What makes DNNs stand out in approximation

Yebiao Jin

University of Pennsylvania

yebiao@sas.upenn.edu

October 24, 2019

Overview

1. Motivation for my research on DNN approximation
   - Literature about smooth/non-smooth function appxoimation
2. DNN for smooth function approximation
3. DNN for general non-smooth function approximation
4. Summary

Two papers of interest

- Shiyu Liang & R.Srikant - Why **deep** neural networks for function approximation? (ICLR 2017)
- Masaaki Imaizumi & Kenji Fukumizu - Deep neural networks learn **non-smooth** functions effectively (2018)

Some related topics: Candes and Donoho (2002,2004) on curvelet, Kutyniok and Lim(2011) on shearlet with Harmonic analysis.

- Number of neurons ($\mathcal{O}(poly(1/\epsilon))$) needed by a shallow network to approximate a function is **exponentially** larger than the number($\mathcal{O}(polylog(1/\epsilon))$) needed by a deep neural network.
- Neural networks use a combination of ReLUs and binary step units, based on a simple observation: multiplication of two **bits** can be represened by a ReLU.
- Results can be extended to certain classes of important multivariate functions.

Notations and setup

- $\tilde{f} : \mathbf{R}^d \to \mathbf{R}$ denotes a feedforward neural network
- $N = \sum_{l=1}^{L} N_l$ denotes the number of neurons on L hidden layers.
- Only consider two types of activation functions: ReLU and BSU.
- $\mathcal{F}(N, L)$ denotes the family of all feedforward neural networks of depth L and size N and composed of a combination of ReLU and BSU.
- Consider $\min_{\tilde{f} \in \mathcal{F}(N,L)} \|f - \tilde{f}\|_\infty \leq \epsilon$
  - Existence of upper bound $L(\epsilon)$ and $N(\epsilon)$?
  - Given a fixed depth L, the minimum of size N?

Begin with $f(x) = x^2$

Theorem

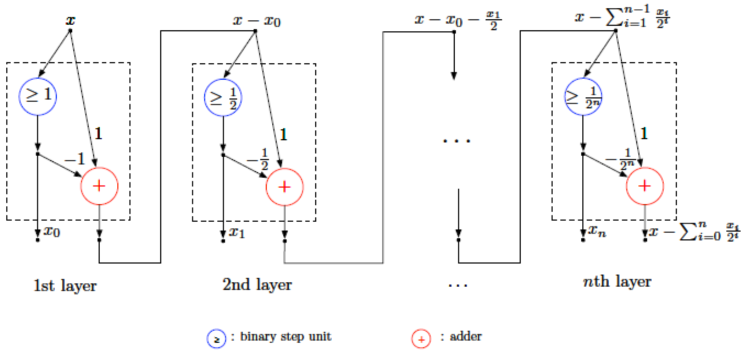
For function $f(x) = x^2$, $x \in [0,1]$, there exists a multilayer neural network $\tilde{f}(x)$ with $\mathcal{O}(\log \frac{1}{\epsilon})$ BSU and $\mathcal{O}(\log \frac{1}{\epsilon})$ ReLU such that $|f(x) - \tilde{f}(x)| \leq \epsilon$, $\forall x \in [0,1]$.

Proof sketch:

- Show BSU for binary expansion $\tilde{x} = \sum_{i=0}^{n} \frac{x_i}{2^i}$ with multilayer network
- Construct 2-layer ReLU neural network for $f(\tilde{x})$
- Check approximation error $|f(x) - \tilde{f}(x)|$

Proof of theorem

BSU for finding the binary expansion



Proof of theorem

Implementing the function $\tilde{f}(x) = f(\sum_{i=0}^{n} \frac{x_i}{2^i})$ by a two-layer ReLU neural network:

$$\tilde{f}(x) = (\sum_{i=0}^{n} \frac{x_i}{2^i})^2 = \sum_{i=0}^{n} x_i \left(\frac{1}{2^i} \sum_{j=0}^{n} \frac{x_j}{2^j} \right) \tag{1}$$

$$= \sum_{i=0}^{n} \max \left\{ 0, 2(x_i - 1) + \frac{1}{2^i} \sum_{j=0}^{n} \frac{x_j}{2^j} \right\} \tag{2}$$

