Neural Network Complexity

Approximation Rates: Shallow vs. Deep

Outline

The Fundamental Question

The Goal

Given a regular function $f : \mathbb{R}^d \to \mathbb{R}$ (e.g., Lipschitz, Sobolev) and a desired accuracy $\epsilon > 0$, we want to find a Neural Network $\hat{f}$ such that:

$$\|f(x) - \hat{f}(x)\| \leq \epsilon$$

Key Questions:

- Existence: can we always do this? ✓
- Complexity: how many resources do we need ?
- Is a **Shallow** (wide) network better, or a **Deep** (narrow) one ?

Universal Approximation Theorem (UAT)

Theorem (Cybenko 1989, Hornik 1991)

A feedforward network with a **single hidden layer** containing a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$, under mild assumptions on the activation function σ.

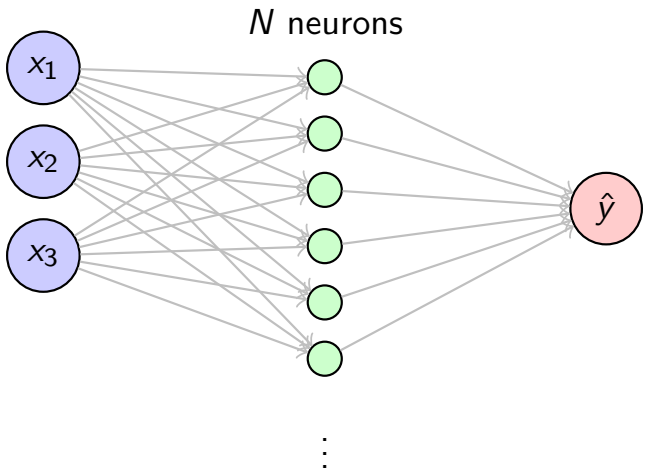
Meaning: The theorem guarantees *existence*, but says nothing about *efficiency*.

- The required number of neurons (width W) might be astronomically large.
- Sometimes W grows exponentially with the input dimension d ("Curse of Dimensionality").

Shallow Networks: The Architecture

A shallow network has 1 hidden layer of width N.

$$\hat{f}(x) = \sum_{i=1}^{N} a_i \sigma(\mathbf{w}_i^T x + b_i)$$



It acts as a linear combination of basis functions.

Approximation Rate: Shallow

Theorem

Let σ be infinitely differentiable and not a polynomial then for a function f with smoothness r (derivatives up to order r are bounded), the approximation error behaves roughly like:

$$\boxed{\|f - \hat{f}_N\| \approx O\left(N^{-\frac{r}{d}}\right)}$$

where

- N: Number of neurons;
- d: Input dimension;
- r: Regularity of the function;
- $\hat{f}_N$: Neural network approximation.

Implication: To get error ϵ, we need: $N_s \approx \epsilon^{-\frac{d}{r}}$

If d is large (e.g., images), N becomes exponentially large. This is the bottleneck of shallow networks.

Approximation Rate: Deep (The Advantage)

Why do Deep Networks work better for high-dimensional data (e.g., images)?

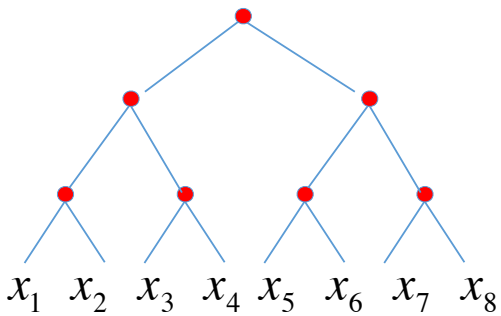
Assumption: Real-world functions are often **compositional** (hierarchical).

$$f(x) = h_L \circ h_{L-1} \circ \cdots \circ h_1(x)$$

where each h_i depends on only a few variables (dimension $\tilde{d} \ll d$).

