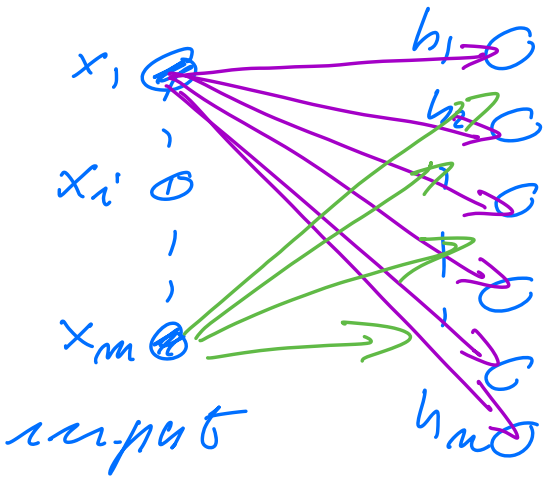
Lecture October 15

Typical NN

- m input values
- First hidden layer
  n - nodes / neurons



weight matrix

$W \in \mathbb{R}^{m \times n}$

$b \in \mathbb{R}^{n}$

$$z(x) = W^T x + b$$

$$x \mapsto \sigma(z(x))$$

σ: activation function

$$= [a_1(x), a_2(x), \ldots a_n(x)]$$

With L- layers-

$z_1 : \mathbb{R}^m \to \mathbb{R}^n$

$z_\ell : \mathbb{R}^{m_{\ell-1}} \to \mathbb{R}^{m_\ell}$ for $2 \le \ell \le L$

(L → output layer)

our model

$$\Theta = \{W, b\}$$

$$f(x;\Theta) = \sigma_L(z_L(\ldots \sigma_1(z_1(x))))$$

Math of activation functions-

consider a simple NN where x, w, b are scalar quantities. $L = 2$

$$f(x;\Theta) = \sigma_2(w_2 \sigma_1(w_1 x + b_1) + b_2)$$

in Backpropagation + gradient optimization

$\partial_{w_1} f(x;\theta)$, $\partial_{w_2} f(x;\theta)$

↓

$$\partial_{w_1} f(x;\theta) = \sigma_2'\left(w_2 \sigma_1\left(\overbrace{w_1 x + b_1}^{z_1}\right) + b_2\right) \cdot$$

$$\frac{\partial f}{\partial w_1} \qquad \otimes \underline{w_2}\, \sigma_1'(w_1 x + b_1) x$$

L - layers

$$\partial_{w_1} f(x;\theta) = \left[\prod_{\ell=2}^{L} w_\ell\right] \otimes \left[\prod_{\ell=1}^{L} \sigma_\ell'(z_\ell)\right] x$$

standard activation functions

$\sigma(z) = \dfrac{1}{1+e^{-z}}$ sigmoid

tanh

when $|z_e| \gg 0$, then

$\sigma'(z)$ can become small.

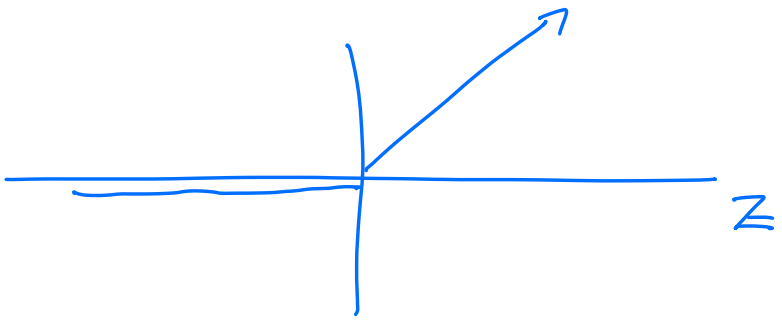
$\Rightarrow$ vanishing gradients

$$RELU = \sigma(z) = \max(0, z)$$

$$\sigma'(z) = 1 \quad \text{for} \quad z > 0$$

$$\text{Leaky ReLU} = \sigma(z) = \begin{cases} \alpha \cdot z & z < 0 \\ z & z \geq 0 \end{cases}$$

$\alpha \sim 10^{-2} - 10^{-3}$



$$\sigma'(z) = \begin{cases} \alpha & \text{for } z < 0 \\ 1 & \text{for } z \geq 0 \end{cases}$$

$$\sigma(z) = ELU = \begin{cases} \alpha(e^z - 1) & z < 0 \end{cases}$$

$$\sigma'(z) = \begin{cases} \alpha e^{z} & z < 0 \\ 1 & z \geq 0 \end{cases}$$

$$\sigma(z) = \tanh(z)$$

$$\sigma'(z) = 1 - (\tanh(z))^2$$

$$\sigma(z) = \text{sigmoid} = \frac{1}{1+e^{-z}}$$

$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

Classical approximations

$y = F(x)$ (continuous)

$x \in [0,1]^d$

Given $F(x)$ and $\varepsilon > 0$ there exists a function

$f(x;\theta)$

$|F(x) - f(x;\theta)| < \varepsilon$

for all $x \in [0,1]^d$

Examples

$$f(x;\theta) = \sum_j \theta_j \cdot x^j$$

Theorem: Stone-Weierstrass

$F \in \mathbb{C}[0,1]^d$, for each ε there exists a polynomial

$$|F(x) - f(x;\theta)| < \varepsilon$$

$\forall \; x \in [0,1]^d$

Example:

$$f(x;\theta) = \sum_j \theta(\beta_j) e^{i 2\pi x \beta_j}$$

$$\left[\int (F(x) - f(x;\theta))^2 dx \right]^{1/2}$$

$$\left| \quad I_{[0,1]^d} \quad \right| < \varepsilon$$

For NNs [Cybenko, 1989]

- Let σ be any continuous sigmoidal function

$$\sigma(z) \to \begin{cases} 1 & \text{as } z \to \infty \\ 0 & \text{as } z \to -\infty \end{cases}$$

Given an $F \in C([0,1]^d)$ and $\varepsilon > 0$, there is a one layer neural network

$$f(x; W, b) = f(x; \theta)$$

with $W \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ for which

$$| f(x; \theta) - F(x) | < \varepsilon$$

for all $x \in [0,1]^d$

or:

Any continuous $F(x)$ for $[0,1]^d$ can be approximated by a one layer sigmoidal network to arbitrary accuracy.

Called Universal approximation theorem

Hornik (1991) refined the theorem by letting any non-constant bounded function to be included.

- The theorems do not

say anything about the number of nodes or the values of the weights and biases

- does not mean that any NN can be used to compute exactly any function,

  We get an approximation that is as good as we want.

- The $F(x)$ functions are continuous functions.