Mathematics of Deep Learning
-
Practical Session

Jan Macdonald & Rafael Reisenhofer

Mathematisches Forschungszentrum Oberwolfach
16 October 2018

Outline

1. Approximation of functions in Sobolev spaces with deep networks
2. TensorFlow Tutorial I (Proof of upper bound)
3. Approximation of functions in Sobolev spaces with shallow networks
4. TensorFlow Tutorial II (Experimental verification)
5. Discussion of Results

Some Definitions

- $\mathcal{W}^{r,p}([0,1]^d)$: Sobolev space of r-times weakly differentiable functions in $L^p([0,1]^d)$ with norm

$$\|f\|_{\mathcal{W}^{r,p}([0,1]^d)} = \begin{cases} \left(\sum_{|\boldsymbol{\alpha}| \le r} \|D^{\boldsymbol{\alpha}} f\|_p^p \right)^{1/p} & 1 \le p < \infty, \\ \max_{|\boldsymbol{\alpha}| \le r} \|D^{\boldsymbol{\alpha}} f\|_\infty & p = \infty \end{cases}$$

- $F_{r,d}^p$: Unit ball in $\mathcal{W}^{r,p}([0,1]^d)$.
- Activation functions
  - ReLU: $\sigma(x) = \max(0, x)$
  - Sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}}$

Approximation of Functions in $F^{\infty}_{r,d}$ with Deep Networks

Theorem – Upper Bound (Yarotsky, 2017)

For any d, r, and $\epsilon \in (0, 1)$, there is a ReLU network architecture that can express any function from $F^{\infty}_{r,d}$ with error ϵ with depth at most $C(\ln(1/\epsilon) + 1)$ and **at most** $C\epsilon^{-d/r}(\ln(1/\epsilon) + 1)$ **weights**, *with some constant C depending on d and r.*

Theorem – Lower Bound (Yarotsky, 2017)

Fix d and r. For any $\epsilon \in (0, 1)$, a ReLU network architecture capable of approximating any function $f \in F^{\infty}_{r,d}$ with error ϵ must have **at least** $C\epsilon^{-d/2r}$ **weights**, *with some constant C depending on d and r.*

Proof Idea for Upper Bound

The proof that a ReLU architecture with depth at most $C(\ln(1/\epsilon)+1)$ and at most $C\epsilon^{-d/r}(\ln(1/\epsilon)+1)$ connections suffices to obtain an approximation error ϵ for any function in $F^{\infty}_{r,d}$ requires the following steps:

1. $f(x) = x^2$ on $[0,1]$ can be approximated with arbitrary precision with $O(ln(1/\epsilon))$ layers (explicit construction via iterated sawtooth functions)

Proof Idea for Upper Bound

The proof that a ReLU architecture with depth at most $C(\ln(1/\epsilon)+1)$ and at most $C\epsilon^{-d/r}(\ln(1/\epsilon)+1)$ connections suffices to obtain an approximation error ϵ for any function in $F^{\infty}_{r,d}$ requires the following steps:

1. $f(x) = x^2$ on $[0,1]$ can be approximated with arbitrary precision with $O(ln(1/\epsilon))$ layers (explicit construction via iterated sawtooth functions)
2. Due to $xy = ((x+y)^2 - x^2 - y^2)/2$, this implies that multiplication can be efficiently implemented.

Proof Idea for Upper Bound

The proof that a ReLU architecture with depth at most $C(\ln(1/\epsilon)+1)$ and at most $C\epsilon^{-d/r}(\ln(1/\epsilon)+1)$ connections suffices to obtain an approximation error ϵ for any function in $F^{\infty}_{r,d}$ requires the following steps:

1. $f(x) = x^2$ on $[0,1]$ can be approximated with arbitrary precision with $O(ln(1/\epsilon))$ layers (explicit construction via iterated sawtooth functions)
2. Due to $xy = ((x+y)^2 - x^2 - y^2)/2$, this implies that multiplication can be efficiently implemented.
3. Approximate arbitrary f as

$$\tilde{f} = \sum_n \varphi_n P_n,$$

where φ_n are piecewise linear functions that define a partition of unity and P_n are local Taylor polynomials.

Proof Idea for Upper Bound

The proof that a ReLU architecture with depth at most $C(\ln(1/\epsilon)+1)$ and at most $C\epsilon^{-d/r}(\ln(1/\epsilon)+1)$ connections suffices to obtain an approximation error ϵ for any function in $F^{\infty}_{r,d}$ requires the following steps:

1. $f(x) = x^2$ on $[0,1]$ can be approximated with arbitrary precision with $O(ln(1/\epsilon))$ layers (explicit construction via iterated sawtooth functions)

2. Due to $xy = ((x+y)^2 - x^2 - y^2)/2$, this implies that multiplication can be efficiently implemented.

3. Approximate arbitrary f as

$$\tilde{f} = \sum_n \varphi_n P_n,$$

   where φ_n are piecewise linear functions that define a partition of unity and P_n are local Taylor polynomials.