Example:

$f(x_1, ..., x_8) = h_3(h_{21}(h_{11}(x_1, x_2), h_{12}(x_3, x_4)), h_{22}(h_{13}(x_5, x_6), h_{14}(x_7, x_8)))$



Approximation Rate: Deep (compositional)

Theorem (e.g., Poggio et al., 2017)

Let σ be infinitely differentiable and not a polynomial then for compositional functions, a Deep Network achieves error ϵ with size:

$$N_d \approx O\left((d-1)\cdot \epsilon^{-\frac{\tilde{d}}{r}}\right)$$

Remark

It is possible to prove that every function f can be approximated by and ϵ-close binary function f_B (binarization).

Remark

Both results assume σ infinitely differentiable. ReLU does not satisfy this hypothesis but it is possible to smooth the ReLU function in arbitrarily small interval around the origin.

ReLU (non-smooth) functions

Theorem

Let f be a L-Lipshitz continuous function of d variables. Then, the complexity of a network which is a linear combination of ReLU providing an approximation with accuracy at least ϵ is

$$N_s = O\left(\left(\frac{\epsilon}{L}\right)^{-d}\right)$$

whereas that of a deep binary compositional architecture is

$$N_d = O\left((d-1)\left(\frac{\epsilon}{L}\right)^{-2}\right)$$

Comparison (smooth σ)

Shallow vs Deep

- **Shallow:** $N_s \approx \epsilon^{-d/r}$ (Exponential in d)
- **Deep:** $N_d \approx d \cdot \epsilon^{-\tilde{d}/r}$ (**Linear** in d)

The dimension d becomes a **multiplicative factor**, curing the curse of dimensionality.

Example

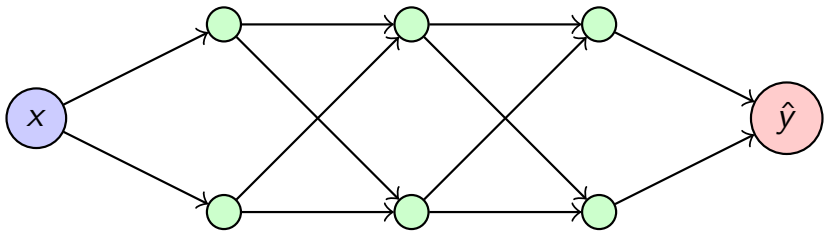
Suppose $d = 1e3, r = 10, \tilde{d} = 2$ and $\epsilon = 1e-2$, then we have:

- **Shallow case:** $N_s \approx 10^{200}$
- **Deep case (compositional):** $N_d \approx 2500$

Deep Networks: Composition

Deep networks create function compositions:

$$\hat{f}(x) = f_L \circ f_{L-1} \circ \cdots \circ f_1(x)$$



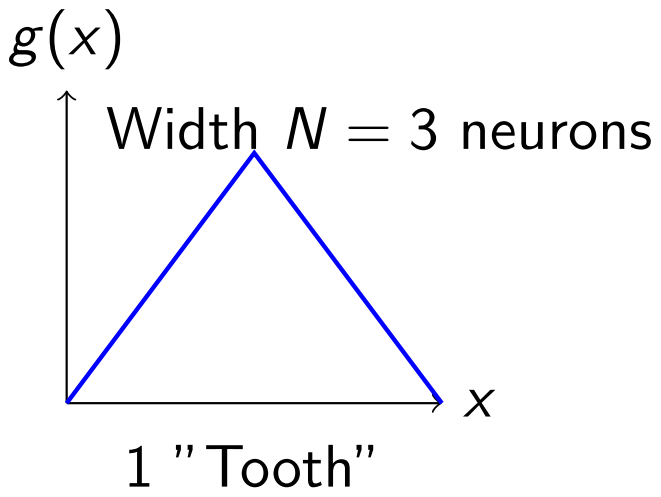
Depth L, Width W

Depth allows the network to **recycle** computations and create high-frequency features through "folding."

