Foundations of Deep Learning

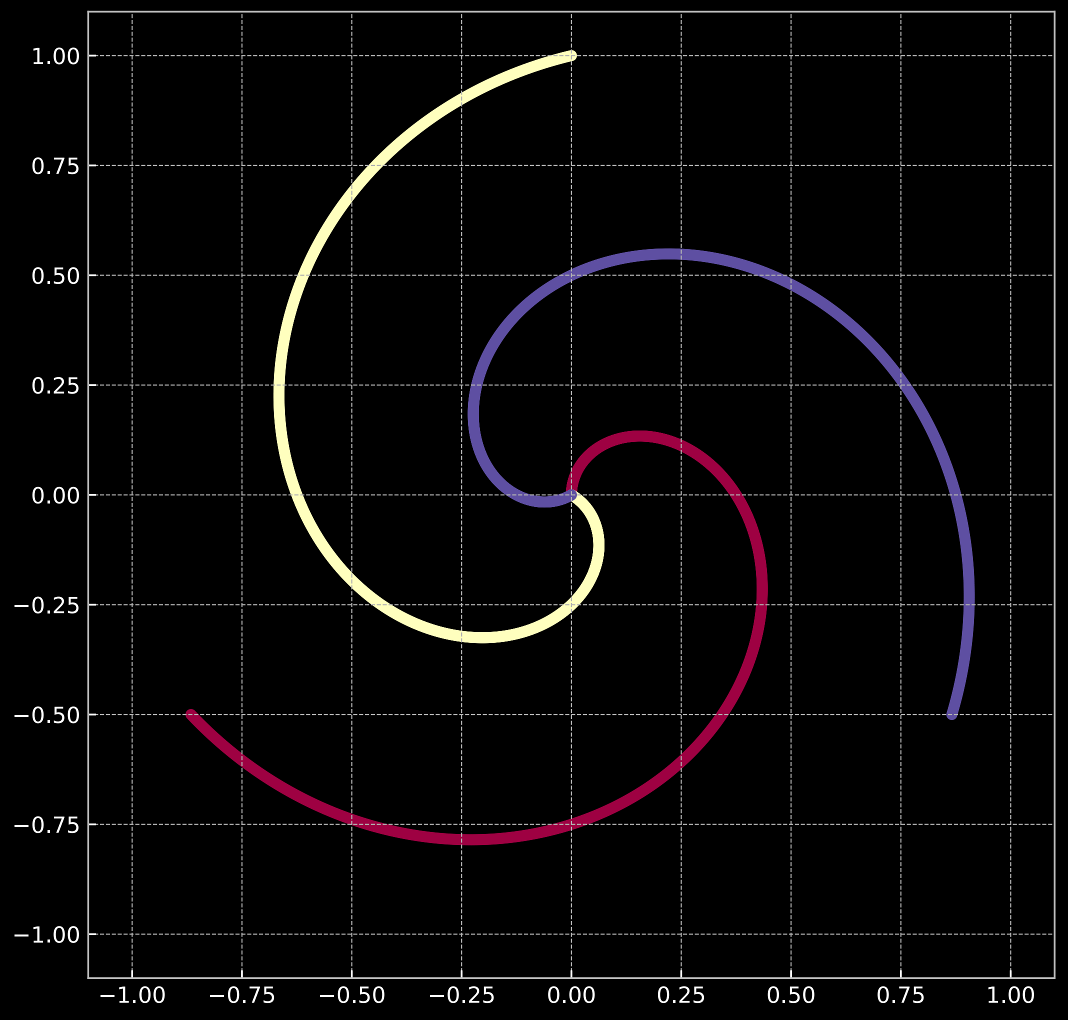
ALF

Alfredo Canziani

@alfcnz

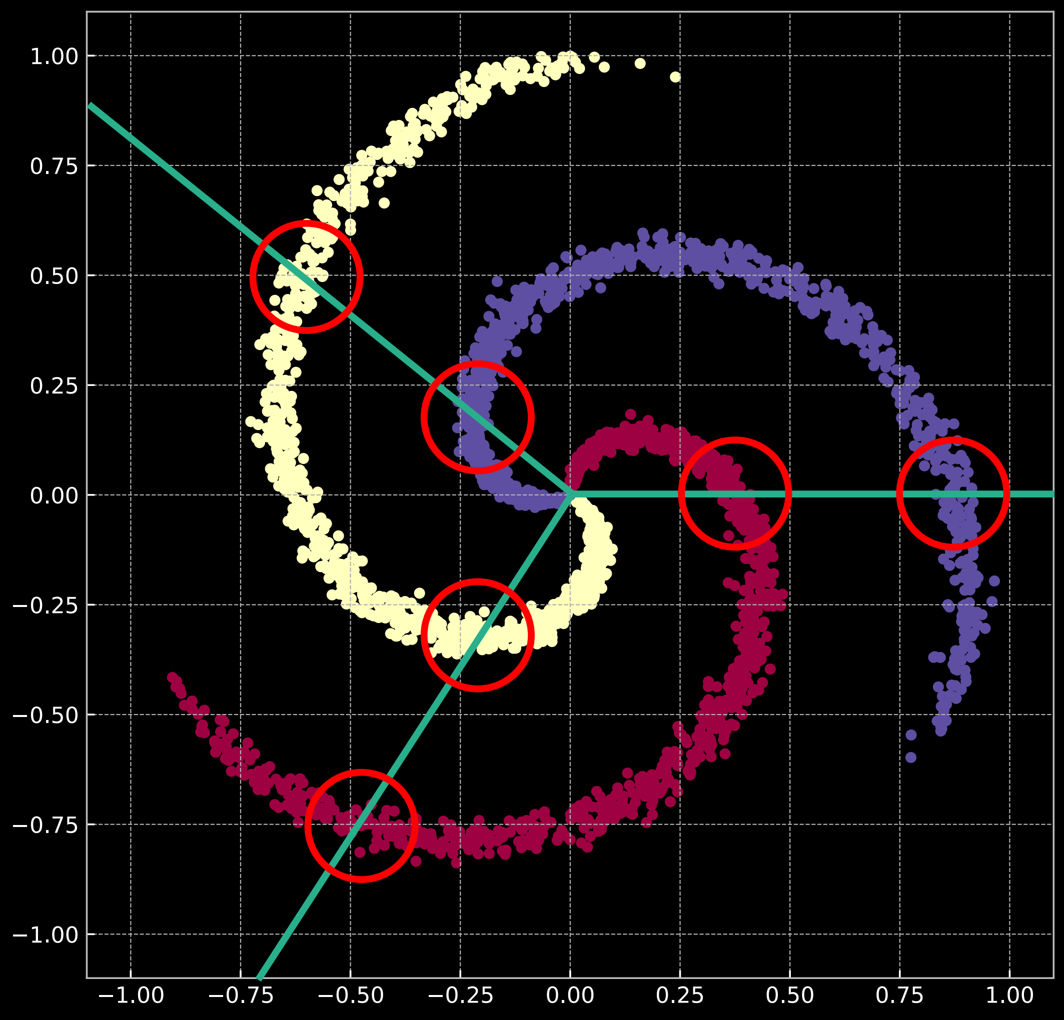