Hence, the weight matrix can be represented as

$$w_{ij} = \begin{cases} 2 + \frac{1}{2^{2i}} & i = j \\ \frac{1}{2^{i+j}} & i \neq j \end{cases}$$

for $0 \leq i, j \leq n$

Proof of theorem

The approximation error is trivial:

$$|f(x) - \tilde{f}(x)| \leq 2 \left| x - \sum_{i=0}^{n} \frac{x_i}{2^i} \right| = 2 \left| \sum_{i=n+1}^{\infty} \frac{x_i}{2^i} \right| \leq \frac{1}{2^{n-1}} \tag{3}$$

To achieve ϵ-approximation error, $n = \lceil \log_2 \frac{1}{\epsilon} \rceil + 1$. In summary, the DNN needs $\mathcal{O}(\log \frac{1}{\epsilon})$ layers, $\mathcal{O}(\log \frac{1}{\epsilon})$ BSU and $\mathcal{O}(\log \frac{1}{\epsilon})$ ReLU.

Generalization to polynomials

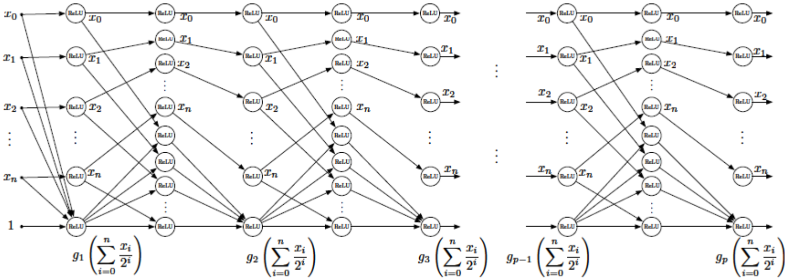
Theorem

For polynomials $f(x) = \sum_{i=0}^{p} a_i x^i$,$x \in [0,1]$ *and* $\sum_{i=1} p|a_i| \leq 1$, *there exists a multilayer neural network* $\tilde{f}(x)$ *with* $\mathcal{O}(p + \log\frac{p}{\epsilon})$ *layers,* $\mathcal{O}(\log\frac{p}{\epsilon})$ *BSU and* $\mathcal{O}(p\log\frac{p}{\epsilon})$ *ReLU such that* $|f(x) - \tilde{f}(x)| \leq \epsilon$, $\forall x \in [0,1]$.

The proof is quite similar, if we let $g_i(x) = x^i$ and hence we can rewrite

$$g_{m+1}(\sum_{i=0}^{n} \frac{x_i}{2^i}) = \sum_{i=0}^{n} \max\left[0, 2(x_i - 1) + \frac{1}{2^i} g_m(\sum_{j=0}^{n} \frac{x_j}{2^j})\right] \tag{4}$$

Generalization to polynomials



The rest of proof is trivial!

More generalizations

Theorem

Assume that function f is continuous on $[0, 1]$ and $\lceil \log \frac{2}{\epsilon} \rceil + 1$ times differentiable in $(0, 1)$. Let $f^{(n)}$ denote the derivative of f of nth order and $\|f\| = \max_{x \in [0,1]} f(x)$. If $\|f^{(n)}\| \leq n!$ holds for all $n \in [\lceil \log \frac{2}{\epsilon} \rceil + 1]$, then there exists a deep neural network $\tilde{f}$ with $\mathcal{O}(\log \frac{1}{\epsilon})$ layers, $\mathcal{O}(\log \frac{1}{\epsilon})$ BSU, $\mathcal{O}((\log \frac{1}{\epsilon})^2)$ ReLU such that $\|f - \tilde{f}\| \leq \epsilon$.

And this theorem can lead to corollaries with minor difference for function addition, multiplication and composition if $h_1, h_2, \cdots, h_k$ satisfy condition in theorem.

Proceed to next one!

- The main contribution of the first paper is about shallow vs deep neural networks. The proof is easy but solid.
- Personal opinion: the binary expansion is like harmonic analysis used with Fourier transform, curvelet transform and shearlet transform for function approximation.
- In contrast, the next paper is about DNNs vs other popular models for approximation, and for certain classes of **non-smooth multivariate** functions.

Deep neural networks learn non-smooth functions effectively

Abstract

- It's known that many standard methods attain the optimal rate of generalization errors for **smooth** functions in large sample asymptotics, so DNNs do not stand out in this case.
- This paper theoretically derives the generalization error of estimators by DNNs with ReLU activation and shows that the convergence rate are **almost optimal**.

