Why use Deep Neural Networks?

Nikolaj Takata Mücke (nikolaj.mucke@CWI.nl)

19.2.2019

Overview

Who is Nikolaj?

Introduction

Degree of Approximation

Compositional Functions

Shallow Networks

Deep Networks

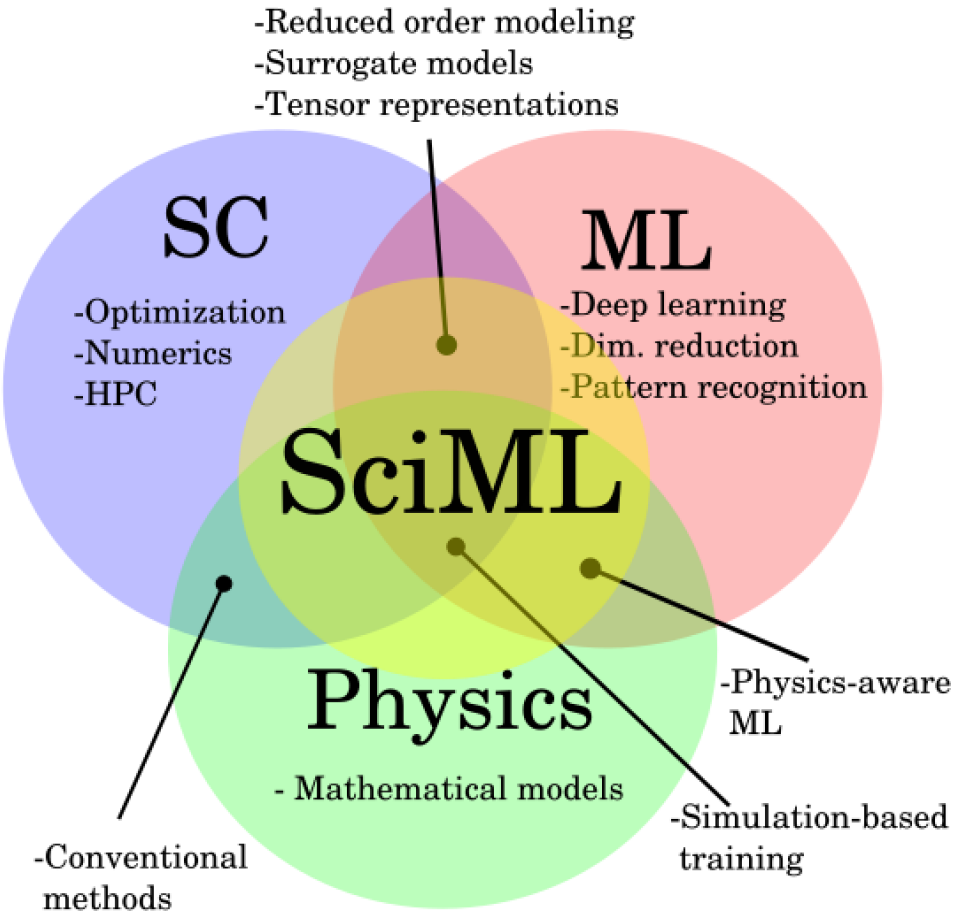
Who is Nikolaj?

Who is Nikolaj?

- Danish
- B.Sc. and M.Sc in Applied/Computational Mathematics from the Technical University of Denmark (DTU)
- PhD student at CWI
  - Started September 2019
  - Scientific Computing Group
  - TU Delft
  - Supervisors: Kees Oosterlee and Sander Bothe

Research Interests

- PDEs
- Scientific Computing
- Reduced Order Modeling
- Physics-Informed Machine Learning
- Scientific Machine Learning



And now... NEURAL NETWORKS!!

IT’S TIME TO GO DEEP!!

Introduction

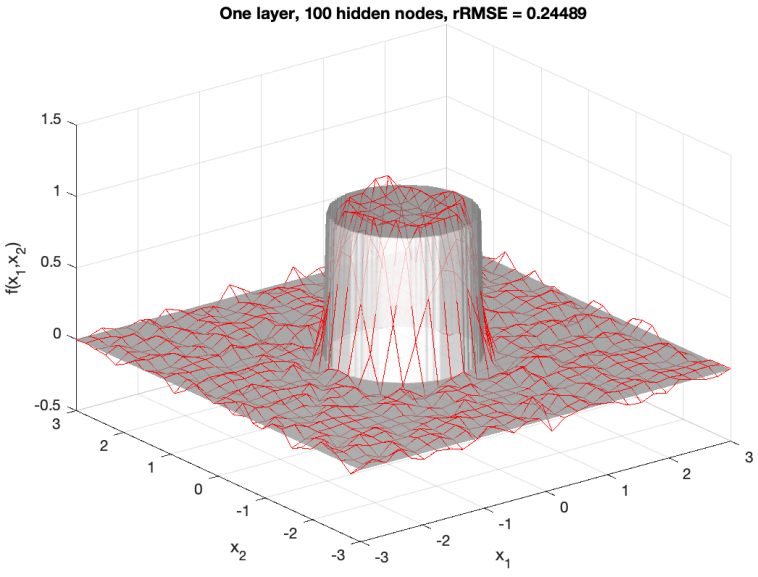
Deep vs. Shallow

- When does it make sense to go deep?
- Why does it make sense to go deep?