The "Sawtooth" Argument: Intuition

How can we measure "Expressive Power"? Look at the number of linear regions (oscillations) a network can create.
Consider the "Hat Function" (or Triangle Wave) $g : [0, 1] \to [0, 1]$:

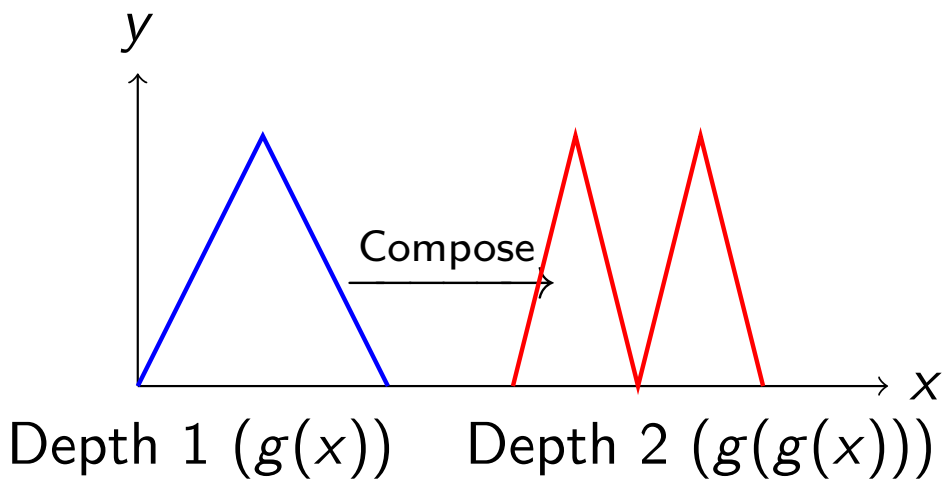
$$g(x) = \begin{cases} 2x & 0 \leq x < 1/2 \\ 2(1 - x) & 1/2 \leq x \leq 1 \end{cases}$$



This function "folds" the domain $[0, 1]$ in half.

The "Sawtooth" Argument: Effect of Depth

What happens if we compose this function with itself? $g(g(x))$?

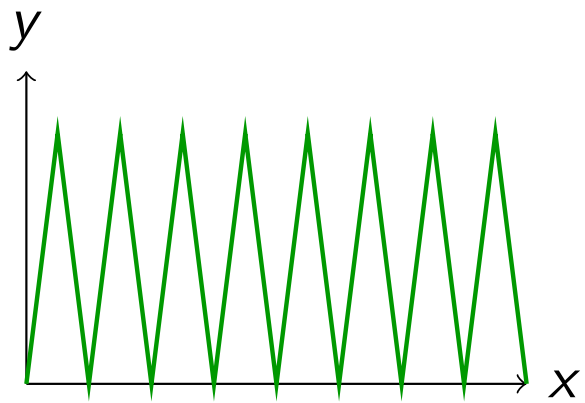


- Depth 1: 1 Peak (2^0)
- Depth 2: 2 Peaks (2^1)

Each composition doubles the number of linear regions!

The "Sawtooth" Argument: Exponential Growth

Adding just one more layer ($g(g(g(x)))$) doubles the frequency again.



Depth 4 $\implies$ $2^{4-1} = 8$ Peaks

Telgarsky's Result (2016)

To approximate this function with a **Shallow Network**, you would need $O(2^k)$ neurons (one per peak).
A **Deep Network** needs only $O(k)$ neurons (3 per layer).

Complexity Comparison

Feature	**Shallow Network**	**Deep Network**
Structure	Parallel (Wide)	Serial (Deep)
Operation	Linear Combination	Composition
High Frequency	Expensive ($O(N)$)	Efficient ($O(\log N)$)
Symmetries	Hard to capture	Easy (e.g., CNNs)
Optimization	Convex (if layer fixed)	Highly Non-Convex

Table: Trade-offs in Approximation Power

Takeaways

1. **Universality:** Both shallow and deep networks are universal approximators. Existence is not the issue.
2. **Efficiency:** Deep networks are exponentially more efficient at representing certain classes of functions (compositional, high-frequency, symmetrical).
3. **The Trade-off:**
   - Shallow: Easier optimization theory, but requires massive width for complex data.
   - Deep: High expressivity with fewer parameters, but harder to train (vanishing gradients, landscape issues).