Comp Sci, January 24, 2023

NN architecture (model)

- # hidden layers
- # hidden neurons in a layer
- activation function (Sigmoid, tanh, ReLU, ELU, ...)

cost function & optimization

- Type of cost/loss
- Regularization, l1, l2
- GD, stochastic, batches, learning rate, epochs
  - optimization schemes

other specifics; pretraining, initialization, dropout, ...

Universal approximation theorem:

$y = F(x) \qquad x \in [0,1]^d$

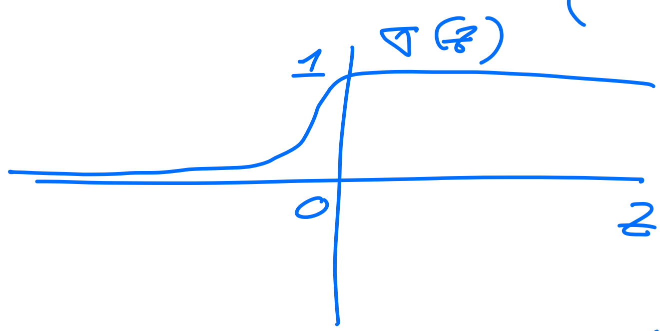
deterministic function assumed to be continuous

$$F \in \mathbb{C}\,[0,1]^d$$

(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 a function $F \in [0,1]^d$ and $\varepsilon > 0$, there is a one-layer NN $f(x;\Theta)$ of the form $W \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^n$ for which

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

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

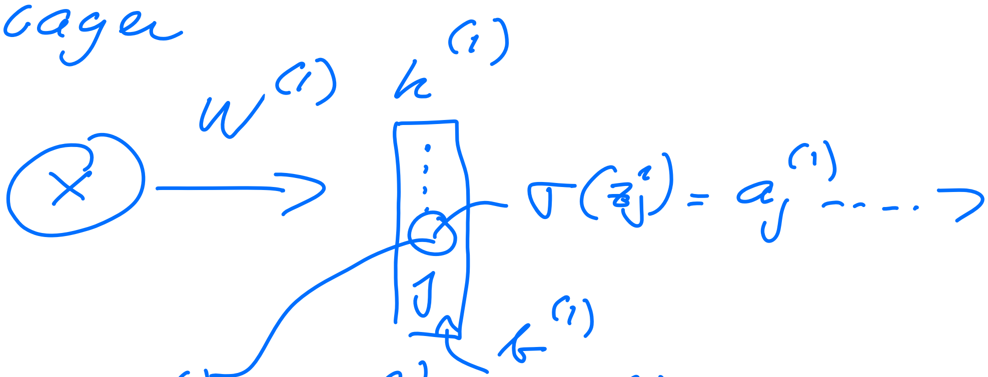
for all $x \in [0,1]^d$

Hornik (1991) extended the theorem to apply to any non-constant, bounded activation function.