References

- Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., & Liao, Q. (2017). Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. International Journal of Automation and Computing, 14(5), 503-519.
- Mhaskar, H., Liao, Q., & Poggio, T. (2017, February). When and why are deep networks better than shallow ones?. In Thirty-First AAAI Conference on Artificial Intelligence.
- Mhaskar, H. N., & Poggio, T. (2016). Deep vs. shallow networks: An approximation theory perspective. Analysis and Applications, 14(06), 829-848.

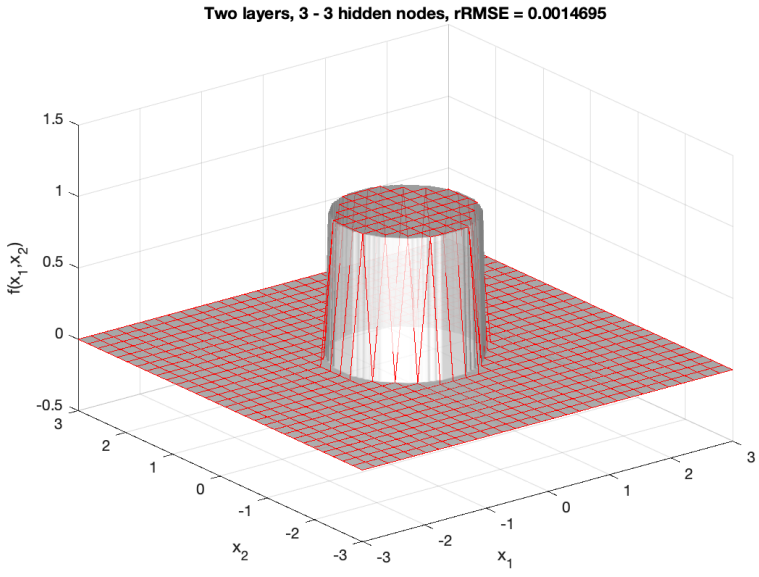
Introduction



RMSE = 0.2449

1 hidden layer with 302 neurons

Total number of neurons: 302



RMSE = 0.0014

3 hidden layers with 3 neurons

Total number of neurons: 25

Degree of Approximation

Degree of Approximation

- **Complexity**: Number of neurons/units in a network
- Let V_N be the set of networks with complexity N
- We assume $V_N \subseteq V_{N+1}$
- **Degree of Approximation** is defined by

$$\text{dist}(f, V_N) = \inf_{P \in V_N} ||f - P||_X,$$

  for $f \in X$
- If

$$\text{dist}(f, V_N) = \mathcal{O}(N^{-\gamma})$$

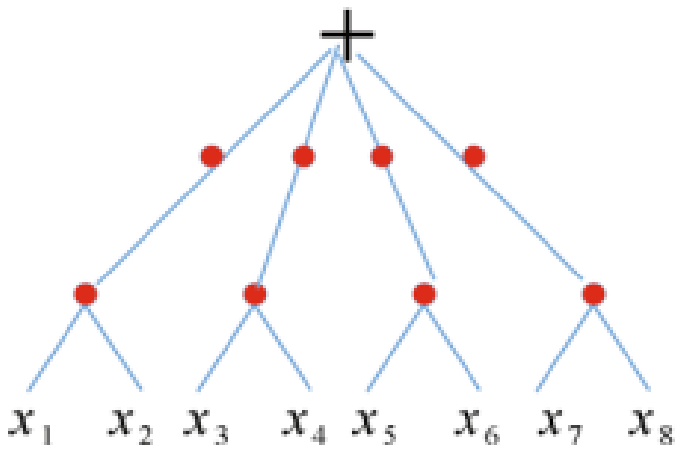
  then a network with complexity $N = \mathcal{O}(\epsilon^{-\frac{1}{\gamma}})$ is sufficient to guarantee accuracy at least ϵ

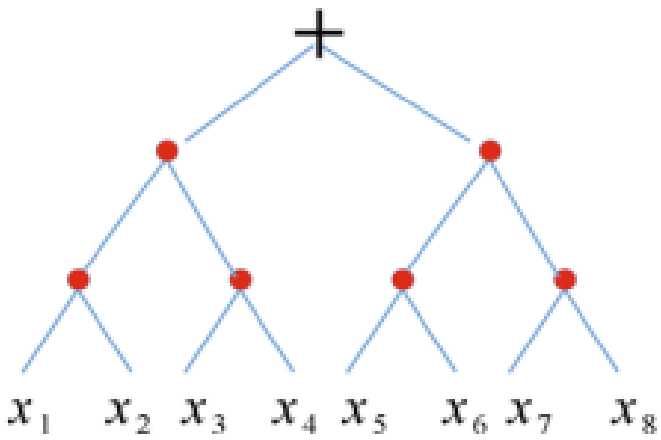
Compositional Functions

Compositional Functions

We will consider **hierarchical compositions** of functions, such as

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





Compositional Functions - Function Class

- Let $I^n = [-1, 1]^n$ and $C(I^n)$ be the space of continuous functions on I^n with norm

$$||f|| = \max_{x \in I^n} |f(x)|$$

- Then consider the subspace $W_r^n \subset C(I^n)$ consisting of r times continuously differentiable functions on I^n, where

