



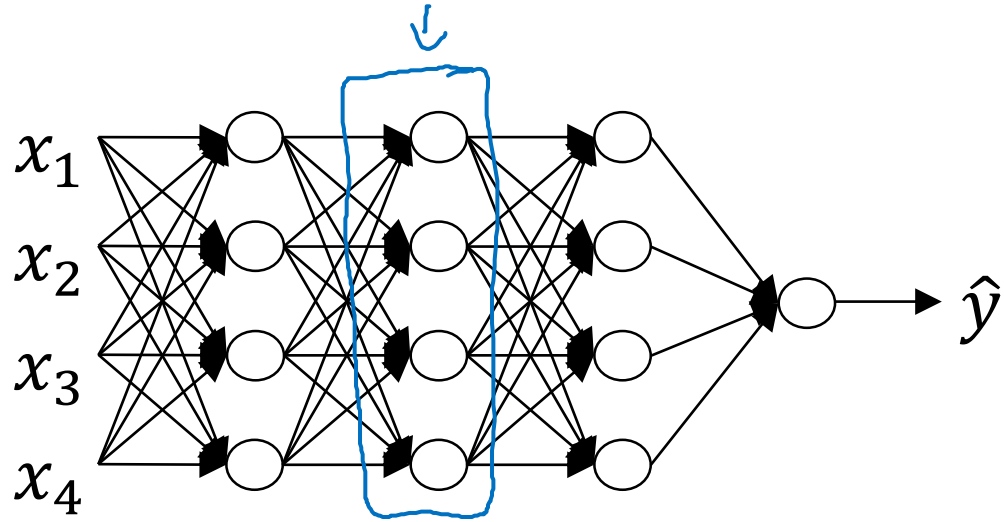
deeplearning.ai

# Deep Neural Networks

---

Building blocks of  
deep neural networks

# Forward and backward functions



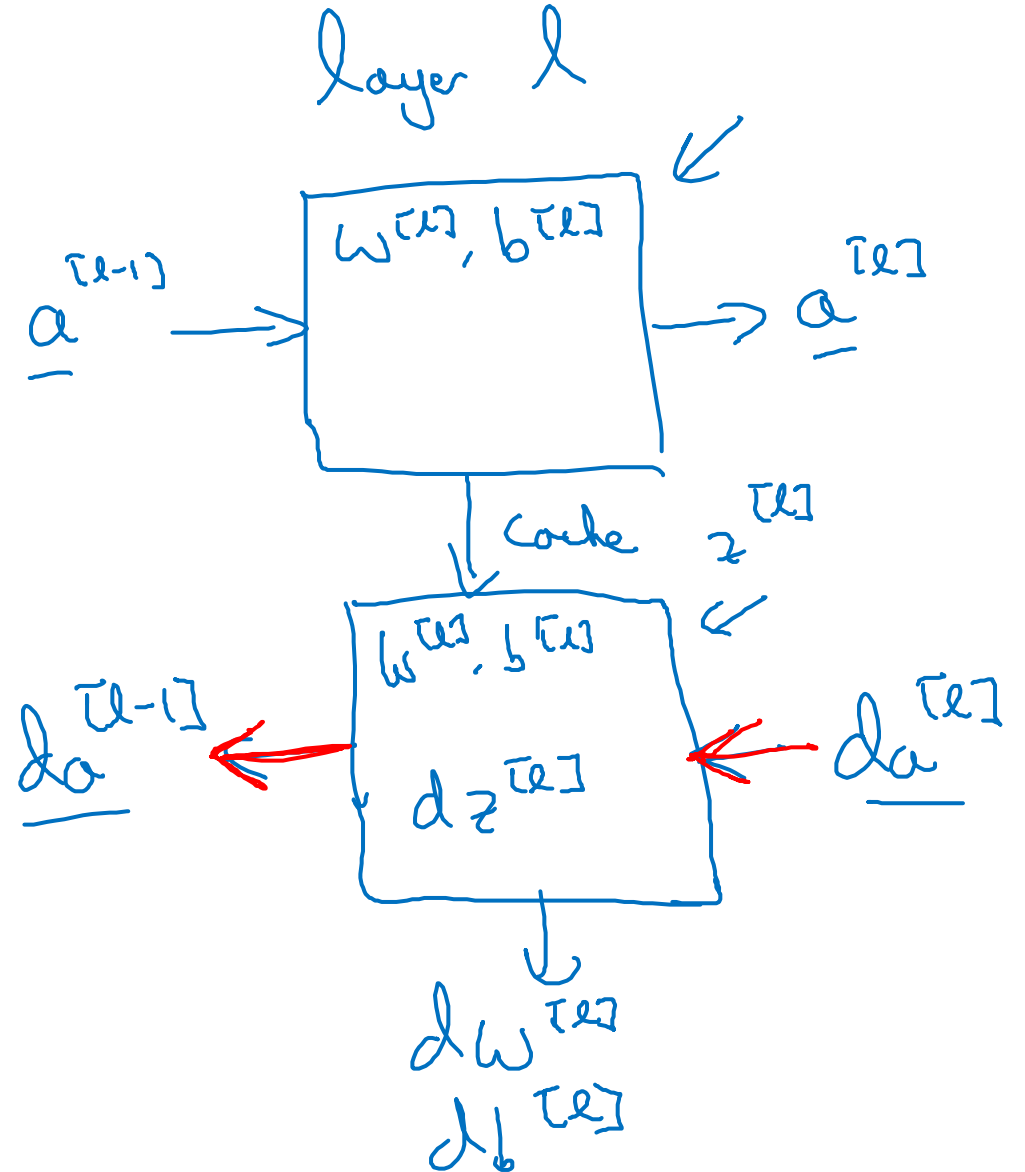
