Deep Network Approximation: Error Characterization in Terms of Width and Depth

Haizhao Yang
Department of Mathematics
Purdue University

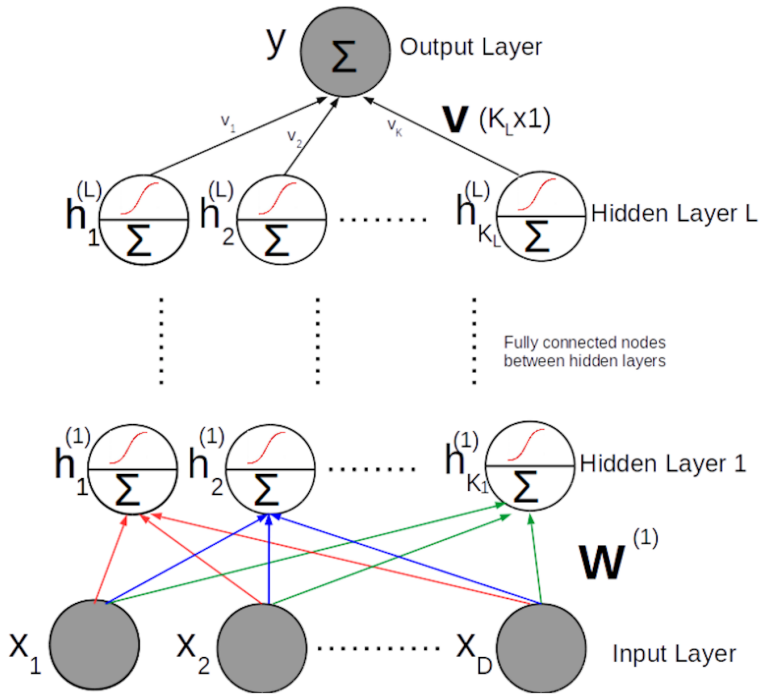
Scientific Machine Learning Seminar
October 30, 2021

Deep neural networks

$$y = h(x;\theta) := T \circ \phi(x) := T \circ h^{(L)} \circ h^{(L-1)} \circ \cdots \circ h^{(1)}(x)$$

where

- $h^{(i)}(x) = \sigma(W^{(i)\,T} x + b^{(i)})$;
- $T(x) = V^T x$;
- $\theta = (W^{(1)}, \ldots, W^{(L)}, b^{(1)}, \ldots, b^{(L)}, V)$.



Supervised deep learning

Conditions

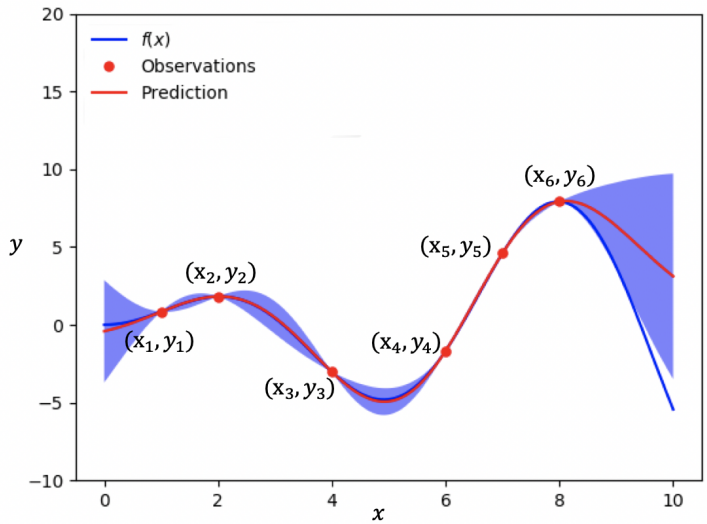
- Given data pairs $\{(x_i, y_i = f(x_i))\}$ from an unknown map $f(x)$ defined on Ω
- $\{x_i\}_{i=1}^n$ are sampled randomly from an unknown distribution $U(x)$ on Ω

Goal

Recover the unknown map $f(x)$

Deep learning

- Design a family of DNNs $\{h(x;\theta)\}_\theta$ of a given size
- Find the best DNN $h(x;\theta) \approx f(x)$ on Ω