Notation

- $f : I^D = [0,1]^D \to \mathbf{R}$
- $H^\beta(\Omega)$ denotes the space of smooth function $f : \Omega \to \mathbf{R}$ such that f are $\lfloor \beta \rfloor$-times differentiable and the $\lfloor \beta \rfloor$-th derivatives are $\beta - \lfloor \beta \rfloor$ **-Hölder continuous**.
- $A = \{x \in I^D | \Psi_h(x) = 1\}$ where $h \in H^\alpha(I^{D-1})$ and $\Psi_h(x) = \Psi(x_1, \cdots, x_d \pm h(x \setminus x_d), \cdots, x_D)$, $\Psi : I^D \to \{0,1\}$ denotes a **basis piece**.
- $\mathcal{R}_{\alpha,J} = \left\{ R \subset I^D : R = \cap_{j=1}^J A_j \right\}$ denotes the set of piecewise α-smooth boundaries.
- $\mathcal{F}_{M,J,\alpha,\beta} = \left\{ \sum_{m=1}^M f_m 1_{Rm} : f_m \in H^\beta(I^D), R_m \in \mathcal{R}_{\alpha,J} \right\}$ denotes set of piecewise smooth fnctions.
- $\mathcal{F}_{NN,\eta}(S, B, L)$ denotes the set of DNNs with activation η, parameter sparsity upper bound S, parameter bound B and depth bound L.

Optimal rate of generalization with smoothness assumption

Suppose the data $\{(Y_i, X_i)\}$ are given by

$$Y_i = f(X_i) + \xi_i, \quad \xi_i \sim \mathcal{N}(0, \sigma^2)$$

with $f \in H^\beta(I^D)$ and $X_i \in I^D$.
Methods such as kernel methods, Gaussian processes, series methods, as well as DNNs, achieve generalization errors of the order of

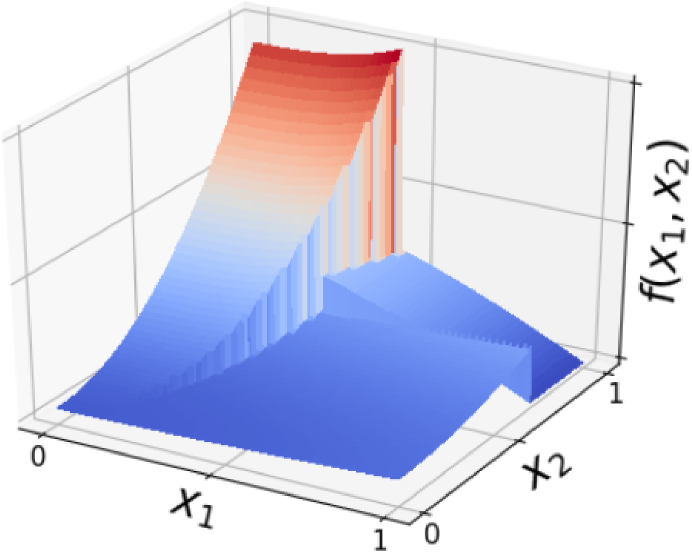
$$O(n^{-2\beta/(2\beta+D)})$$

How about the piecewise smooth case?

The contribution of the paper

- Derive a rate of convergence of the generalization errors in the estimators by DNNs for the class of piecewise smooth functions.
- Prove that DNNs theoretically outperform other standard methods for data from non-smooth generating processes.
- Provide a practical guideline on the structure of DNNs, i.e show a necessary number of layers and parameters of DNNs to achieve the rate of convergence.

An example of piecewise function



Convergence rate of generalization errors

Theorem

Suppose $f^* \in \mathcal{F}_{M,J,\alpha,\beta}$. *Then, there exist constants* c_1, c_1', $C_L > 0$, $s \in \mathbb{N} \setminus \{1\}$ *and* (S, B, L) *satisfying*

(i) $S = c_1' \max\{n^{D/(2\beta+D)}, n^{(D-1)/(\alpha+D-1)}\}$

(ii) $B \geq c_1 n^s$

(iii) $L \leq c_1(1 + \max\{\beta/D, \alpha/2(D-1)\})$

such that $\hat{f}^L \in \mathcal{F}_{NN,\eta}(S, B, L)$ *provides*

$$\|\hat{f}^L - f^*\|_{L^2(P_X)}^2 \leq C_L M^2 J^2 \max\{n^{-2\beta/(2\beta+D)}, n^{-\alpha/(\alpha+D-1)}(\log n)^2\}$$

with probability at least $1 - c_1 n^{-2}$

Minimax optimal rate of convergence

Theorem

Consider $\bar{f}$, an **arbitrary estimator** *of $f^* \in \mathcal{F}_{M,J,\alpha,\beta}$. Then, there exists a constant $C_{mm} > 0$ such that*

$$\inf_{\bar{f}} \sup_{f^* \in \mathcal{F}_{M,J,\alpha,\beta}} \mathbb{E}_{f^*}\left[\|\bar{f} - f^*\|^2_{L^2(P_X)}\right] \geq C_{mm} \max\{n^{-\frac{2\beta}{2\beta+D}}, n^{-\frac{\alpha}{\alpha+D-1}}\}$$

The rate of convergence of the DNN estimators is optimal in the minimax sense, since the rate is only up to a log factor.

