Deep Neural Networks

1. Deep L-layer neural networks

Deep neural network notation

L = 4 (# layers)

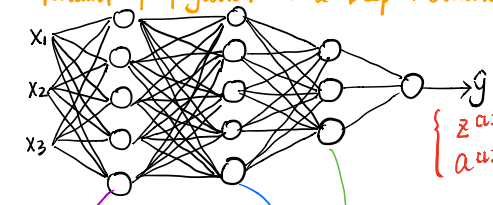
$n^{[l]}$ = # units of layer l, eg: $n^{[1]}=5$, $n^{[2]}=5$, $n^{[3]}=3$, $n^{[4]}=n^{[L]}=1$, $n^{[0]}=n_x=3$

$a^{[l]}$ = activations of layer l

$a^{[l]} = g^{[l]}(z^{[l]})$, $w^{[l]}$ = weights for $z^{[l]}$

↓ $X = a^{[0]}$, $a^{[L]} = \hat{y}$

2. Forward Propagation in a Deep Network



$$\begin{cases} z^{[l]} = w^{[l]}a^{[l-1]} + b^{[l]} \\ a^{[l]} = g^{[l]}(z^{[l]}) \end{cases} \xrightarrow{\text{Vectorized}} \begin{cases} Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]} \\ A^{[l]} = g^{[l]}(Z^{[l]}) \end{cases}$$

$X: z^{[1]} = w^{[1]}a^{[0]} + b^{[1]}$ ($a^{[0]} = x$)

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

$z^{[2]} = w^{[2]}a^{[1]} + b^{[2]}$

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

...

$z^{[4]} = w^{[4]}a^{[3]} + b^{[4]}$

$a^{[4]} = g^{[4]}(z^{[4]}) = \hat{y}$

Vectorized

$$\begin{cases} Z^{[1]} = W^{[1]}X + b^{[1]} \quad (X = A^{[0]}) \\ A^{[1]} = g^{[1]}(Z^{[1]}) \\ Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]} \rightarrow \begin{bmatrix} | & | & & | \\ z^{[2](1)} & z^{[2](2)} & \cdots & z^{[2](m)} \\ | & | & & | \end{bmatrix} \text{ for } l=1,2,3,4 \text{ layer.} \\ A^{[2]} = g^{[2]}(Z^{[2]}) \\ \cdots \\ \hat{Y} = g(Z^{[4]}) = A^{[4]} \end{cases}$$

This loop is 不可避免的.

3. Getting your matrix dimensions right.

When $L=5$, $n^{[1]}=3$, $n^{[2]}=5$, $n^{[3]}=4$, $n^{[4]}=2$, $n^{[5]}=1$

$Z^{[1]} = W^{[1]}X + b^{[1]}$; $(3,1)$ ← $(3,2)$ $(2,1)$; $b^{[1]}$ $(3,1)$

$(n^{[1]},1) \leftarrow (n^{[1]},n^{[0]}) \times (n^{[0]},1)$

$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$; $(5,1)$ ← $(5,3)$ $(3,1)$; $b^{[2]}$ $(5,1)$

$(n^{[2]},1) \leftarrow (n^{[2]},n^{[1]}) \times (n^{[1]},1)$

$Z^{[3]} = W^{[3]}A^{[2]} + b^{[3]}$; $(4,1)$ ← $(4,5)$ $(5,1)$

For dimensions:

$$\begin{cases} w^{[l]} : (n^{[l]}, n^{[l-1]}) \\ b^{[l]} : (n^{[l]}, 1) \\ dw^{[l]} : (n^{[l]}, n^{[l-1]}) \\ db^{[l]} : (n^{[l]}, 1) \end{cases} \qquad z^{[l]}, a^{[l]} : (n^{[l]}, 1)$$

Vectorized implementation

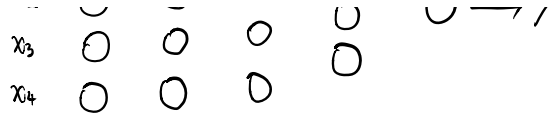
$Z^{[l]} = W^{[l]}X + b^{[l]} \rightarrow (n^{[l]}, 1)$ [through broadcasting, it will transform to $(n^{[l]}, m)$]

$(n^{[l]}, m)$ ← $(n^{[l]}, n^{[0]})$ $(n^{[0]}, m)$

$$\begin{cases} Z^{[l]}, A^{[l]} : (n^{[l]}, m) \\ \text{When } l=0,\ A^{[0]} = X : (n^{[0]}, m) \\ dZ^{[l]}, dA^{[l]} : (n^{[l]}, m) \end{cases}$$