ANN – supervised learning

Classification



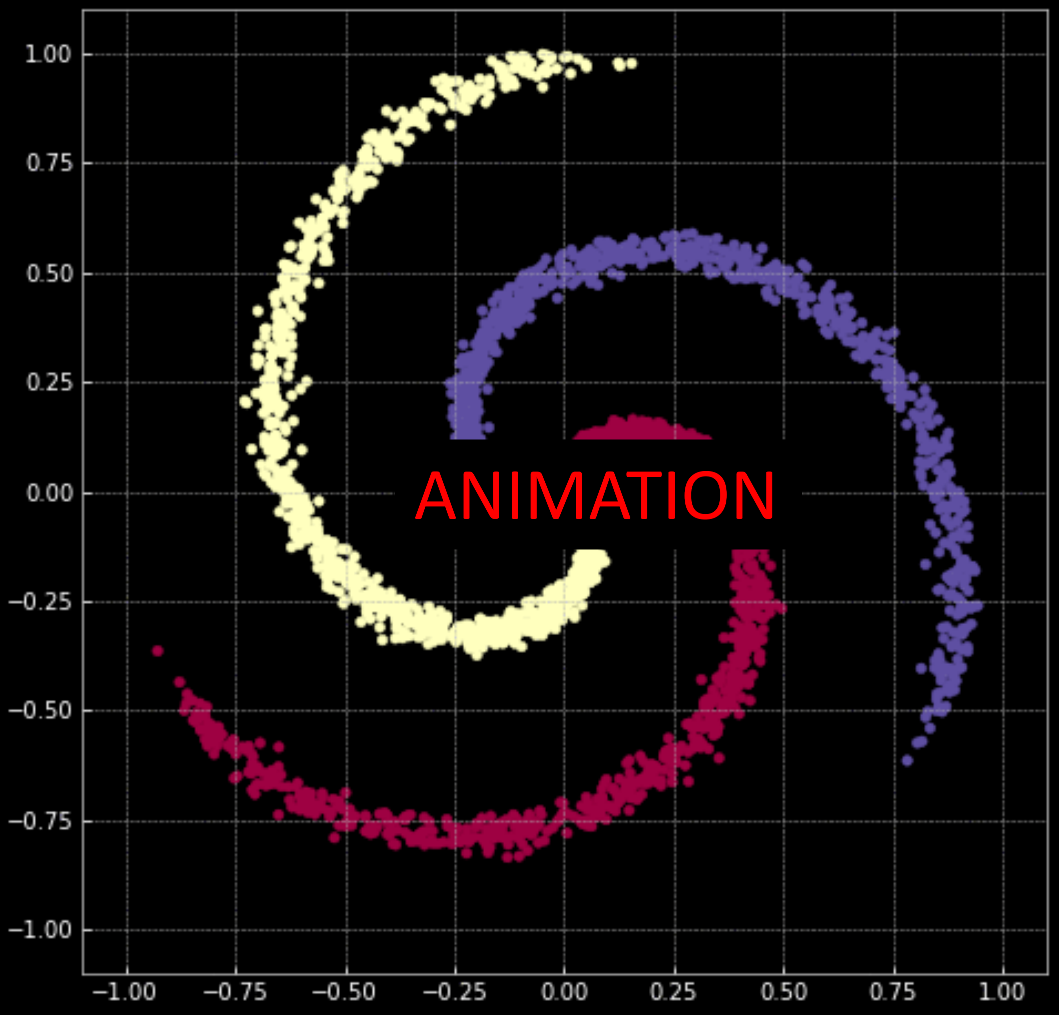
$$X_c(t) = t\begin{pmatrix}\sin\left(\frac{2\pi}{C}\left(2t+c-1\right)\right)\\ \cos\left(\frac{2\pi}{C}\left(2t+c-1\right)\right)\end{pmatrix}$$