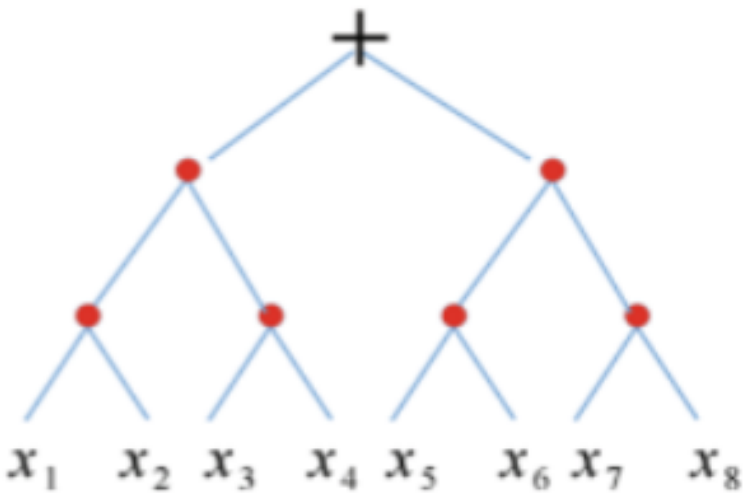
$$||f|| + \sum_{1 \leq |k|_1 \leq r} ||D^k f|| \leq 1,$$

where k is a multiindex and D^k denotes the partial derivative indicated by k, and $|k|_1$ is the sum of the multiindex

Compositional Functions - Function Class

- Let $W_r^{n,2}$ be the class of functions with a hierarchical structure with each constituent function, $h_{ij} \in W_r^2$
  - Binary tree functions
- Note $W_r^{n,2} \subset W_r^n$
- Example of a $W_r^{4,2}$ function:

$$f(x_1, x_2, x_3, x_4) = \underbrace{h_2}_{\in W_r^2}\Big(\underbrace{h_{11}}_{\in W_r^2}(x_1, x_2), \underbrace{h_{12}}_{\in W_r^2}(x_3, x_4)\Big)$$



Shallow Networks

Shallow Networks

- Shallow network: One hidden layer
- Let $S_{N,n}$ be the class of shallow networks with N neurons and n-dimensional input of the form

$$x \mapsto \sum_{k=1}^{N} a_k \sigma(w_k \cdot x + b_k), \quad w_k \in \mathbb{R}^n, \quad a_k, b_k \in \mathbb{R},$$

- where $\sigma : \mathbb{R} \to \mathbb{R}$ is the (nonlinear) activation function.
- Number of trainable parameters: $(n+2)N \sim N$

Shallow Networks

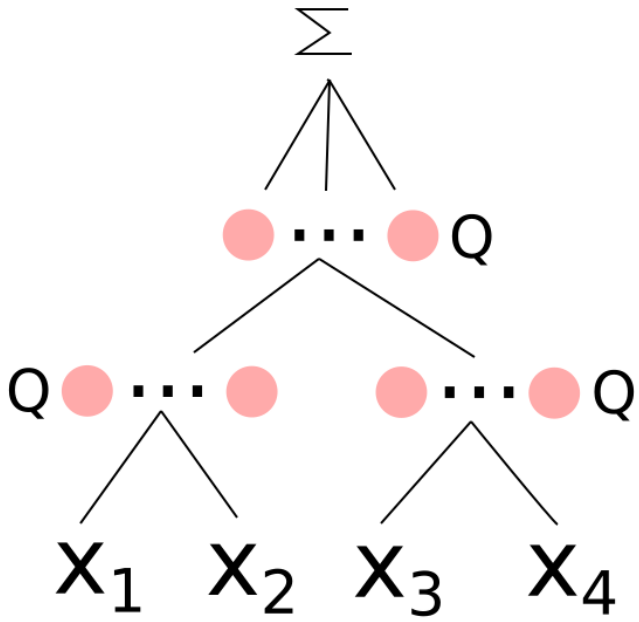
Theorem
Let $\sigma : \mathbb{R} \to \mathbb{R}$ be infinitely differentiable, and not a polynomial on any subinterval of $\mathbb{R}$.
For $f \in W_r^n$ the complexity of shallow networks that provide accuracy at least ϵ is $N = \mathcal{O}(\epsilon^{-n/r})$ and is the best possible.

- The complexity increases **exponentially** with the input dimension - Curse of dimensionality

Deep Networks

Deep Networks

- Deep network: At least two hidden layers
- Let $D_{N,2}$ be the class of Deep networks that take n-dimensional input with N neurons and binary tree structure
- Each constituent function in the deep network is a $S_{Q,2}$ network
- Complexity: $(n-1)Q = N$
- Note: This type of network is a convolutional neural network!



Deep Networks

Theorem
For $f \in W_r^{n,2}$ the complexity of a deep network (with the same compositional achitecture) that provide accuracy at least ϵ is $N = \mathcal{O}((n-1)\epsilon^{-2/r})$ and is the best possible.

- The complexity **does NOT** increase exponentially with the input dimension!