Supervised deep learning

Deep learning ideally

- Quantify how good $h(x;\theta) \approx f(x)$ via the population loss:

$$R_D(\theta) \overset{\text{e.g.}}{=} \mathrm{E}_{x\sim U(\Omega)}\left[|h(x;\theta) - f(x)|^2\right]$$

- The best solution is $h(x;\theta_D)$ with

$$\theta_D = \operatorname{argmin} R_D(\theta)$$

- But $U(\Omega)$ is not known

Deep learning in practice

- Only the empirical loss is available:

$$R_S(\theta) := \frac{1}{N}\sum_{i=1}^{N}(h(x_i;\theta) - y_i)^2$$

- The best empirical solution is $h(x;\theta_S)$ with

$$\theta_S = \operatorname{argmin} R_S(\theta)$$

- Numerical optimization to obtain a numerical solution $h(x;\theta_N)$.
- In practice, $\theta_N \neq \theta_S \neq \theta_D$ and how good $R_D(\theta_N)$ is?

A full error analysis of $R_D(\theta_N)$

$$
\begin{aligned}
R_D(\theta_N) &= [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_N) - R_S(\theta_S)] + [R_S(\theta_S) - R_S(\theta_D)] \\
&\quad + [R_S(\theta_D) - R_D(\theta_D)] + R_D(\theta_D) \\
&\leq R_D(\theta_D) + [R_S(\theta_N) - R_S(\theta_S)] \\
&\quad + [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_D) - R_D(\theta_D)],
\end{aligned}
$$

A full error analysis of $R_D(\theta_N)$

$$
\begin{aligned}
R_D(\theta_N) &= [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_N) - R_S(\theta_S)] + [R_S(\theta_S) - R_S(\theta_D)] \\
&\quad + [R_S(\theta_D) - R_D(\theta_D)] + R_D(\theta_D) \\
&\leq R_D(\theta_D) + [R_S(\theta_N) - R_S(\theta_S)] \\
&\quad + [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_D) - R_D(\theta_D)],
\end{aligned}
$$

- $R_D(\theta_D) = \int_\Omega (h(x;\theta_D) - f(x))^2 d\mu(x) \leq \int_\Omega (h(x;\tilde{\theta}) - f(x))^2 d\mu(x)$ can be bounded by a constructive approximation of $\tilde{\theta}$

A full error analysis of $R_D(\theta_N)$

$$\begin{aligned} R_D(\theta_N) &= [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_N) - R_S(\theta_S)] + [R_S(\theta_S) - R_S(\theta_D)] \\ &\quad + [R_S(\theta_D) - R_D(\theta_D)] + R_D(\theta_D) \\ &\leq R_D(\theta_D) + [R_S(\theta_N) - R_S(\theta_S)] \\ &\quad + [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_D) - R_D(\theta_D)], \end{aligned}$$

- $R_D(\theta_D) = \int_\Omega (h(x;\theta_D) - f(x))^2 d\mu(x) \leq \int_\Omega (h(x;\tilde{\theta}) - f(x))^2 d\mu(x)$ can be bounded by a constructive approximation of $\tilde{\theta}$
- $[R_S(\theta_N) - R_S(\theta_S)]$ is the optimization error

A full error analysis of $R_D(\theta_N)$

$$
\begin{aligned}
R_D(\theta_N) &= [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_N) - R_S(\theta_S)] + [R_S(\theta_S) - R_S(\theta_D)] \\
&\quad + [R_S(\theta_D) - R_D(\theta_D)] + R_D(\theta_D) \\
&\leq R_D(\theta_D) + [R_S(\theta_N) - R_S(\theta_S)] \\
&\quad + [R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_D) - R_D(\theta_D)],
\end{aligned}
$$

- $R_D(\theta_D) = \int_\Omega (h(x;\theta_D) - f(x))^2 d\mu(x) \leq \int_\Omega (h(x;\tilde{\theta}) - f(x))^2 d\mu(x)$ can be bounded by a constructive approximation of $\tilde{\theta}$
- $[R_S(\theta_N) - R_S(\theta_S)]$ is the optimization error
- Other two terms concern the generalization from samples to distribution