4. Construct a ReLU network that approximates φ_n, P_n, and the product $\varphi_n P_n$ sufficiently well.

TensorFlow Tutorial Part I

https://github.com/jmaces/mfo_dl_seminar_2018

Approximation of Functions in $F^p_{r,d}$ with Shallow Networks

Theorem – Upper Bound (Mhaskar, 1996)

Let $1 \leq p \leq \infty$. For any r, d, and $\epsilon \in (0, 1)$, there is a network with a smooth activation function and only one hidden layer that can express any function from $F^p_{r,d}$ with error ϵ **at most** $C\epsilon^{-d/r}$ **weights**, *with some constant C depending on d and r.*

Approximation of Functions in $F^p_{r,d}$ with Shallow Networks

Theorem – Upper Bound (Mhaskar, 1996)

Let $1 \le p \le \infty$. For any r, d, and $\epsilon \in (0, 1)$, there is a network with a smooth activation function and only one hidden layer that can express any function from $F^p_{r,d}$ with error ϵ **at most** $C\epsilon^{-d/r}$ **weights**, *with some constant C depending on d and r.*

Observation: Deep ReLU networks and shallow networks with smooth activation functions seem to have a similar approximation behavior with respect to functions in $F^p_{r,d}$.

Approximation of Functions in $F^p_{r,d}$ with Shallow Networks

Theorem – Upper Bound (Mhaskar, 1996)

Let $1 \leq p \leq \infty$. For any r, d, and $\epsilon \in (0, 1)$, there is a network with a smooth activation function and only one hidden layer that can express any function from $F^p_{r,d}$ with error ϵ **at most** $C\epsilon^{-d/r}$ **weights**, *with some constant C depending on d and r.*

Observation: Deep ReLU networks and shallow networks with smooth activation functions seem to have a similar approximation behavior with respect to functions in $F^p_{r,d}$.

Can we verify this numerically?

Experimental Setup – Network Topologies

- We consider network architectures with different parameters in three dimensions:

Activation Function:	$\{\mathrm{ReLU}, \mathrm{sigmoid}\}$
Depth:	$\{1, \ldots, 18\}$ (hidden layers)
Width:	$\{5, 10, \ldots, 50\}$

- We allow interconnections between layers: Each computational node (neuron) is connected to all nodes from all earlier layers.

- In the cases of only 1, 2, and 3 hidden layers, we consider widths up to 500.

Experimental Setup – Training and Testing

- We try to learn the sum of a smooth function $B(x, y)$ and a function $g_{r_1,r_2}(x, y)$ in $\mathcal{W}^{\min\{r_1,r_2\},p}([0,1]^d)$ on $[0,1]^2$:

$$B(x,y) = \underbrace{\binom{n_1}{k_1}_1 x^{n_1}(1-x)^{n_1-k_1}\binom{n_2}{k_2} y^{n_2}(1-y)^{n_2-k_2}}_{\text{smooth 2D Bernstein polynomial}},$$

$$g_{r_1,r_2}(x,y) = \underbrace{\operatorname{sgn}(x-0.4)(x-0.4)^{r_1}\operatorname{sgn}(y-0.6)(y-0.6)^{r_2}}_{\text{only } \min\{r_1,r_2\} \text{ weak derivatives at } (0.4,0.6)},$$

$$f(x,y) = B(x,y) + g_{r_1,r_2}(x,y)$$

Experimental Setup – Training and Testing

- We try to learn the sum of a smooth function $B(x,y)$ and a function $g_{r_1,r_2}(x,y)$ in $\mathcal{W}^{\min\{r_1,r_2\},p}([0,1]^d)$ on $[0,1]^2$:

$$B(x,y) = \underbrace{\binom{n_1}{k_1}_1 x^{n_1}(1-x)^{n_1-k_1}\binom{n_2}{k_2} y^{n_2}(1-y)^{n_2-k_2}}_{\text{smooth 2D Bernstein polynomial}},$$

$$g_{r_1,r_2}(x,y) = \underbrace{\operatorname{sgn}(x-0.4)(x-0.4)^{r_1}\operatorname{sgn}(y-0.6)(y-0.6)^{r_2}}_{\text{only } \min\{r_1,r_2\} \text{ weak derivatives at } (0.4,0.6)},$$

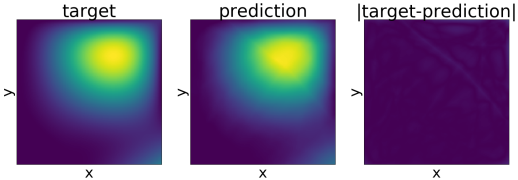
$$f(x,y) = B(x,y) + g_{r_1,r_2}(x,y)$$

- Training: 100k iterations of gradient descent with uniformly sampled training points and batch size 4096.
- Loss function: $\|\cdot\|_2$
- Learning rate: Exponential decrease from 0.2 to 0.001.
- Evaluation: $\|\cdot\|_\infty$ on 200×200 equidistant grid.