Layer  $l$ :  $W^{[l]}, b^{[l]}$

→ Forward: Input  $a^{[l-1]}$ , output  $a^{[l]}$

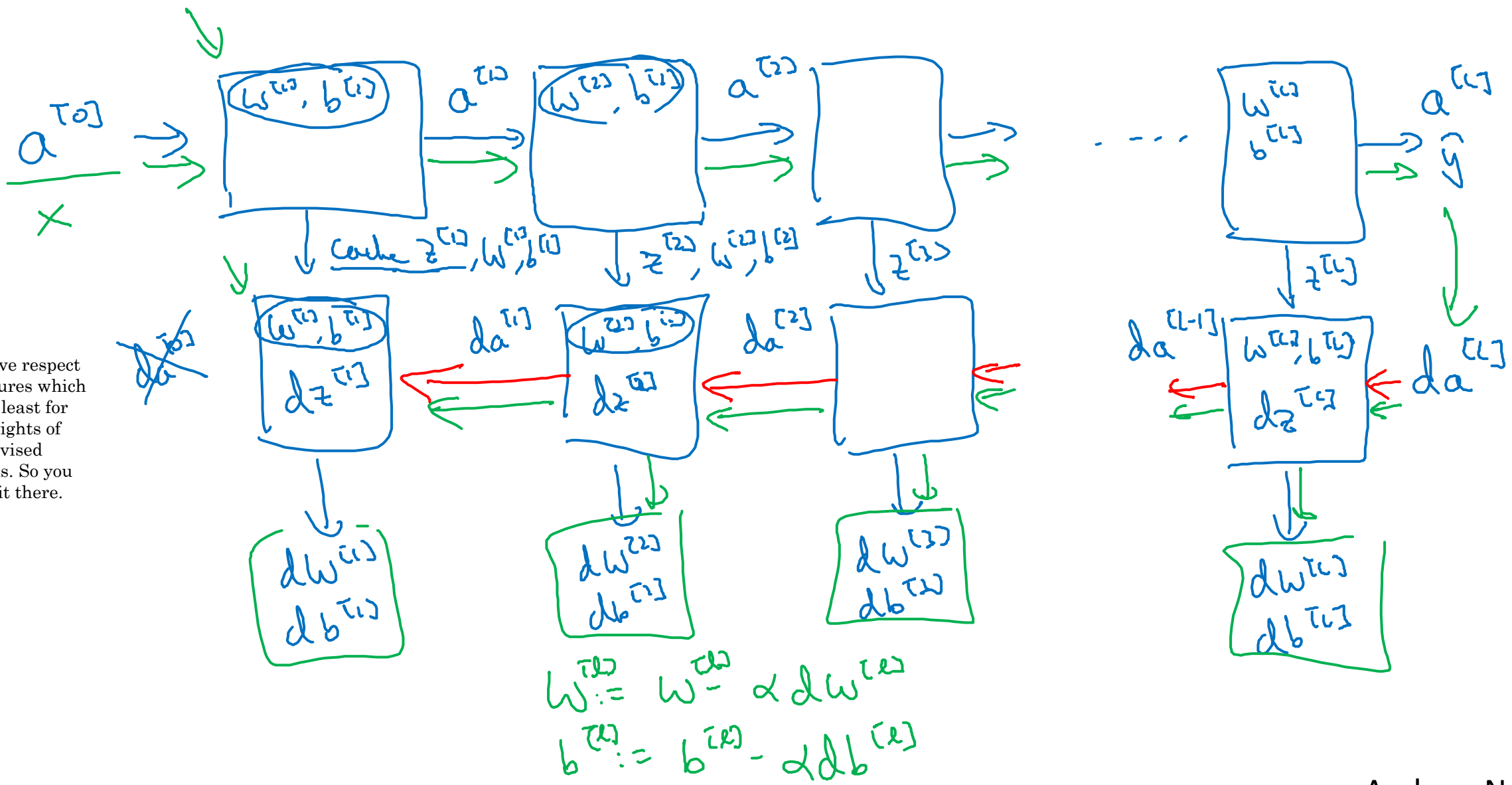
$$z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]} \quad \text{cache } z^{[l]}$$