Deep Network Approximation

Approximation Problem Statement:

$$R_D(\theta_D) = \int_\Omega (h(x;\theta_D) - f(x))^2 d\mu(x) \leq \int_\Omega (h(x;\tilde{\theta}) - f(x))^2 d\mu(x) \leq \epsilon$$

Conditions:

- Fix an architecture of DNN
- $f(x)$ is known
- $\tilde{\theta}$ can take any real or complex values

Deep Network Approximation

Optimization Problem Statement:

$$0 \leq R_S(\theta_N) - R_S(\theta_S)$$

Conditions:

- Fix an architecture of DNN
- $f(x)$ is unknown and only $\{x_i, f(x_i)\}_{i=1}^n$ are given
- θ_N and the optimization path need to be encoded on computers

Remark: $\tilde{\theta}$ and θ_N are different

Deep Network Approximation

Generalization Problem Statement:

$$[R_D(\theta_N) - R_S(\theta_N)] + [R_S(\theta_D) - R_D(\theta_D)],$$

Conditions:

- Fix an architecture of DNN
- Bounded by Rademacher complexity + $O\left(\frac{1}{\sqrt{n}}\right)$

Deep Network Approximation

Approximation Problem Statement:

$$R_D(\theta_D) = \int_\Omega (h(x;\theta_D) - f(x))^2 d\mu(x) \leq \int_\Omega (h(x;\tilde{\theta}) - f(x))^2 d\mu(x) \leq \epsilon$$

Conditions:

- Fix an architecture of DNN
- $f(x)$ is known
- $\tilde{\theta}$ can take any real or complex values

Our goals in approximation

- Approximation error in terms of width and depth
- Is an exponentially large number of parameters required? e.g., # parameters not $(\frac{1}{\epsilon})^d$
- Is exponential approximation rate available? e.g., # parameters $\log(\frac{1}{\epsilon})$

Literature Review

Active research directions

Cybenko, 1989; Hornik et al., 1989; Barron, 1993; Liang and Srikant, 2016; Yarotsky, 2017; Poggio et al., 2017; Schmidt-Hieber, 2017; E and Wang, 2018; Petersen and Voigtlaender, 2018; Chui et al., 2018; Yarotsky, 2018; Nakada and Imaizumi, 2019; Gribonval et al., 2019; Gühring et al., 2019; Chen et al., 2019; Li et al., 2019; Suzuki, 2019; Bao et al., 2019; E et al., 2019; Opschoor et al., 2019; Yarotsky and Zhevnerchuk, 2019; Bölcskei et al., 2019; Montanelli and Du, 2019; Chen and Wu, 2019; Zhou, 2020; Montanelli et al., 2020, etc.

Functions spaces

- Continuous functions
- Smooth functions
- Functions with integral representations

ReLU DNNs, continuous functions $C([0,1]^d)$

ReLU; Fixed network width $O(N)$ and depth $O(L)$

- Nearly tight error rate $5\omega_f(8\sqrt{d}N^{-2/d}L^{-2/d})$ simultaneously in N and L with L^∞-norm. Shen, Y., and Zhang (CiCP, 2020)
- ω_f is the modulas of continuity
- Improved to a tight rate $O\left(\sqrt{d}\,\omega_f\left(\left(N^2L^2\log_3(N+2)\right)^{-1/d}\right)\right)$. Shen, Y., and Zhang (J Math Pures Appl, 2021)

Curse of dimensionality exists!

ReLU DNNs, smooth functions $C^s([0,1]^d)$

Does smoothness help?

ReLU; Fixed network width $O(N \log N)$ and depth $O(L \log L)$

- Nearly tight rate $85(s+1)^d 8^s \|f\|_{C^s([0,1]^d)} N^{-2s/d} L^{-2s/d}$ simultaneously in N and L with L^∞-norm
- Lu, Shen, Y., and Zhang (SIMA, 2021)

The curse of dimensionality exists if s is fixed.

ReLU DNNs, smooth functions $C^s([0,1]^d)$

What about other norms?

ReLU or ReLU^2; Fiexed width $\mathcal{O}(N \log N)$ and depth $\mathcal{O}(L \log L)$

- Rate $\mathcal{O}(N^{-2(s-n)/d} L^{-2(s-n)/d})$
- The $\mathcal{W}^{n,p}([0,1]^d)$-norm
- Hon and Y. (arXiv:2109.00161)

The curse of dimensionality exists if s is fixed.

