Error Analysis of Deep Neural Networks

Shijun Zhang*

November 12, 2021

Given a target function f defined on a domain $\mathcal{X}$, our goal is to find a function h in the hypothesis space $\mathcal{H}$ to approximate f well. To make the hypothesis space $\mathcal{H}$ simple and computable, we generally use deep neural networks to realize its elements. In our current papers [1, 2, 3, 4, 5, 6, 7, 8], we only consider the fully-connected neural network and let us briefly introduce its architecture. An fully-connected neural network with a vector input $\boldsymbol{x} \in \mathbb{R}^d$ and an output $h(\boldsymbol{x})$ can be represented in a form of function compositions as follows:

$$h(\boldsymbol{x}) = \mathcal{L}_L \circ \sigma \circ \quad \cdots \quad \circ \sigma \circ \mathcal{L}_1 \circ \sigma \circ \mathcal{L}_0(\boldsymbol{x}) \quad \text{for any } \boldsymbol{x} \in \mathbb{R}^d, \tag{1}$$

where σ is a univariate activation function acting on a vector element-wisely, and $\mathcal{L}_i$ is an in affine linear map given by $\mathcal{L}_i(\boldsymbol{y}) = \boldsymbol{A}_i\boldsymbol{y} + \boldsymbol{b}_i$, where $\boldsymbol{A}_i$ and $\boldsymbol{b}_i$ are the weight matrix and the bias vector for each $i \in \{0, 1, \cdots, L\}$, respectively. See Figure 1 for an example. In our

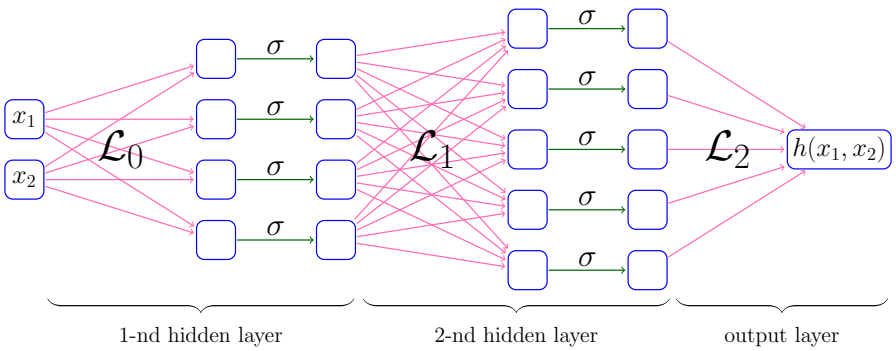

Figure 1: An example of a σ-activated network of width 5 and depth 2.

setting, the width and depth of networks are the maximum width of all hidden layers and the number of hidden layers, respectively.

For finitely many available samples $\big\{\big(\boldsymbol{x}_i, f(\boldsymbol{x}_i)\big)\big\}_{i=1}^n$, deep learning algorithms aim to identify the empirical risk minimizer $h_{\mathcal{S}}$ to approximate f well for the purpose of inferring the sample $\big(\boldsymbol{x}, f(\boldsymbol{x})\big)$ that is not available, where $h_{\mathcal{S}}$ is given by

$$h_{\mathcal{S}} \in \arg\min_{h\in\mathcal{H}} R_{\mathcal{S}}(h), \quad \text{where } R_{\mathcal{S}}(h) := \frac{1}{n}\sum_{i=1}^{n} \ell\big(h(\boldsymbol{x}_i), f(\boldsymbol{x}_i)\big) \tag{2}$$

with a loss function $\ell(\cdot,\cdot)$ typically taken as $\ell(y, y') = \frac{1}{2}|y - y'|^2$.

*Department of Mathematics, National University of Singapore (zhangshijun@u.nus.edu).

In fact, the best network-generated function in $\mathcal{H}$ to infer $f(\boldsymbol{x})$ is the expected risk minimizer $h_{\mathcal{D}}(\boldsymbol{x})$, but not the empirical risk minimizer $h_{\mathcal{S}}(\boldsymbol{x})$, where $h_{\mathcal{D}}(\boldsymbol{x})$ is given by

$$h_{\mathcal{D}} \in \arg\min_{h\in\mathcal{H}} R_{\mathcal{D}}(h), \quad \text{where } R_{\mathcal{D}}(h) := \mathbb{E}_{\boldsymbol{x}\sim U(\mathcal{X})}\left[\ell\big(h(\boldsymbol{x}), f(\boldsymbol{x})\big)\right],$$

where U is a data distribution over $\mathcal{X}$. Thus, the best possible inference error is $R_{\mathcal{D}}(h_{\mathcal{D}})$. In real applications, $U(\mathcal{X})$ is unknown and only finitely many samples from this distribution are available. Hence, the empirical risk $R_{\mathcal{S}}(h)$ is minimized, hoping to obtain $h_{\mathcal{S}}$, instead of minimizing the expected risk $R_{\mathcal{D}}(h)$ to obtain $h_{\mathcal{D}}$. In practice, a numerical optimization method to solve (2) may result in a numerical solution (denoted as $h_{\mathcal{N}}$) that may not be a global minimizer $h_{\mathcal{S}}$. Therefore, the actually learned network-generated function to infer $f(\boldsymbol{x})$ is $h_{\mathcal{N}}(\boldsymbol{x})$ and the corresponding inference error is measured by $R_{\mathcal{D}}(h_{\mathcal{N}})$, which is bounded by