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

→ Backward: Input  $da^{[l]}$ , output  $da^{[l-1]}$   
 cache  $(z^{[l]})$   
 $\frac{dL}{dW^{[l]}}$   
 $\frac{dL}{db^{[l]}}$



# Forward and backward functions



This is derivative respect your input features which is not useful at least for training the weights of these are supervised neural networks. So you could just stop it there.



deeplearning.ai

# Deep Neural Networks

---

Forward and backward  
propagation

# Forward propagation for layer $l$

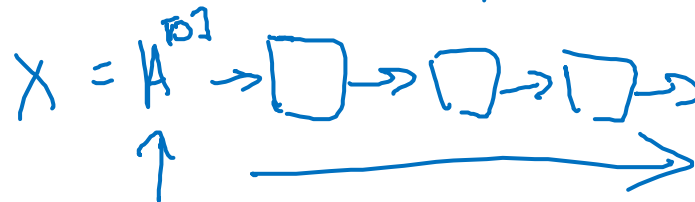
→ Input  $a^{[l-1]}$  ←

→ Output  $a^{[l]}$ , cache ( $z^{[l]}$ )

$$z^{[l]} = W^{[l]} \cdot a^{[l-1]} + b^{[l]}$$

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

$a^{[0]}$   
 $A^{[0]}$



Vectorized:

$$z^{[l]} = W^{[l]} \cdot A^{[l-1]} + b^{[l]}$$

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

So that's the input to the first forward function in the chain and then just repeating this allows you to compute forward propagation from left to right

# Backward propagation for layer $l$

→ Input  $da^{[l]}$

→ Output  $da^{[l-1]}$ ,  $dW^{[l]}$ ,  $db^{[l]}$

element-wise product

$$dz^{[l]} = da^{[l]} * g^{[l]'}(z^{[l]})$$

$$dW^{[l]} = dz^{[l]} \cdot a^{[l-1]}$$

$$db^{[l]} = dz^{[l]}$$

$$da^{[l-1]} = W^{[l]T} \cdot dz^{[l]}$$

$$dz^{[l]} = W^{[l+1]T} dz^{[l+1]} * g^{[l]'}(z^{[l]})$$

$$dz^{[l]} = dA^{[l]} * g^{[l]'}(z^{[l]})$$

$$dW^{[l]} = \frac{1}{n} dz^{[l]} \cdot A^{[l-1]T}$$

$$db^{[l]} = \frac{1}{n} \text{np.sum}(dz^{[l]}, \text{axis}=1, \text{keepdims}=\text{True})$$

$$dA^{[l-1]} = W^{[l]T} \cdot dz^{[l]}$$

# Summary

