdf

Function approximation ability of deep networks: ongoing

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')[10]
- That neural networks achieve the same approximation rate as methods such as wavelets, ridgelets, curvelets, shearlets, $\alpha-$molecules; Bölcskei, Grohs, Kutyniok, and Petersen (18')[11]

While these results show that deep networks can achieve remarkable approximation rates for a diverse set of data, and that in particular depth is key to their fast approximation rates. Note however, one needs to be able to train the network parameters to achieve these rates and avoid overfitting, etc...

[10] https://arxiv.org/pdf/1905.02199.pdf

[11] https://www.mins.ee.ethz.ch/pubs/files/deep-approx-18.pdf

Summary on expressivity of deep feedforward networks

- Cybenko (89') showed that a single layer network with sigmoidal nonlinear activation can approximate any continuous function with arbitrary accuracy.
- Hornik (90') extended Cybenko's results to a much broader class of nonlinear activations, including ReLu.
- Telgarsky (15') used a specific deep network to construct a function and associated classification task to show that there are functions for which deep networks can exactly classify the data using polynomially many parameters (weights and bias), while exact reconstruction with a shallow network would require exponentially many parameters, otherwise has an $\mathcal{O}(1)$ fraction classification error.
- Yarotsky (17') showed ϵ approximation of smooth functions, such as x^2, on bounded domains with width being proportional to $\ln(1/\epsilon)$; extended to Sobolev spaces in $\mathbb{R}^d$.
- Bölcskei et al. (18'), Devore et al. (19') and many more...