4. Why deep representations?

5. Building blocks of deep neural networks



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

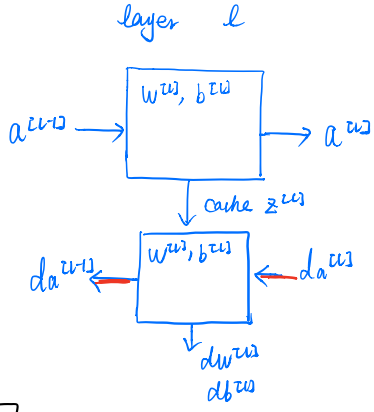
Forward: Input $a^{[l-1]}$, output $a^{[l]}$

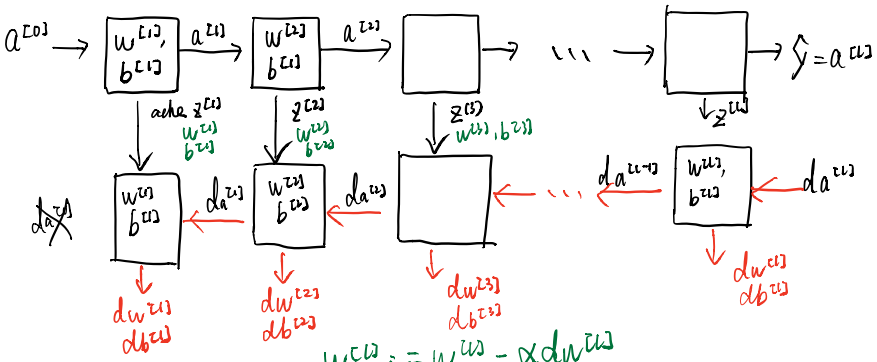
$$\begin{cases} z^{[l]} = W^{[l]} a^{[l-1]} + b^{[l]} \\ a^{[l]} = g^{[l]}(z^{[l]}) \end{cases}$$

cache $z^{[l]}$ (把 $z^{[l]}$ 存储备后续来)

Backward: Input $da^{[l]}$, output $da^{[l-1]}$, $dW^{[l]}$, $db^{[l]}$

cache ($z^{[l]}$)





$$W^{[l]} := W^{[l]} - \alpha dW^{[l]}$$
$$b^{[l]} := b^{[l]} - \alpha db^{[l]}$$

6. Forward and Backward Propagation

Forward propagation for layer l:

Input $a^{[l-1]}$

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

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

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

$X = A^{[0]} \to \square \to \square \to \cdots$

Backward propagation for layer l:

Input $da^{[l]}$

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

$dz^{[l]} = da^{[l]} * g^{[l]\prime}(z^{[l]}) \to dz^{[l]} = W^{[l+1]T} dz^{[l+1]} * g^{[z]\prime}(z^{[l]})$

$dW^{[l]} = dz^{[l]} * a^{[l-1]T}$

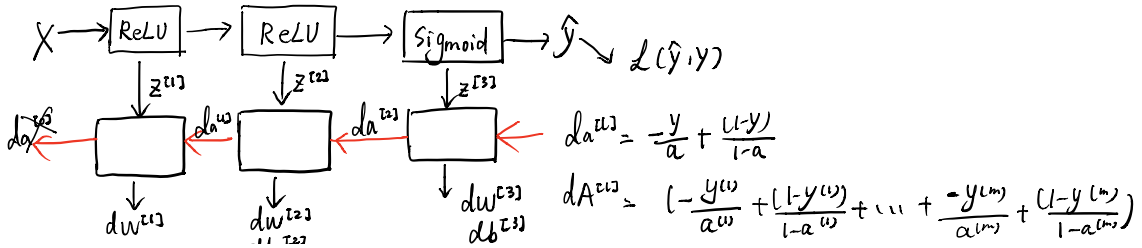
$db^{[l]} = dz^{[l]}$

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

↓ Vectorized

$$\begin{cases} dZ^{[l]} = dA^{[l]} * g^{[l]\prime}(Z^{[l]}) \\ dW^{[l]} = \frac{1}{m} dZ^{[l]} \cdot A^{[l-1]T} \\ db^{[l]} = \frac{1}{m} np.sum(dZ^{[l]}, axis=1, keepdims=True) \\ dA^{[l-1]} = W^{[l]T} \cdot dZ^{[l]} \end{cases}$$

Summary: (3-layer network)



$db^{[1]}$ αb ...

7. Parameters vs. Hyperparameters

Parameters: $W^{[1]}$, $b^{[1]}$, $W^{[2]}$, $b^{[2]}$, $W^{[3]}$, $b^{[3]}$, ...

Hyperparameters:
- Learning rate α
- # iterations
- # hidden layers L
- # hidden units $n^{[1]}, n^{[2]}, \ldots$
- choice of activation function

这些东西是控制 parameters 和最终结果的

Other Hyperparameters: Momentum, minibatch size, regularizations ...