Deep Networks - Proof

- Each constituent function of f can be approximated by a network in $S_{N,2}$ up to accuracy $\epsilon = cN^{-r/2}$.
- Consider the simple case with three constituent function, h, h_1, h_2, being approximated by shallow networks P, P_1, P_2,

$$||h - P|| \leq \epsilon, \quad ||h_1 - P_1|| \leq \epsilon, \quad ||h_2 - P_2|| \leq \epsilon.$$

- Then we have

$$\begin{aligned}
&||h(h_1, h_2) - P(P_1, P_2)|| \\
&\leq ||h(h_1, h_2) - h(P_1, P_2) + h(P_1, P_2) - P(P_1, P_2)|| \\
&\leq ||h(h_1, h_2) - h(P_1, P_2)|| + ||h(P_1, P_2) - P(P_1, P_2)|| \\
&\leq ||h(h_1, h_2) - h(P_1, P_2)|| + c_3\epsilon
\end{aligned}$$

- Since $h \in W_r^2$ it is Lipschitz continuous, which gives us

$$\begin{aligned}
&\leq c_L \left(||h_1 - P_1|| + ||h_2 - P_2||\right) + c_3\epsilon \\
&\leq (2c_L + c_3)\epsilon \\
&= 3c\epsilon
\end{aligned}$$

Deep Networks - Proof

- Conclusively

$$||h(h_1, h_2) - P(P_1, P_2)|| \leq 3c\epsilon$$

- This generalizes easily to deeper hierarchies
- For a function in $W_r^{n,2}$ there are $(n-1)$ nodes in the graph, i.e. $(n-1)$ constituent function to approximate. This gives us

$$||f - NN|| \leq (n-1)c\epsilon = (n-1)cN^{-r/2},$$

Which proves the theorem $\square$

Deep Networks - Example

$$f(x) = (e^{\cos(0.2x^2)} - 1)^2, \quad x \in [-2\pi, 2\pi]$$

Training

- Adam optimizer
- Learning rate: 10^{-3}
- 100000 training samples
- Early stopping and L^2-regularization
- multiple start
- Softplus activation function (smoothed ReLU)

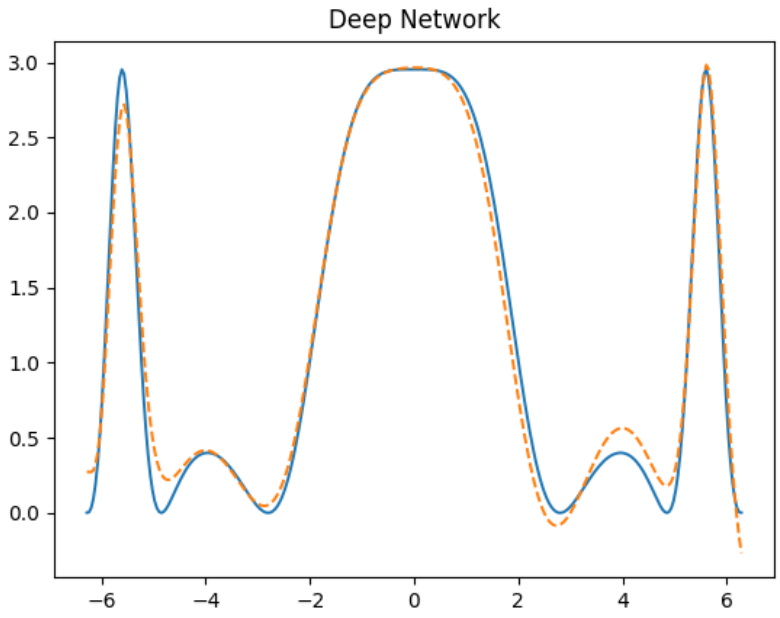
Deep

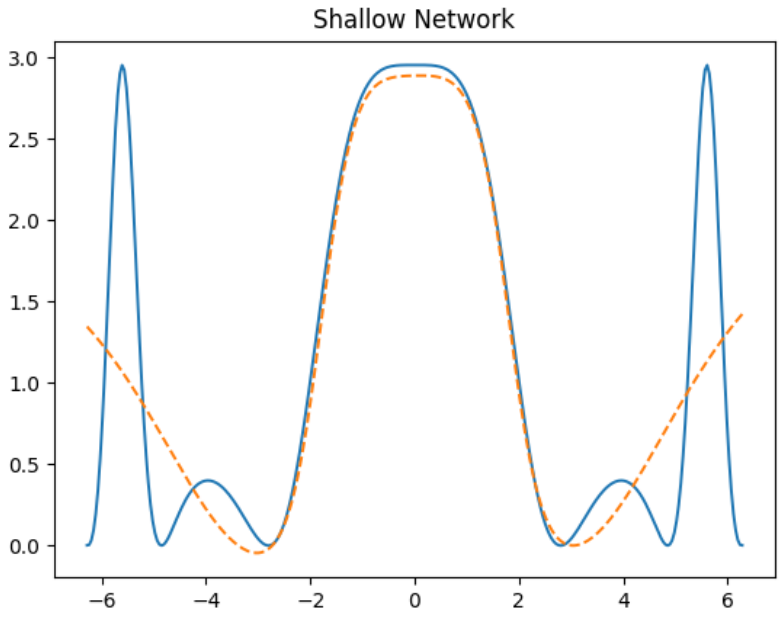
- 4 hidden layers
- 1 to 12 neurons in each layer
- Batch normalization