TensorFlow Tutorial II

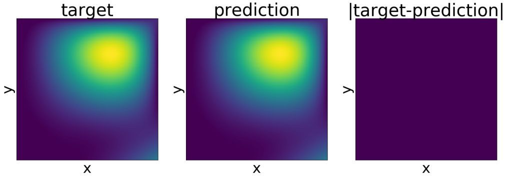
Results – Best ReLU vs. Best Sigmoid

- Best ReLU-based approximation



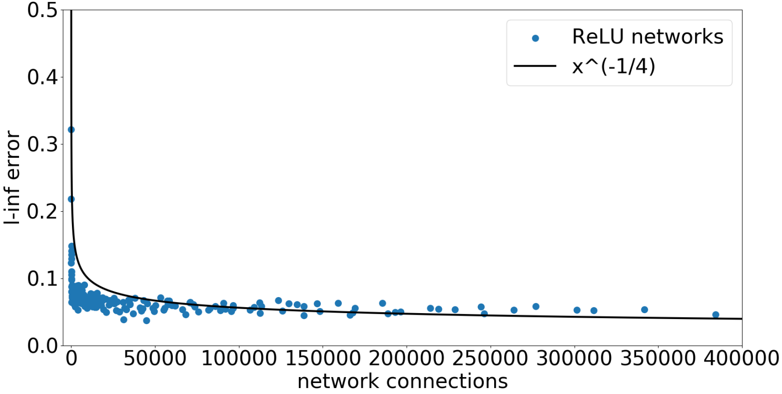
depth: 11, width: 35, $M = 44731$, ℓ_∞-error ≈ 0.00372.

- Best sigmoid-based approximation

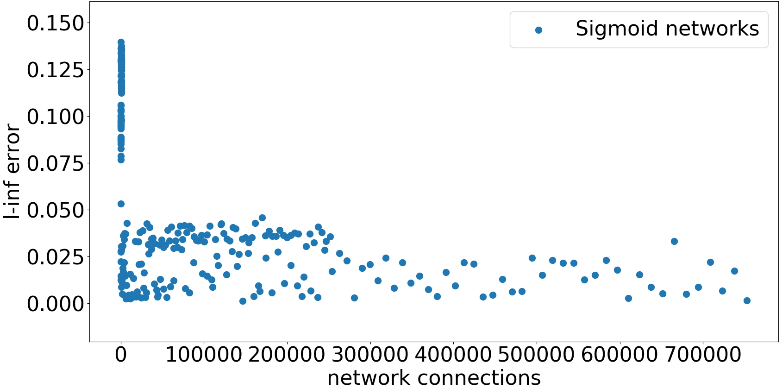


depth: 5, width: 220, $M = 146521$, ℓ_∞-error ≈ 0.00013

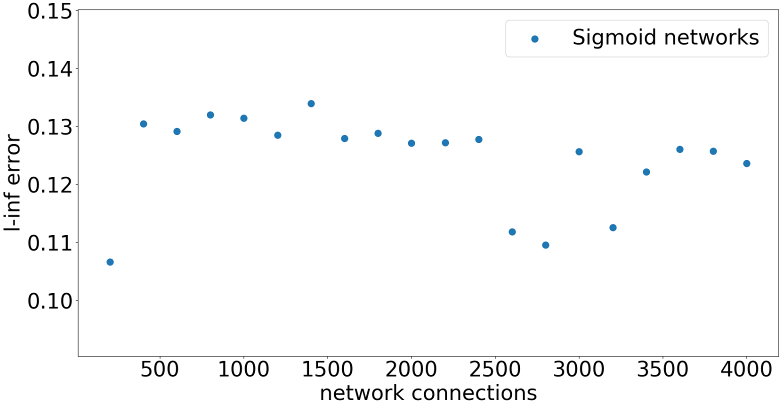
Size vs. Error in ReLU Networks



Size vs. Error in Shallow Sigmoid Networks



Size vs. Error in 3-Layer Sigmoid Networks



References

- D. Yarotsky. "Error bounds for approximations with deep ReLU networks". In: **Neural Networks** 94 (2017), pp. 103–114
- H. N. Mhaskar. "Neural networks for optimal approximation of smooth and analytic functions". In: **Neural computation** 8.1 (1996), pp. 164–177