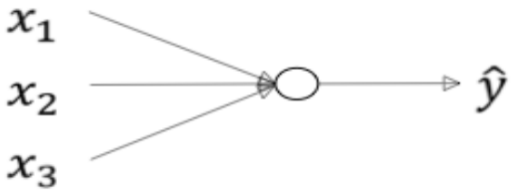
Deep learning

Dr. Aissa Boulmerka
a.boulmerka@centre-univ-mila.dz

2023-2024

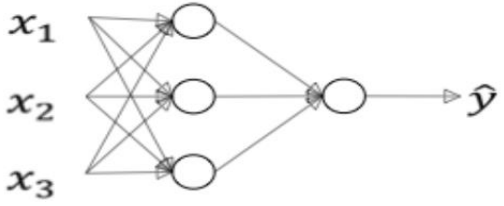
CHAPTER 3
DEEP NEURAL NETWORKS

What is a deep neural network?



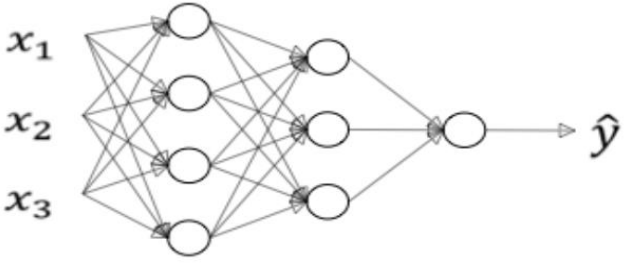
logistic regression

Shallow



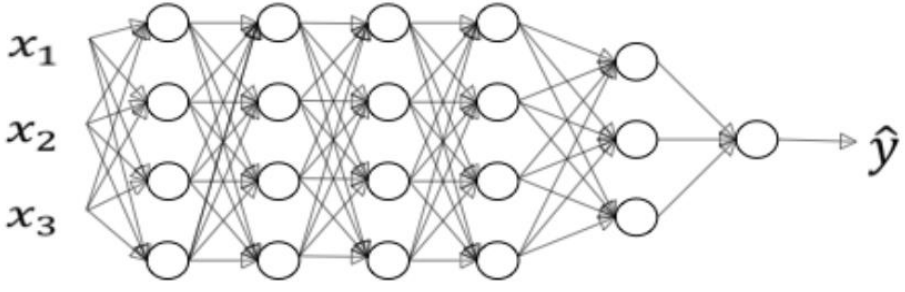
1 hidden layer

2 layer NN



2 hidden layers