Shallow

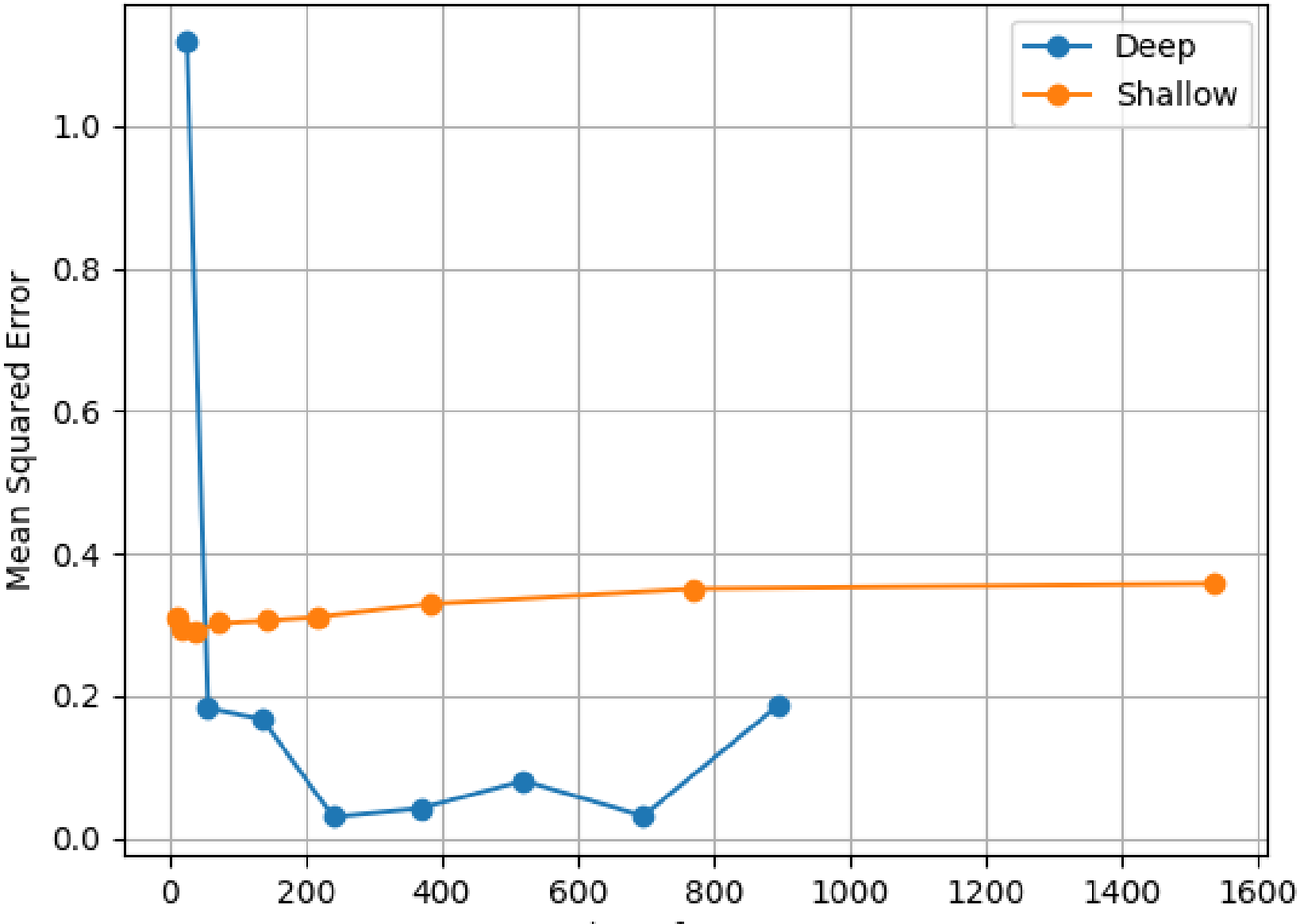
- 3 to 512 neurons in hidden layer

Deep Networks - Example





Deep Networks - Example



Deep Networks - Example

8th degree polynomial in 8 variables (12870 parameters) with a square root

$$f(x_1, \ldots, x_8) = \sqrt{((x_1 + 2x_2)^4 + (x_3 + 2x_4)^4)^2 + ((x_5 + 2x_6)^4 + (x_7 + 2x_8)^4)^2}$$

Training

- Adam optimizer
- Learning rate: 10^{-3}
- 100000 training samples
- Early stopping and L^2-regularization
- multiple start
- Softplus activation function