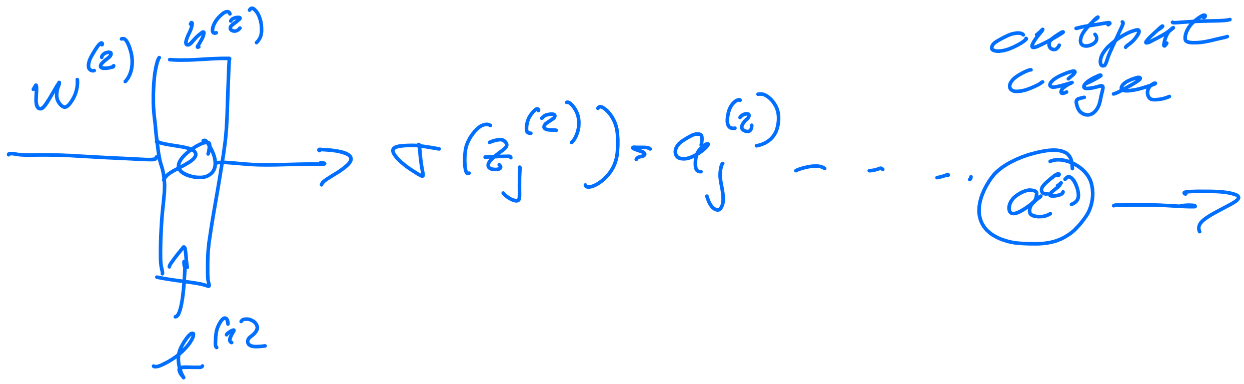
Basics of an NN

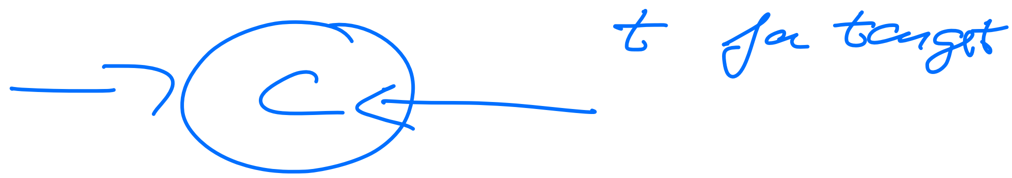
— Feed Forward stage

input layer hidden layers



$$z_j^{(1)} = W^{(1)} X + b^{(1)}$$





- Back propagation stage

Final output layer

$$a^{(L)}(x;\theta)$$

$$\theta = \{ w^{(1)}, b^{(1)}, w^{(2)}, b^{(2)}, \dots, w^{(L)}, b^{(L)} \}$$

MSE

$$C(\theta) = \frac{1}{n} \| (t - a^{L}(x;\theta)) \|_2^2$$

$$a^{(L)}(x;\theta) = \sigma^{(L)}(\sigma^{(L-1)}(\sigma^{(L-2)}(\dots \sigma^{(1)}(xW^{(1)} + b^{(1)})) \dots))$$

Back propagation algo

- problems with vanishing gradient (or exploding)

Consider a simple NN in which x, w, b are all scalars and reals.

$L = 2$

$$f(x;\theta) = \sigma_2\left(w_2\,\sigma_1(w_1 x + b_1) + b_2\right)$$

$$\partial_{w_1} f(x;\theta) = \sigma_2'\left(w_2\,\sigma_1(w_1 x + b_1) + b_2\right) \times \left(w_2\,\sigma_1'(w_1 x + b_1)\,x\right)$$

with L-layers

$$\left[\prod_{\ell=2}^{L} w_\ell\right] \times \left[\prod_{\ell=1}^{L} \sigma_\ell'(z_\ell)\right] \times$$

$$z_\ell = A_\ell\left[\sigma_{\ell-1}\left(A_{\ell-1}\left(\cdots \sigma_1\left(A_1(x)\right)\cdots\right)\right]$$

if σ_ℓ is a sigmoid function (or tanh) then $\sigma_\ell'(z_\ell)$ will be small when $|z_\ell| \gg 0$

Derivation of Back prop algo

- analytical expressions for $\frac{\partial C}{\partial w^{(\ell)}}$ & $\frac{\partial C}{\partial b^{(\ell)}}$

- then we can update

$$w^{(\ell)} \leftarrow w^{(\ell)} - \eta^{(\ell)} \frac{\partial C}{\partial w^{(\ell)}}$$

$$b^{(\ell)} \leftarrow b^{(\ell)} - \eta^{(\ell)} \frac{\partial C}{\partial b^{(\ell)}}$$

input to node j in layer ℓ

$$z_j^\ell = \sum_{i=1}^{N_{\ell-1}} w_{ji}^\ell a_i^{\ell-1} + b_j^\ell$$

$$a_i^{\ell-1} = \sigma(z_i^{\ell-1})$$

$$\frac{\partial z_j^\ell}{\partial w_{ij}^\ell} = a_i^{\ell-1}$$

$$\frac{\partial z_j^\ell}{\partial a_i^{\ell-1}} = w_{ij}^\ell$$

$$\frac{\partial a_j^\ell}{\partial z_j^\ell} = \ ?$$

assume $a_j^\ell = \sigma^\ell(z_j^\ell)$

$$= \frac{1}{1+e^{-z_j^\ell}}$$

$$\frac{\partial a_j^\ell}{\partial z_j^\ell} = \sigma(z_j^\ell)\left(1-\sigma(z_j^\ell)\right)$$

$$= a_j^l (1 - a_j^l)$$

Computational note:
these derivatives will change when we change activation function

Example of $C(\theta)$

$$C(\theta) = \frac{1}{2} \sum_i (a_i^L - t_i)^2$$

$$\frac{\partial C(\theta)}{\partial w_{jk}^L} \quad , \quad \frac{\partial C}{\partial b^L}$$

$$\frac{\partial C}{\partial w_{jk}^L} = (a_j^L - t_i) \frac{\partial a_j^L}{\partial w_{jk}^L}$$

chain rule:

$$\frac{\partial a_j^L}{\partial w_{jk}^L} = \frac{\partial a_j^L}{\partial z_j^L} \frac{\partial z_j^L}{\partial w_{jk}^L}$$

with sigmoid for $\sigma(z)$

$$\frac{\partial a_j^L}{\partial w_{jk}^L} = \underbrace{a_j^L(1-a_j^L)}_{\sigma'(z)} a_k^{L-1}$$

$$\frac{\partial C}{\partial w_{jk}^L} = (a_j^L - t_j) a_j^L (1-a_j^L) a_k^{L-1}$$

$$= (a_j^L - t_j)\, \sigma'(z_j^L)\, a_k^{L-1}$$

$$\delta_j^L = a_j^L(1-a_j^L)(a_j^L - t_j)$$

$$= \sigma'(z_j^L) \frac{\partial C}{\partial a_j^L}$$

$$\boxed{\frac{\partial C}{\partial w_{jk}^L} = \delta_j^L a_k^{L-1}}$$

$$\delta_j^L = \frac{\partial C}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L}$$

$$\delta_j^L = \frac{\partial C}{\partial b_j^L}\frac{\partial b_j^L}{\partial z_j^L} = \frac{\partial C}{\partial b_j^L}$$