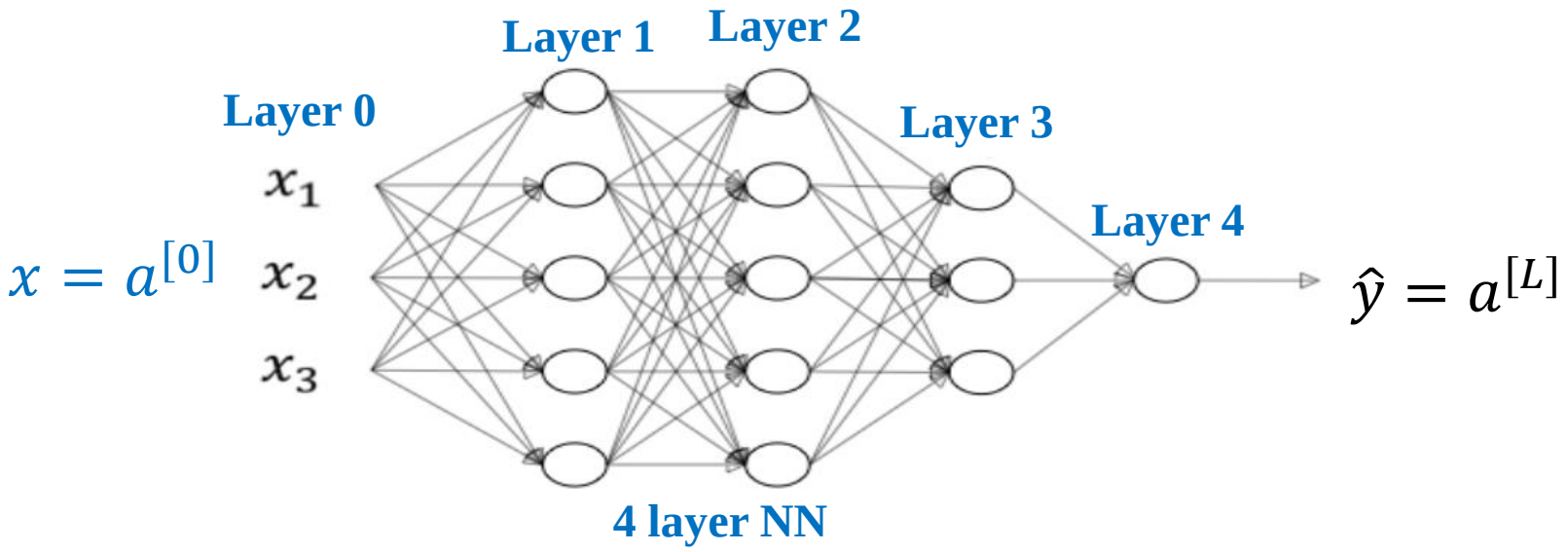
3 layer NN



5 hidden layers

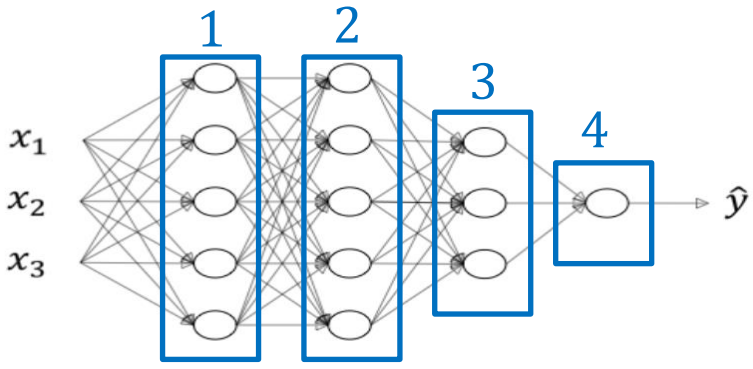
Deep

Deep neural network notation



- $n^{[0]} = n_x = 3, n^{[1]} = 5, n^{[2]} = 5, n^{[3]} = 3, n^{[4]} = n^{[l]} = 1$
- L = 4 : number of layers
- $n^{[l]}$: number of units in layer l
- $a^{[l]}$: activations in layer l with $a^{[l]} = g^{[l]}(z^{[l]})$
- $W^{[l]}$, $b^{[l]}$: weights for $z^{[l]}$