Deep

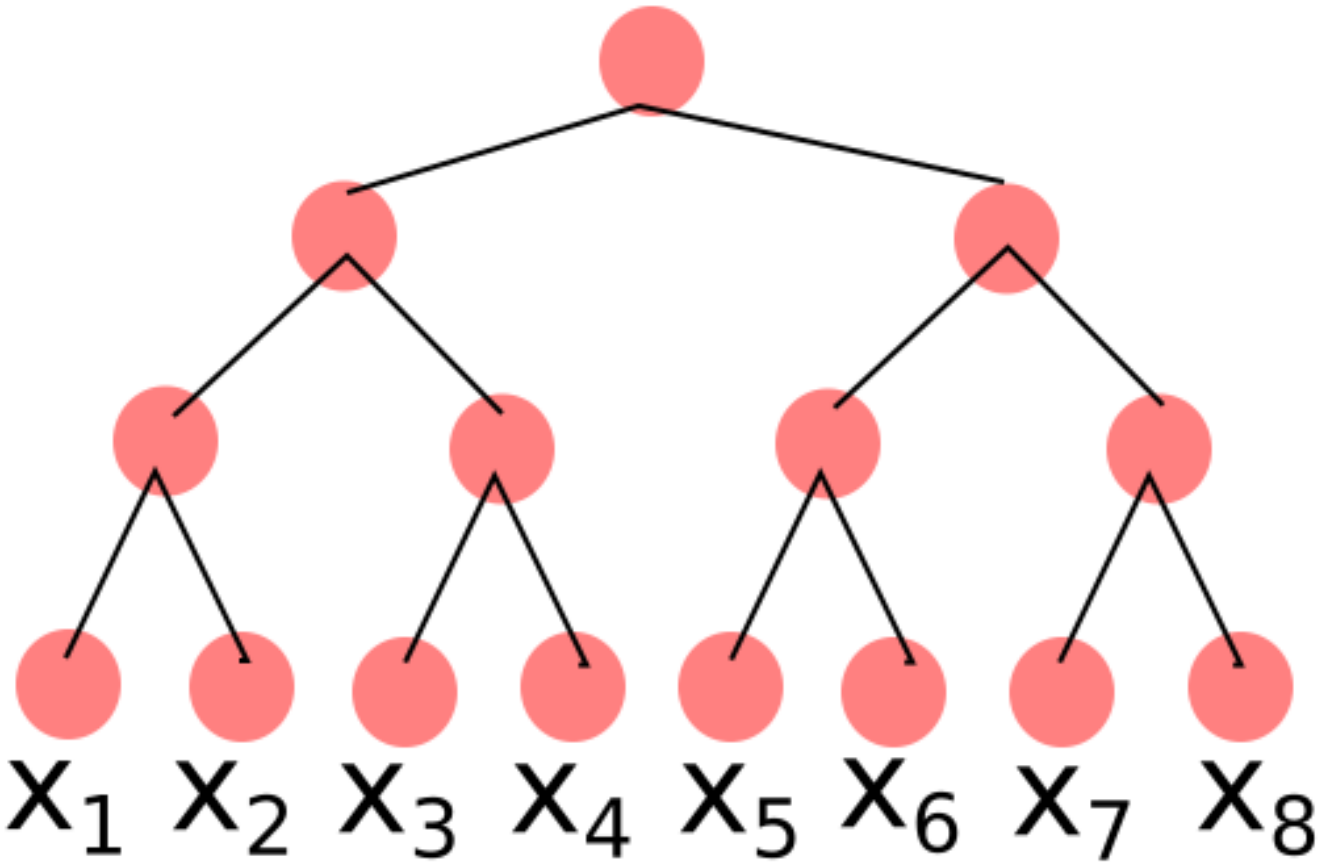
- 3 convolutional layers
- 1 to 14 filters in each layer
- Batch normalization