Non-Optimality of other methods

We consider a class of linear estimators:

$$\hat{f^{lin}}(x) = \sum_{i \in [n]} \Psi_i(x; X_1, \cdots, X_n) Y_i$$

which contains kernel methods, Fourier estimators, splines, Gaussian process and others.

Theorem

Linear estimators do not attain the optimal rate for $\mathcal{F}_{M,J,\alpha,\beta}$. Hence, there exist $f^* \in \mathcal{F}_{M,J,\alpha,\beta}$ such that $\hat{f}$ and any $\hat{f}^{lin}$, for large n we have

$$\mathbb{E}_{f^*}\left[\|\hat{f}^L - f^*\|^2_{L^2(P_X)}\right] < \mathbb{E}_{f^*}\left[\|\hat{f}^{lin} - f^*\|^2_{L^2(P_X)}\right]$$

Intuition behind optimality of DNN with ReLU

One notable intuition on why DNNs are optimal: DNNs can approximate non-smooth functions with a small number of parameters, due to activation functions and multi-layer structures.

$$\mathbf{1}_{\{x\geq 0\}} \approx \eta(ax) - \eta(ax-1) = \begin{cases} 1 & x \geq \frac{1}{a} \\ ax & 0 < x < \frac{1}{a} \\ 0 & x \leq 0 \end{cases}$$

with sufficiently large $a > 0$.

Experiments

$$f^*(x) = \mathbf{1}_{R_1}(x)(0.2 + x_1^2 + 0.1x_2) + \mathbf{1}_{R_2}(x)(0.7 + 0.01|4x_1 + 10x_2 - 9|^{1.5})$$

with $R_1 = \{(x_1, x_2) \in I^2 : x_2 \geq -0.6x_1 + 0.75\}$ and $R_2 = I^2 \setminus R_1$. And the DNN is of $D_1 = 2, D_l = 3$ for $l \in \{2, 3, 4\}$ and $D_5 = 1$ with ReLU activation.

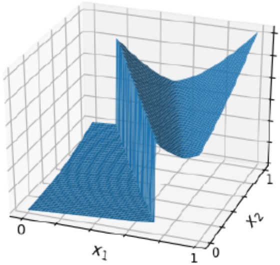


FIGURE 2. A plot for $f^*(x_1, x_2)$ with $(x_1, x_2) \in I^2$.

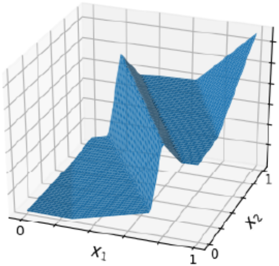
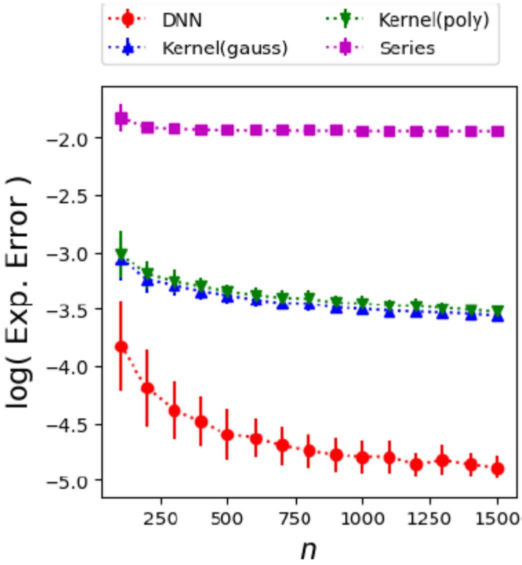


FIGURE 3. A plot for the estimator $\widehat{f}^L$.

Comparison of errors



Summary

- Both papers contribute to my project either practically or theoretically.
- A rough guidance on architecture of DNN is provided.
- That DNNs learn non-smooth functions effectively provides me with theoretical backup to extend the setting in my model where control variables can be a collection of piecewise continuous and discrete functions.

The End