Collect for $l = L$

$$\boxed{\begin{aligned} \frac{\partial C}{\partial w_{jk}^L} &= \delta_j^L a_k^{L-1} \\ \delta_j^L &= \sigma'(z_j^L)\frac{\partial C}{\partial a_j^L} \\ \delta_j^L &= \frac{\partial C}{\partial b_j^L} \end{aligned}}$$

$L \Rightarrow l \ (\neq L)$

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_k \frac{\partial C}{\partial z_k^{l+1}}\frac{\partial z^{l+1}}{\partial z_j^l}$$

$$= \sum_k \delta_k^{l+1}\frac{\partial z_k^{l+1}}{\partial z_j^l}$$

$$z_j^{l+1} = \sum_{i=1}^{N_l} w_{ij}^{l+1} \underbrace{a_i^l}_{= \sigma(z_j^l)} + b_j^{l+1}$$

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1} \sigma'(z_j^l)$$

$$\boxed{\delta_j^l = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \sigma'(z_j^l)}$$

Final algo

(i) initialize $\theta = \{w, b\}$

(ii) Perform the 1st Feed forward pass

(iii) a_i^L is used to compute $C(\theta)$

(iv) Perform Back prop

for $l = L-1, L-2, \ldots 2$

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \sigma'(z_j^l)$$

$$w_{jk}^l \leftarrow w_{jk}^l - \eta\, \delta_j^l a_k^{l-1}$$

$$b_j^l \leftarrow b_j^l - \eta \frac{\partial C}{\partial b_j^l} = b_j^l - \eta\, \delta_j^l$$

learning

end for

(v) repeat (iii) – (iv) till $C(\theta)$ stabilizes.

output $\theta = \{W, b\}$, which gives us the optimal description of $F(x)$

Automatic differentiation

JAX replaces autograd.

Example

$$f(x) = \exp(x^2)$$
$$f'(x) = 2x\exp(x^2)$$

Def $\quad a = x^2$

$\quad\quad b = \exp(a) = f(x)$

$$x \xrightarrow{\;x^2\;} a \xrightarrow{\;\exp(x^2)\;} b \longrightarrow f$$

$$\frac{df}{dx} = \left[\underbrace{\frac{df}{db}}_{1}\ \frac{db}{da}\right]\frac{da}{dx}$$

Reverse mode

$$= \frac{df}{db}\left[\frac{db}{da}\ \frac{da}{dx}\right]$$

Forward mode

New example

$$f(x) = \sqrt{x^2 + \exp(x^2)}$$

$$a = x^2 \; ; \quad b = \exp(a)$$

$$c = a + b$$

$$d = \sqrt{c} = f(x)$$

$$\frac{df}{dx} = \frac{x\,(1 + \exp(x^2))}{\sqrt{x^2 + \exp(x^2)}}$$

$$= \frac{x\,(1+b)}{\sqrt{a+b}} = \frac{x(1+b)}{\sqrt{c}}$$

$$= \frac{x\,(1+b)}{d}$$

graphically:

$$\begin{tikzcd}
 & & b \arrow[d] & & \\
x \arrow[r, "x^2"] & a \arrow[ur, bend left, "\exp(x^2)"] \arrow[r] & \boxed{+} \arrow[r] & c \arrow[r, "\sqrt{\ }"] & d \arrow[r] & {}
\end{tikzcd}$$

$$\rightarrow f \qquad f(x) = d$$

$$\frac{\partial a}{\partial x} = 2x \qquad \frac{\partial b}{\partial x} = \frac{\partial b}{\partial a}\frac{\partial a}{\partial x}$$

$$\frac{\partial c}{\partial x} = \left[\frac{\partial c}{\partial a}\frac{\partial a}{\partial x} + \frac{\partial c}{\partial b}\frac{\partial b}{\partial x}\right]$$

$$\frac{\partial d}{\partial c} = \frac{1}{2\sqrt{c}}$$

$$\frac{\partial d}{\partial x} = \frac{\partial d}{\partial c}\frac{\partial c}{\partial x} = \frac{\partial f}{\partial x}$$

Computation in reverse mode : More next week.