$$0 \le t \le 1, \quad c = 1,\cdots,C$$

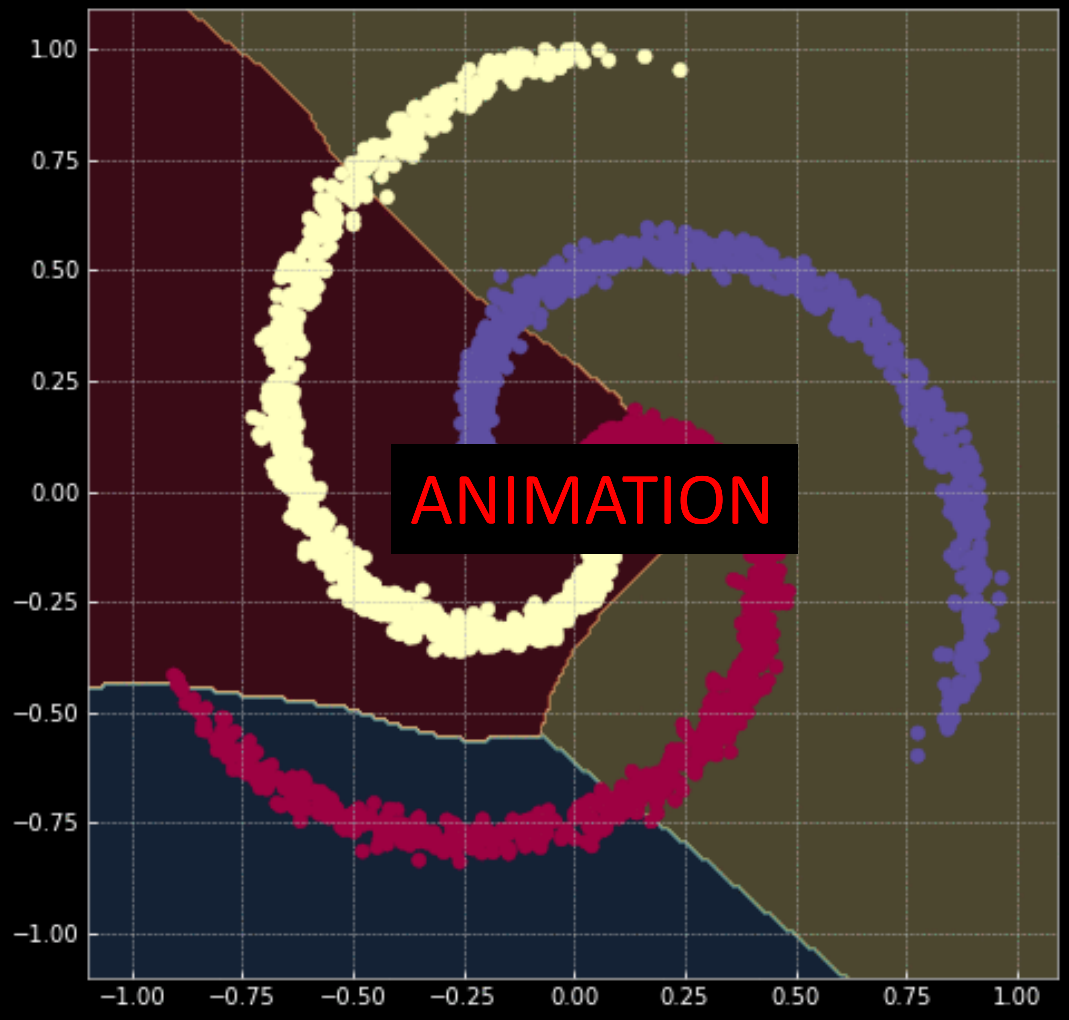


$$X_c(t) = t\begin{pmatrix}\sin\left(\frac{2\pi}{C}\left(2t+c-1\right)\right)\\ \cos\left(\frac{2\pi}{C}\left(2t+c-1\right)\right)\end{pmatrix}$$

$$\boxed{+\mathcal{N}(0,\sigma^2)}$$



ANIMATION



ANIMATION

Training data

$(0\ 0\ 1)$
$(0\ 1\ 0)$ 1-hot encoding
$(1\ 0\ 0)$

$$\begin{array}{cc} \xleftrightarrow{\qquad n \qquad} & \\ X = \begin{bmatrix} — \underline{x}^{(1)} — \\ — \underline{x}^{(2)} — \\ \vdots \\ — \underline{x}^{(m)} — \end{bmatrix} & \updownarrow m \end{array} \qquad \begin{array}{cc} \xleftrightarrow{\qquad G \qquad} & \\ Y = \begin{bmatrix} — \underline{y}^{(1)} — \\ — \underline{y}^{(2)} — \\ \vdots \\ — \underline{y}^{(m)} — \\ \uparrow \end{bmatrix} & \updownarrow m \end{array} \qquad \boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix} \in \mathbb{R}^m$$