DNNs with advanced activation function

Sine-ReLU; Fixed width $O(d)$, varying depth L

- $\exp(-c_{r,d}\sqrt{L})$ with L^∞-norm for $C^r([0,1]^d)$
- Root exponential approximation rate achieved
- Curse of dimensionality is not clear
- Yarotsky and Zhevnerchuk, NeurIPS 2020

Floor and ReLU activation, width $O(N)$ and depth $O(dL)$, $C([0,1]^d)$

- Error rate $\omega_f(\sqrt{d}N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-\sqrt{L}}$ with L^∞-norm
- NO curse of dimensionality for many continuous functions
- Root exponential approximation rate
- Merely based on the compositional structure of DNNs and depth is the key
- Shen, Y., and Zhang (Neural Computation, 2020)

Further interpretation of our result

Explicit error bound

Floor and ReLU activation, width $O(N)$ and depth $O(dL)$, Hölder$([0,1]^d, \alpha, \lambda)$

- Error rate $3\lambda d^{\alpha/2} N^{-\alpha\sqrt{L}}$ with L^∞-norm
- NO curse of dimensionality
- Root exponential approximation rate
- Shen, Y., and Zhang (Neural Computation, 2020)

Further interpretation of our result

Can we get an error bound in terms of the number of parameters $O(W)$?

Floor and ReLU activation, width $O(d)$ and depth $O(dW)$, $C([0,1]^d)$

- Error rate $\omega_f(\sqrt{d}\,2^{-\sqrt{W}}) + 2\omega_f(\sqrt{d})2^{-\sqrt{W}}$ with L^∞-norm
- NO curse of dimensionality for many continuous functions
- Root exponential approximation rate
- Shen, Y., and Zhang (Neural Computation, 2020)

Further interpretation of our result

Does smoothness help? No

Floor and ReLU activation, width $O(N)$ and depth $O(dL)$, $C^s([0,1]^d)$

- Expected error rate $O(\omega_f(\sqrt{d}N^{-s\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-s\sqrt{L}})$ with a prefactor $O((s+1)^d)$ in the L^∞-norm

Further interpretation of our result

Does the domain $[0,1]^d$ matter? No

Floor and ReLU activation, width $O(N)$ and depth $O(dL)$, $C([-M,M]^d)$

- Error rate $\omega_f^{[-M,M]^d}(2M\sqrt{d}\,N^{-\sqrt{L}}) + 2\omega_f^{[-M,M]^d}(2M\sqrt{d})N^{-\sqrt{L}}$ in the L^∞-norm

Further interpretation of our result

Does ω_f matter? Yes

Floor and ReLU activation, width $O(N)$ and depth $O(dL)$, $C([0,1]^d)$

- Error rate $\omega_f(\sqrt{d}N^{-\sqrt{L}}) + 2\omega_f(\sqrt{d})N^{-\sqrt{L}}$ with L^∞-norm

- $\omega_f(r) = \frac{1}{\ln(1/r)}$

  $$3(\sqrt{L}\ln N - \tfrac{1}{2}\ln d)^{-1}$$

- $\omega_f(r) = \frac{1}{\ln^{1/d}(1/r)}$

  $$3(\sqrt{L}\ln N - \tfrac{1}{2}\ln d)^{-1/d}$$

- $\omega_f(r) = r^{\alpha/d}$

  $$3\lambda d^{\frac{\alpha}{2d}} N^{-\frac{\alpha}{d}\sqrt{L}}$$

DNNs with advanced activation function

Can width be as powerful as depth?

Floor, Sign, and 2^x activation, width $O(N)$ and depth 3, $C([0,1]^d)$

- Error rate $\omega_f(\sqrt{d}2^{-N}) + 2\omega_f(\sqrt{d})2^{-N}$ with L^∞-norm
- NO curse of dimensionality for many continuous functions
- Exponential approximation rate
- Merely based on the compositional structure of DNNs and width is the key
- Shen, Y., and Zhang (Neural Networks, 2021)

Further interpretation of our result

Explicit error bound

Floor, Sign, and 2^x activation, width $O(N)$ and depth 3, Hölder$([0,1]^d, \alpha, \lambda)$