Shallow

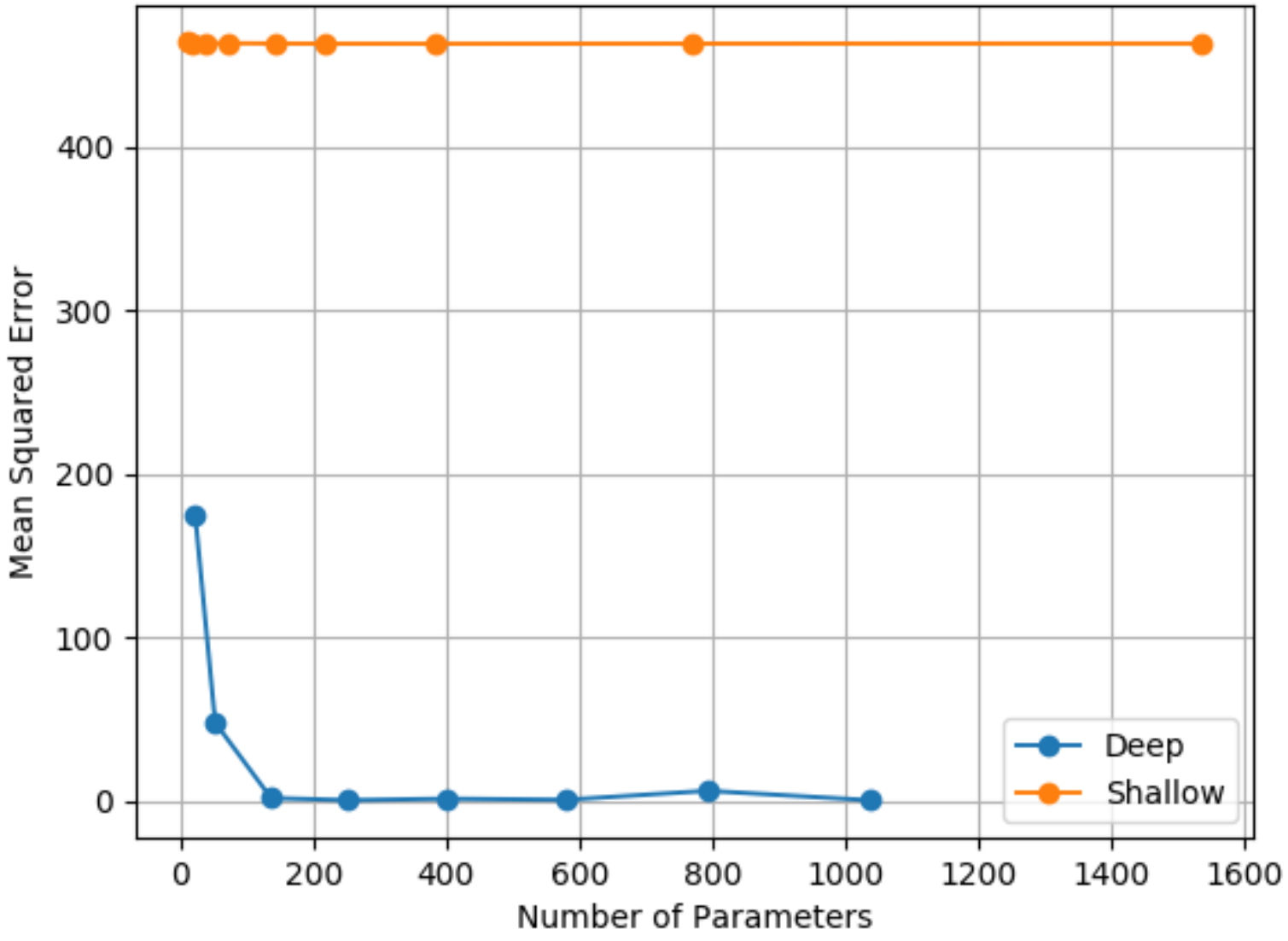
- 3 to 512 neurons in hidden layer

Deep Networks - Example

$$f(x_1, \ldots, x_8) = \sqrt{((x_1 + 2x_2)^4 + (x_3 + 2x_4)^4)^2 + ((x_5 + 2x_6)^4 + (x_7 + 2x_8)^4)^2}$$



Deep Networks - Example



Deep Networks - Example

8th degree polynomial in 8 variables (12870 parameters) with a square root

$$f(x_1, \ldots, x_8) = \text{sqrt}\left(\left((x_1 + 2x_5 + x_7)^4 + (x_8 + 2x_3)^4\right)^2 + \cos(x_4 + 2x_2)^2 + x_6^2 + (x_2 + x_7)^4 + \left(\cos(x_4 + 2x_2) + (x_2 + x_7)^4\right)^2\right)$$

Training

- Adam optimizer
- Learning rate: 10^{-3}
- 100000 training samples
- Early stopping and L^2-regularization
- multiple start
- Softplus activation function

Deep

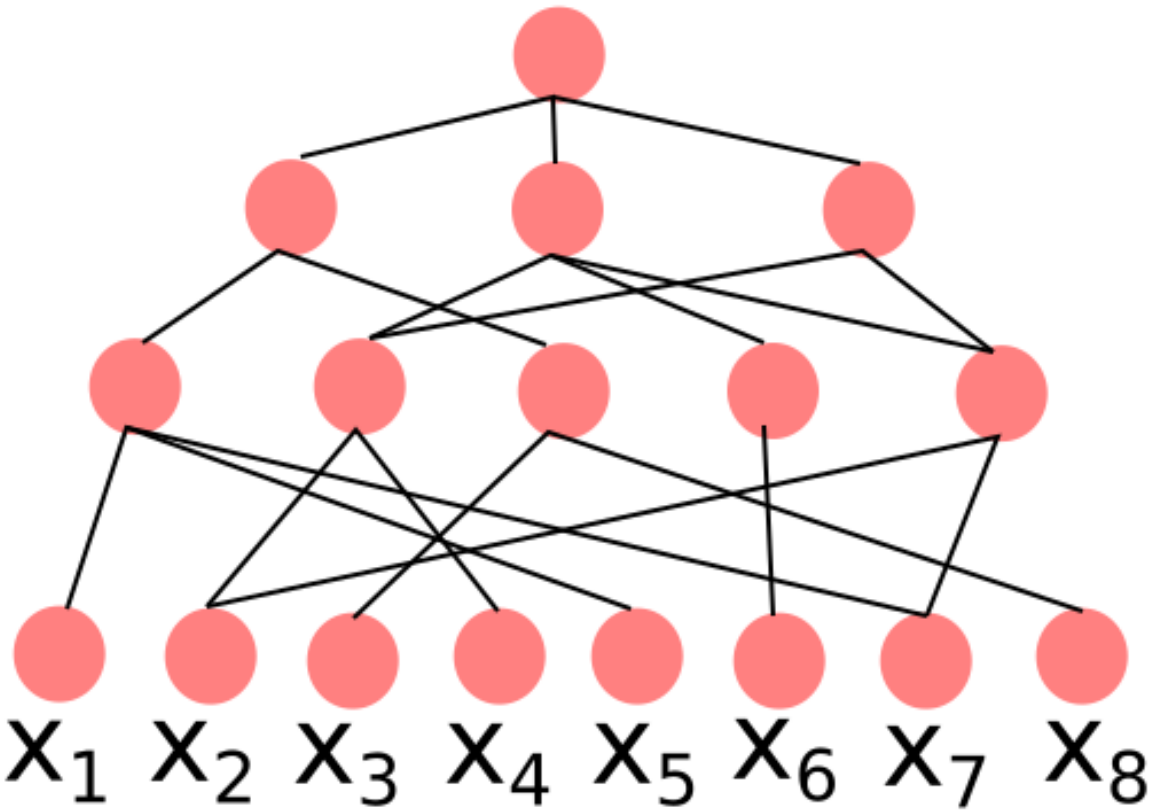
- 3 convolutional layers
- 1 to 14 filters in each layer
- Batch normalization