$\boldsymbol{x}^{(i)} \in \mathbb{R}^n$, $n = 2$

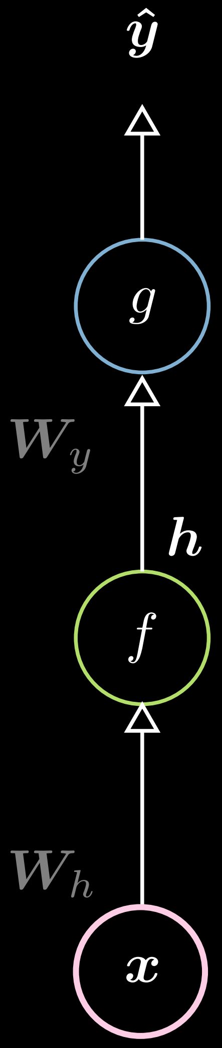
m : # samples in data set

$\boxed{\boldsymbol{y}^{(i)}} \in \mathbb{R}^G$, $G = 3$

$c_i \in \{1, \dots, G\} \subseteq \mathbb{N}^+$

Fully connected (FC) layer

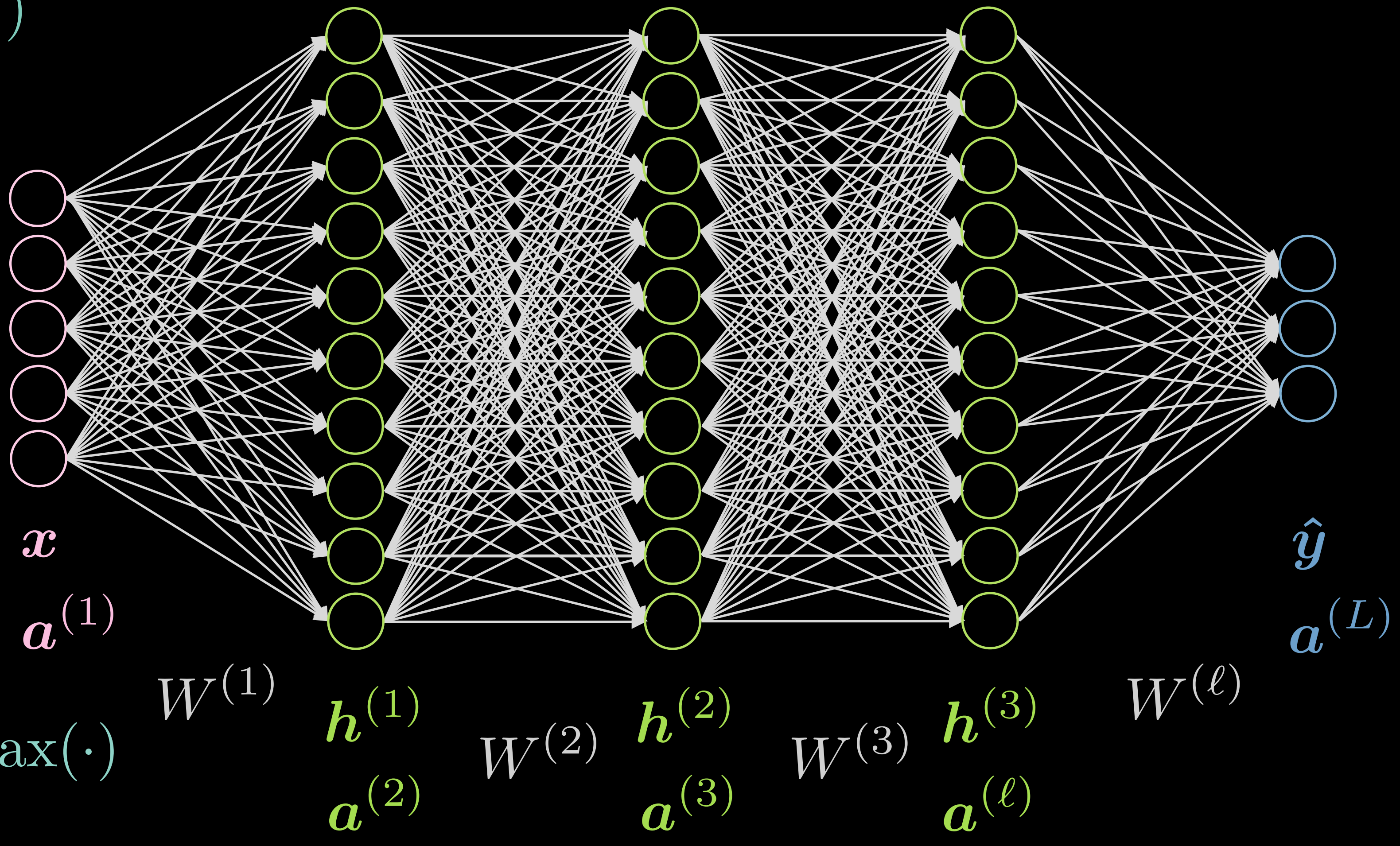
$$a_j^{(2)} = f(\overset{\text{j-th row of } W^{(1)}}{\boxed{\boldsymbol{w}^{(j)}}}\boldsymbol{x} + b_j) = f\Big(\big(\sum_{i=1}^{n} w_i^{(j)} x_i\big) + b_j\Big)$$



