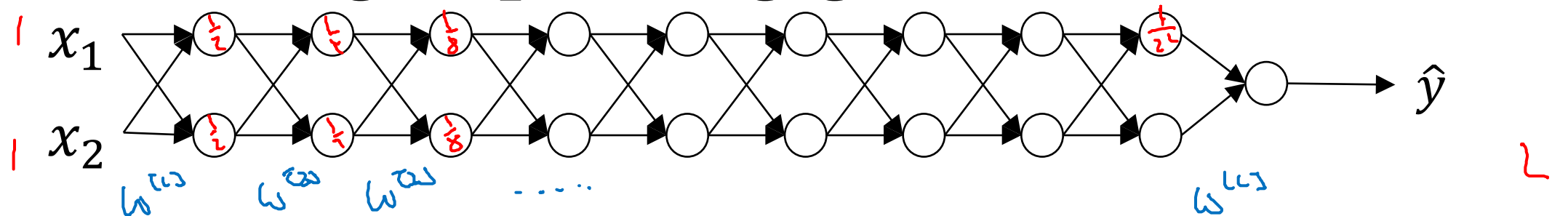deeplearning.ai

Setting up your optimization problem

Vanishing/exploding gradients

Vanishing/exploding gradients

$L = 150$

$g(z) = z.$ $b^{[l]} = 0.$

$\hat{y} = W^{[L]} \underbrace{W^{[L-1]} W^{[L-2]} \cdots} \boxed{W^{[3]} \boxed{W^{[2]} W^{[1]}} x} \qquad a^{[3]}$

1.5^L

0.5^L

$W^{[l]} > I$

$W^{[l]} < I \quad \begin{bmatrix} 0.9 & \\ & 0.9 \end{bmatrix}$

$z^{[1]} = W^{[1]} x$

$a^{[1]} = g(z^{[1]}) = z^{[1]}$

$a^{[2]} = g(z^{[2]}) = g(W^{[2]} a^{[1]})$

$W^{[l]} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}$ 0.5   0.5

$\hat{y} = W^{[l]} \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}^{L-1} x$ 0.5 0.9

$1.5^{L-1} x$

$0.5^{L-1} x$

Andrew Ng

Single neuron example

$a^{[1]}$

x_1

x_2

x_3

x_4

$\hat{y}$

$a = g(z)$

$W^{[l]}$
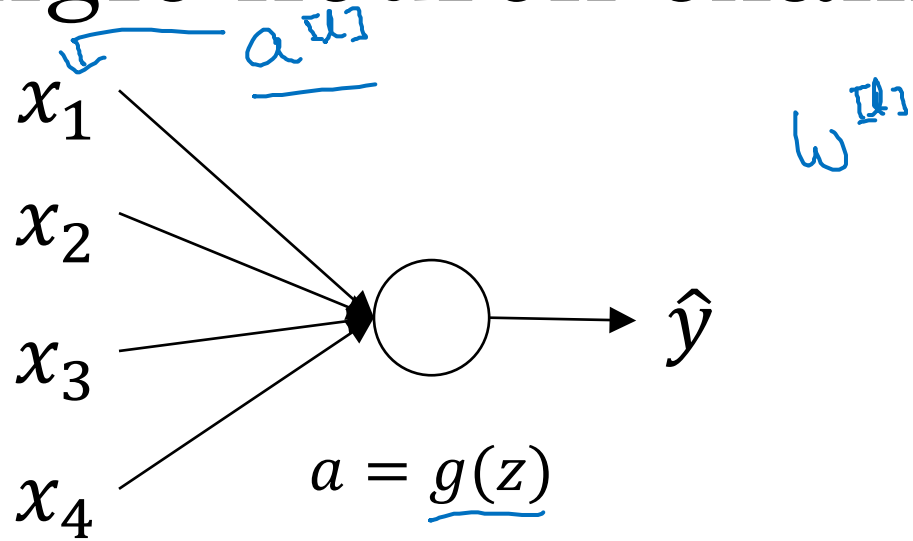
$z = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$

Large $n \rightarrow$ Smaller w_i

$$Var(w_i) = \frac{1}{n} \frac{2}{n}$$

$W^{[l]} = np.random.randn(shape..) * np.sqrt\left(\frac{2}{n^{[l-1]}}\right)$

ReLU

$g^{[l]}(z) = ReLU(z)$

Other variants:

tanh

$$\frac{1}{n^{[l-1]}}$$

Xavier initialization

$$\frac{2}{n^{[l-1]} + n^{[l]}}$$

Andrew Ng