Shallow

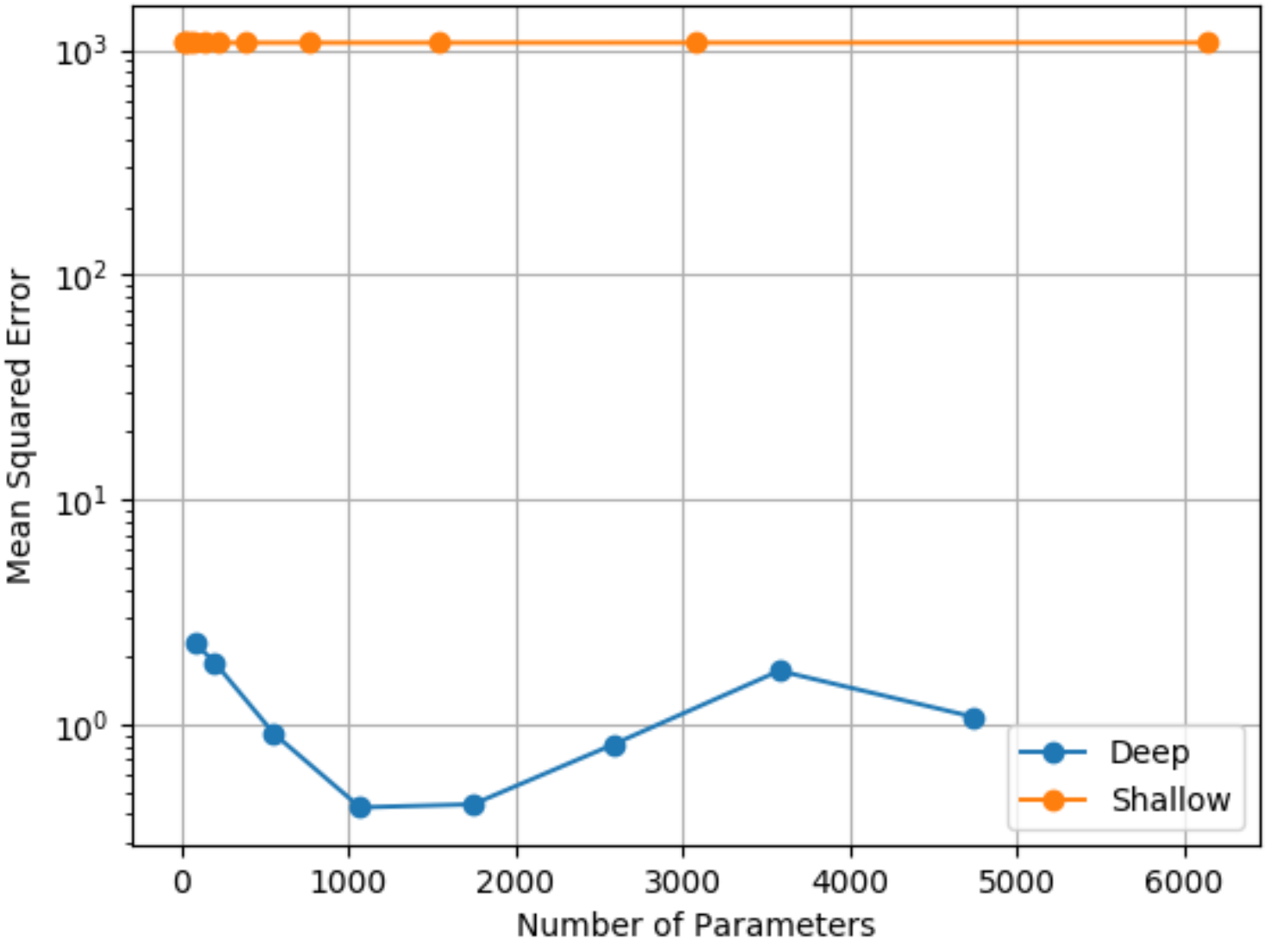
- 3 to 512 neurons in hidden layer

Deep Networks - Example

$$f(x_1, \ldots, x_8) = \text{sqrt}\left(\left((x_1 + 2x_5 + x_7)^4 + (x_8 + 2x_3)^4\right)^2 + \cos(x_4 + 2x_2)^2 \right.$$
$$\left. + x_6^2 + (x_2 + x_7)^4 + \left(\cos(x_4 + 2x_2) + (x_2 + x_7)^4\right)^2\right)$$



Deep Networks - Example



Deep Networks - Comments

- The theorem generalizes to $W_r^{n,d}$
- The constants in the approximation rate depends on the norm of the derivatives of f and σ
- The theorem generalizes to other activation, e.g. ReLU
  - Idea of proof: Smooth ReLU in an arbitrarily small interval and look at the limit.
- The theorem applies to functions with a compositional architecture that is a subgraph of the graph associated with $W_r^{n,d}$

Why is this an interesting result?

- Many functions have an underlying hierarchical composition.
- Example:

$$
\begin{aligned}
& f(x_1, x_2, x_3, x_4) \\
& = ac^2 x_4^4 x_1^3 + 2acx_1^3 x_4^2 x_3^3 + ax_1^3 x_3^6 + bc^2 x_1 x_2 x_4^4 + 2bcx_1 x_2 x_4^4 x_3^3 + bx_1 x_2 x_3^6 \\
& = (ax_1^3 + bx_2 x_1)(x_3^3 + cx_4^2)^2 \\
& = h(h_{11}(x_1, x_2), h_{12}(x_3, x_4))
\end{aligned}
$$

where

$$h(x, y) = xy^2, \quad h_{11}(x, y) = ax^3 + byx, \quad h_{12}(x, y) = x^3 + cy^2$$

- instead of approximating a 9th degree polynomial, a deep network approximates 3 polynomials of 3rd degree