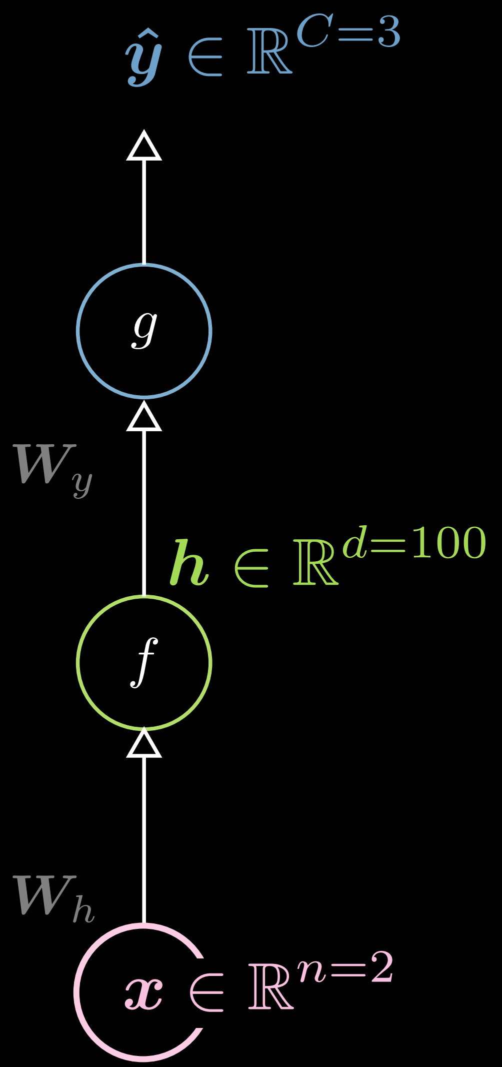
$$\boldsymbol{h} = f(\boldsymbol{W}_h\boldsymbol{x} + \boldsymbol{b}_h)$$

$$\hat{\boldsymbol{y}} = g(\boldsymbol{W}_y\boldsymbol{h} + \boldsymbol{b}_y)$$

$f, g = (\cdot)^+, \sigma(\cdot),$
$\tanh(\cdot), \mathrm{soft}(arg)\mathrm{max}(\cdot)$



Neural network (inference)



$\hat{\boldsymbol{y}} \in \mathbb{R}^{C=3}$

g

$\boldsymbol{W}_y$

$\boldsymbol{h} \in \mathbb{R}^{d=100}$

f

$\boldsymbol{W}_h$

$\boldsymbol{x} \in \mathbb{R}^{n=2}$

$$\boldsymbol{h} = f(\boldsymbol{W_h}\boldsymbol{x} + \boldsymbol{b_h})$$

$$\hat{\boldsymbol{y}} = g(\boldsymbol{W_y}\boldsymbol{h} + \boldsymbol{b_y})$$

$$\boldsymbol{W_h} \in \mathbb{R}^{d \times n}$$

$$\boldsymbol{b_h} \in \mathbb{R}^{d}$$

$$\boldsymbol{W_y} \in \mathbb{R}^{C \times d}$$

$$\boldsymbol{b_y} \in \mathbb{R}^{C}$$

$$f(\cdot), g(\cdot) : \mathrm{ReLU}(\cdot) \doteq (\cdot)^+, \sigma(\cdot), \tanh(\cdot), \mathrm{soft}\,(arg)\mathrm{max}(\cdot)$$

$$\hat{\boldsymbol{y}} = \hat{\boldsymbol{y}}(\boldsymbol{x}), \quad \hat{\boldsymbol{y}} : \mathbb{R}^n \to \mathbb{R}^C, \quad \boldsymbol{x} \mapsto \hat{\boldsymbol{y}}$$

$$\hat{\boldsymbol{y}} : \mathbb{R}^n \to \mathbb{R}^d \to \mathbb{R}^C, \quad d \gg n, C$$

$$\boldsymbol{h} = f(\boldsymbol{W_h x} + \boldsymbol{b_h})$$
$$\hat{\boldsymbol{y}} = g(\boldsymbol{W_y h} + \boldsymbol{b_y})$$