- Error rate $3\lambda(2\sqrt{d})^{\alpha}2^{-\alpha N}$ with L^{∞}-norm
- NO curse of dimensionality
- Exponential approximation rate
- Shen, Y., and Zhang (Neural Networks, 2021)

Key ideas of our approximation

For $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$:
$\boldsymbol{x} \to \phi_1(\boldsymbol{x}) = \boldsymbol{\beta} \to \phi_2(\boldsymbol{\beta}) = k_{\boldsymbol{\beta}} \to \phi_3(k_{\boldsymbol{\beta}}) = f(\boldsymbol{x}_{\boldsymbol{\beta}}) \approx f(\boldsymbol{x})$

- Piecewise constant approximation:
  $f(\boldsymbol{x}) \approx f_p(\boldsymbol{x}) \approx \phi_3 \circ \phi_2 \circ \phi_1(\boldsymbol{x})$
- 2^N pieces per dim and 2^{Nd} pieces with accuracy 2^{-N}
- Floor NN $\phi_1(\boldsymbol{x})$ s.t. $\phi_1(\boldsymbol{x}) = \boldsymbol{\beta}$ for $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ and $\boldsymbol{\beta} \in \mathbb{Z}^d$.
- Linear NN ϕ_2 mapping $\boldsymbol{\beta}$ to an integer $k_{\boldsymbol{\beta}} \in \{1, \ldots, 2^{Nd}\}$
- Key difficulty: NN ϕ_3 of width $O(N)$ and depth $O(1)$ fitting 2^{Nd} samples in 1D with accuracy $O(2^{-N})$
- ReLU NN fails

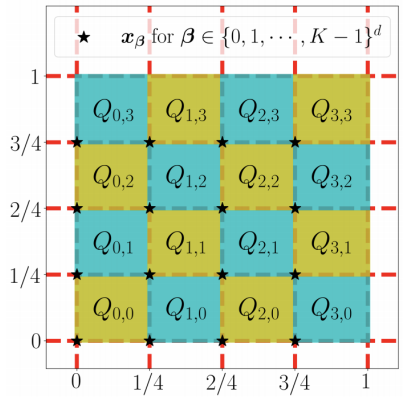


Figure: Uniform domain partitioning.

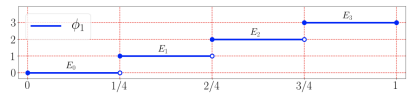


Figure: Floor function.

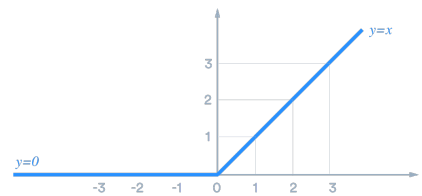


Figure: ReLU function.

Key ideas of our approximation

Binary representation and approximation

$\theta = \sum_{\ell=1}^{\infty} \theta_\ell 2^{-\ell}$ with $\theta_\ell \in \{0, 1\}$ is approximated by $\sum_{\ell=1}^{N} \theta_\ell 2^{-\ell}$ with an error 2^{-N}.

Bit extraction via a floor NN of width 2 and depth 1

$$\phi_k(\theta) := \lfloor 2^k \theta \rfloor - 2\lfloor 2^{k-1} \theta \rfloor = \theta_k$$

Bit extraction via a floor NN of width $2N$ and depth 1

Given $\theta = \sum_{\ell=1}^{\infty} \theta_\ell 2^{-\ell}$

$$\phi(\theta) := \begin{pmatrix} \phi_1(\theta) \\ \vdots \\ \phi_N(\theta) \end{pmatrix} = \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_N \end{pmatrix} \in \mathbb{Z}^N$$

Key ideas of our approximation

Encoding K numbers to one number

- Extract bits $\{\theta_1^{(k)}, \ldots, \theta_N^{(k)}\}$ from $\theta^{(k)} = \sum_{\ell=1}^{\infty} \theta_\ell^{(k)} 2^{-\ell}$ for $k = 1, \ldots, K$
- sum up to get

$$a = \sum_{\ell=1}^{N} \theta_\ell^{(1)} 2^{-\ell} + \sum_{\ell=N+1}^{2N} \theta_{\ell-N}^{(2)} 2^{-\ell} + \cdots + \sum_{\ell=(K-1)N+1}^{KN} \theta_{\ell-(K-1)N}^{(K)} 2^{-\ell}$$