$$\begin{aligned} R_{\mathcal{D}}(h_{\mathcal{N}}) &= \underbrace{[R_{\mathcal{D}}(h_{\mathcal{N}}) - R_{\mathcal{S}}(h_{\mathcal{N}})]}_{\text{GE}} + \underbrace{[R_{\mathcal{S}}(h_{\mathcal{N}}) - R_{\mathcal{S}}(h_{\mathcal{S}})]}_{\text{OE}} + \underbrace{[R_{\mathcal{S}}(h_{\mathcal{S}}) - R_{\mathcal{S}}(h_{\mathcal{D}})]}_{\leq 0 \text{ by (2)}} + \underbrace{[R_{\mathcal{S}}(h_{\mathcal{D}}) - R_{\mathcal{D}}(h_{\mathcal{D}})]}_{\text{GE}} + \underbrace{R_{\mathcal{D}}(h_{\mathcal{D}})}_{\text{AE}} \\ &\leq \underbrace{R_{\mathcal{D}}(h_{\mathcal{D}})}_{\text{Approximation error (AE)}} + \underbrace{[R_{\mathcal{S}}(h_{\mathcal{N}}) - R_{\mathcal{S}}(h_{\mathcal{S}})]}_{\text{Optimization error (OE)}} + \underbrace{[R_{\mathcal{D}}(h_{\mathcal{N}}) - R_{\mathcal{S}}(h_{\mathcal{N}})] + [R_{\mathcal{S}}(h_{\mathcal{D}}) - R_{\mathcal{D}}(h_{\mathcal{D}})]}_{\text{Generalization error (GE)}}. \end{aligned}$$

See Figure 2 for an illustration of the three errors in the equation above.

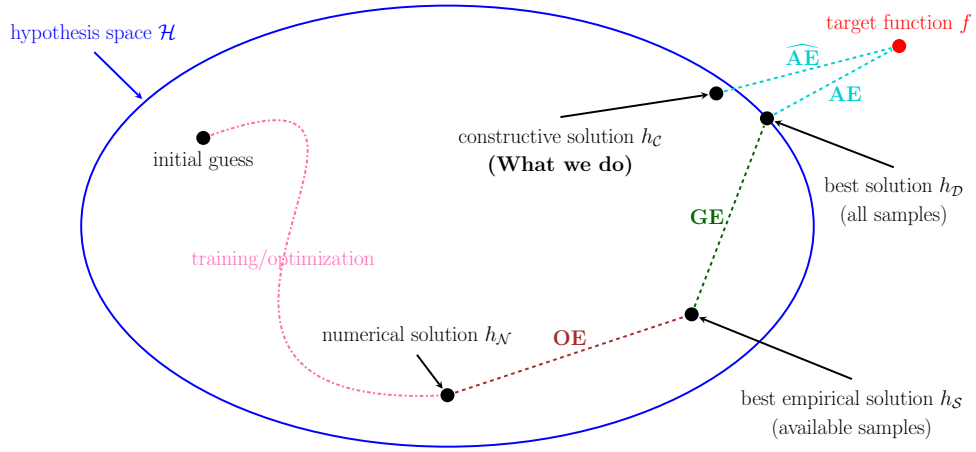

Figure 2: An illustration of the approximation error (AE), the generalization error (GE), and the optimization error (OE). $\widehat{\text{AE}}$ is an upper bound of AE.

As shown in Figure 2, we need to estimate the three errors to bound the distance between the target function f and the numerically learned function $h_{\mathcal{N}}$. Most of our papers aim to construct a solution $h_{\mathcal{C}}$ to provide an upper bound of the approximation error. The construction of $h_{\mathcal{C}}$ is independent of the empirical risk minimization in (2) and the optimization algorithm used to compute the numerical solution. Designing efficient algorithms to control the optimization error and analyzing the generalization error are two other separate future directions.

References

[1] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Nonlinear approximation via compositions. *Neural Networks*, 119:74–84, 2019.

[2] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation characterized by number of neurons. *Communications in Computational Physics*, 28(5):1768–1811, 2020.

[3] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network with approximation error being reciprocal of width to power of square root of depth. *Neural Computation*, 33(4):1005–1036, 03 2021.

[4] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Neural network approximation: Three hidden layers are enough. *Neural Networks*, 141:160–173, 2021.

[5] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021.

[6] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Optimal approximation rate of ReLU networks in terms of width and depth. *Journal de Mathématiques Pures et Appliquées*, to appear.

[7] Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation: Achieving arbitrary accuracy with fixed number of neurons. *arXiv e-prints*, page arXiv:2107.02397, July 2021.

[8] Shijun Zhang. Deep neural network approximation via function compositions. *PhD Thesis, National University of Singapore*, 2020. URL: https://scholarbank.nus.edu.sg/handle/10635/186064.