Neural network (training I)

logits: output of final layer

$$\mathrm{soft}(arg)\mathrm{max}(\boldsymbol{l})[c] \doteq \frac{\exp(\boldsymbol{l}[c])}{\sum_{j=1}^{C} \exp(\boldsymbol{l}[j])} \in (0,1)$$

$$\mathcal{L}(\hat{\boldsymbol{Y}}, \boldsymbol{c}) \doteq \frac{1}{m}\sum_{i=1}^{m} \ell(\hat{\boldsymbol{y}}^{(i)}, c^{(i)}), \quad \ell(\hat{\boldsymbol{y}}, c) \doteq -\log(\hat{\boldsymbol{y}}[c]) \longleftarrow$$

cross entropy / negative log-likelihood

$$\underline{x} \quad c = 1 \implies \underline{y} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\underline{\hat{y}}(\underline{x}) = \begin{pmatrix} \sim 1 \\ \sim 0 \\ \sim 0 \end{pmatrix} \implies \mathcal{L}\left(\begin{pmatrix} \sim 1 \\ \sim 0 \\ \sim 0 \end{pmatrix}, 1\right) \to 0^{+}$$

$$\underline{\hat{y}}(\underline{x}) = \begin{pmatrix} \sim 0 \\ \sim 1 \\ \sim 0 \end{pmatrix} \implies \mathcal{L}\left(\begin{pmatrix} \sim 0 \\ \sim 1 \\ \sim 0 \end{pmatrix}, 1\right) \to +\infty$$

$$\boldsymbol{h} = f(\boldsymbol{W_h x} + \boldsymbol{b_h})$$
$$\hat{\boldsymbol{y}} = g(\boldsymbol{W_y h} + \boldsymbol{b_y})$$

Neural network (training II)

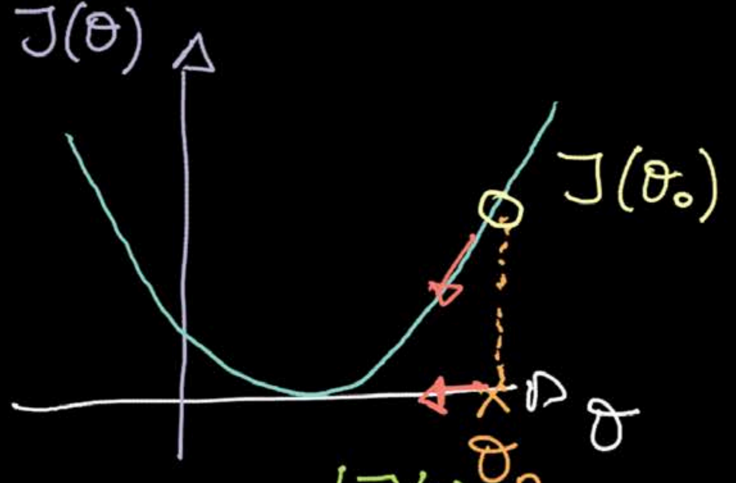
$$\Theta \equiv \{W_h, \boldsymbol{b}_h, W_y, \boldsymbol{b}_y\}$$

$$J(\Theta) \equiv \mathcal{L}(\hat{Y}(\Theta), \underline{c}) \in \mathbb{R}^+$$

$$\frac{\partial J(\Theta)}{\partial W_y} = \frac{\partial J(\Theta)}{\partial \underline{\hat{y}}} \frac{\partial \hat{y}}{\partial W_y}$$

$$\frac{\partial J(\Theta)}{\partial W_h} = \frac{\partial J(\Theta)}{\partial \underline{\hat{y}}} \frac{\partial \hat{y}}{\partial \underline{h}} \frac{\partial \underline{h}}{\partial W_h}$$

Back propagation



$$-\frac{dJ(\theta)}{d\theta}(\theta_0)$$

Gradient descent

Gradient computation example

```
x = torch.tensor([[1, 2], [3, 4]], requires_grad=True)
y = x - 2
z = y * y * 3
out = z.mean()
```

out: $o = \frac{1}{4} \sum_{i=1}^{4} z_i$

$\underline{z} = 3\underline{y}^2 = 3(\underline{x}-2)^2$

$$\frac{\partial o}{\partial x_i} = \frac{1}{\cancel{4}2} 3 \cdot \cancel{2} \, (x-2) = $$

$$= \frac{3}{2}(x-2)$$

$x = 1 \quad \frac{\partial o}{\partial x}\Big|_{x=1} = -1.5$

$\frac{\partial o}{\partial x}\Big|_{x=2} = 0 \quad \frac{\partial o}{\partial x}\Big|_{x=3} = 1.5 \quad \frac{\partial o}{\partial x}\Big|_{x=4} = 3$