Decoding one number to get the k-th number

- Extract bits $\{\theta_1^{(k)}, \ldots, \theta_N^{(k)}\}$ from a via

$$\psi(k) := \phi(2^{(k-1)N} a - \lfloor 2^{(k-1)N} a \rfloor).$$

- sum up to get $\theta^{(k)} \approx \sum_{\ell=1}^{N} \theta_\ell^{(k)} 2^{-\ell} = [2^{-1}, \ldots, 2^{-N}]\psi(k) := \gamma(k)$,
- $\gamma(k)$ is an NN of width $O(N)$ and depth $O(1)$.

Key Lemma

There exists an NN γ of width $O(N)$ and depth $O(1)$ that can memorize arbitrary samples $\{(k, \theta^{(k)}\}_{k=1}^{K}$ with a precision 2^{-N}.

Key ideas of our approximation

For $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$:
$\boldsymbol{x} \to \phi_1(\boldsymbol{x}) = \boldsymbol{\beta} \to \phi_2(\boldsymbol{\beta}) = k_{\boldsymbol{\beta}} \to \phi_3(k_{\boldsymbol{\beta}}) = f(\boldsymbol{x}_{\boldsymbol{\beta}}) \approx f(\boldsymbol{x})$

- Piecewise constant approximation:
  $f(\boldsymbol{x}) \approx f_p(\boldsymbol{x}) \approx \phi_3 \circ \phi_2 \circ \phi_1(\boldsymbol{x})$
- 2^N pieces per dim and 2^{Nd} pieces with accuracy 2^{-N}
- Floor NN $\phi_1(\boldsymbol{x})$ s.t. $\phi_1(\boldsymbol{x}) = \boldsymbol{\beta}$ for $\boldsymbol{x} \in Q_{\boldsymbol{\beta}}$ and $\boldsymbol{\beta} \in \mathbb{Z}^d$.
- Linear NN ϕ_2 mapping $\boldsymbol{\beta}$ to an integer $k_{\boldsymbol{\beta}} \in \{1, \ldots, 2^{Nd}\}$
- Key difficulty: NN ϕ_3 of width $O(N)$ and depth $O(1)$ fitting 2^{Nd} samples in 1D with accuracy $O(2^{-N})$
- Key Lemma: There exists an NN γ of width $O(N)$ and depth $O(1)$ that can memorize arbitrary samples $\{(k, \theta^{(k)}\}_{k=1}^K$ with a precision 2^{-N}.

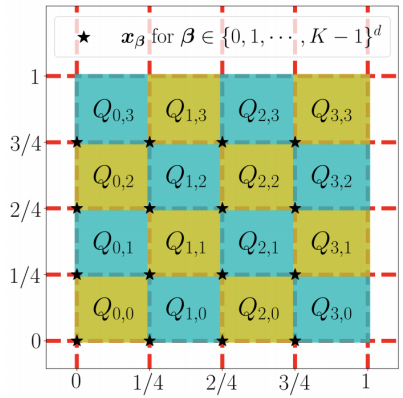


Figure: Uniform domain partitioning.

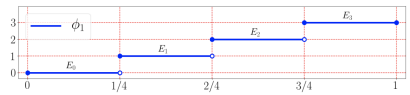


Figure: Floor function.

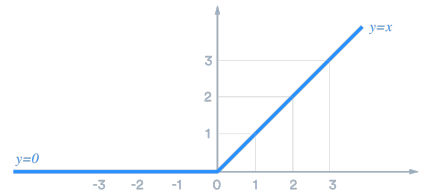


Figure: ReLU function.

Further interpretation of our result

Realistic consideration

- Constructive approximation requires f or exponentially many samples given
- Constructed parameters require high precision computation
- Floor and Sign are discontinuous functions leading to gradient vanishing
- The network size has to be increased when $\epsilon \to 0$

Blue items will be addressed later.

Acknowledgment

Collaborators

Qiang Du, Sean Hon, Jianfeng Lu, Hadrien Montanelli, Zuowei Shen, Shijun Zhang

Funding

National Science Foundation under the grant award 1945029