Forward propagation in a deep network

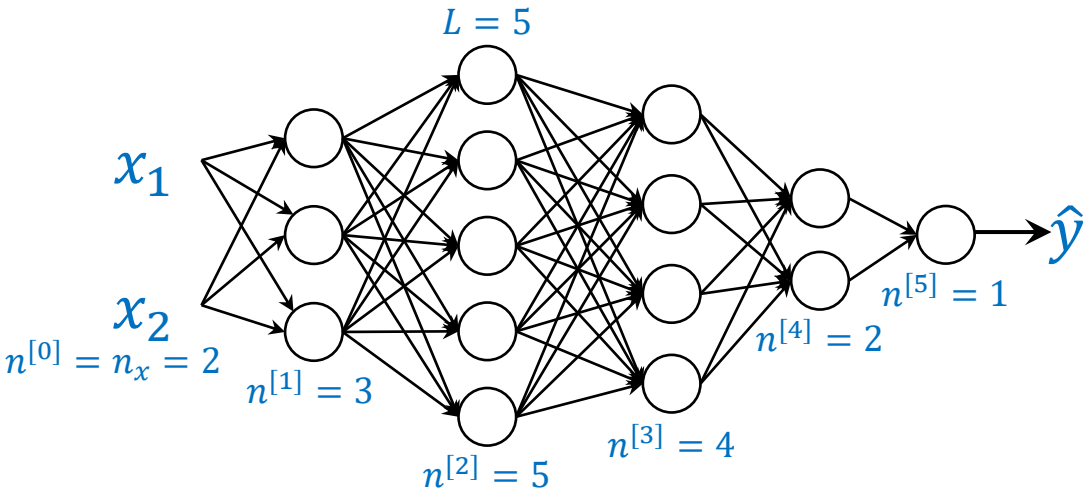


$A^{[0]} = X$

$$Z^{[l]} = W^{[l]} A^{[l-1]} + b^{[l]}$$
$$A^{[l]} = g^{[l]}(Z^{[l]})$$

Iterative	Vectorized
$z^{[1]} = W^{[1]} a^{[0]} + b^{[1]}$	$Z^{[1]} = W^{[1]} A^{[0]} + b^{[1]}$
$a^{[1]} = g^{[1]}(z^{[1]})$	$A^{[1]} = g^{[1]}(Z^{[1]})$
$z^{[2]} = W^{[2]} a^{[1]} + b^{[2]}$	$Z^{[2]} = W^{[2]} A^{[1]} + b^{[2]}$
$a^{[2]} = g^{[2]}(z^{[2]})$	$A^{[2]} = g^{[2]}(Z^{[2]})$
$\vdots$	$\vdots$
$z^{[4]} = W^{[4]} a^{[3]} + b^{[4]}$	$Z^{[4]} = W^{[4]} A^{[3]} + b^{[4]}$
$\hat{y} = a^{[4]} = g^{[4]}(z^{[4]})$	$\hat{Y} = A^{[4]} = g^{[4]}(Z^{[4]})$

Getting your matrix dimensions right
Parameters $W^{[l]}$ and $b^{[l]}$



- $W^{[l]}: (n^{[l]}, n^{[l-1]})$
- $b^{[l]}: (n^{[l]}, 1)$
- $dW^{[l]}: (n^{[l]}, n^{[l-1]})$
- $db^{[l]}: (n^{[l]}, 1)$
- $z^{[l]}: (n^{[l]}, 1)$
- $a^{[l]} = g^{[l]}(z^{[l]}): (n^{[l]}, 1)$

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

(3,1) (3,2) (2,1) (3,1)

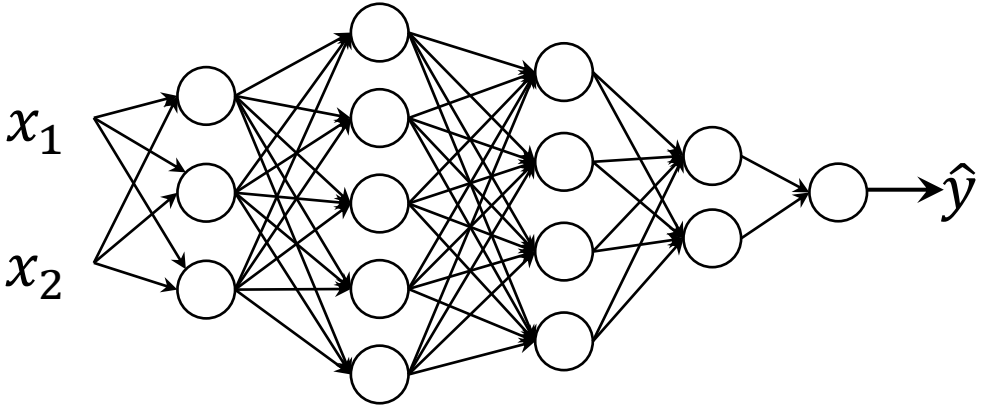
$(n^{[1]}, 1)$ $(n^{[1]}, n^{[0]})$ $(n^{[0]}, 1)$ $(n^{[1]}, 1)$

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

(5,1) (5,3) (3,1) (5,1)

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

Vectorized implementation



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

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

$$Z^{[1]} = W^{[1]}X + b^{[1]}$$

$(n^{[1]}, m)$ $(n^{[1]}, n^{[0]})(n^{[0]}, \mathrm{m})$ $(n^{[1]}, 1)$

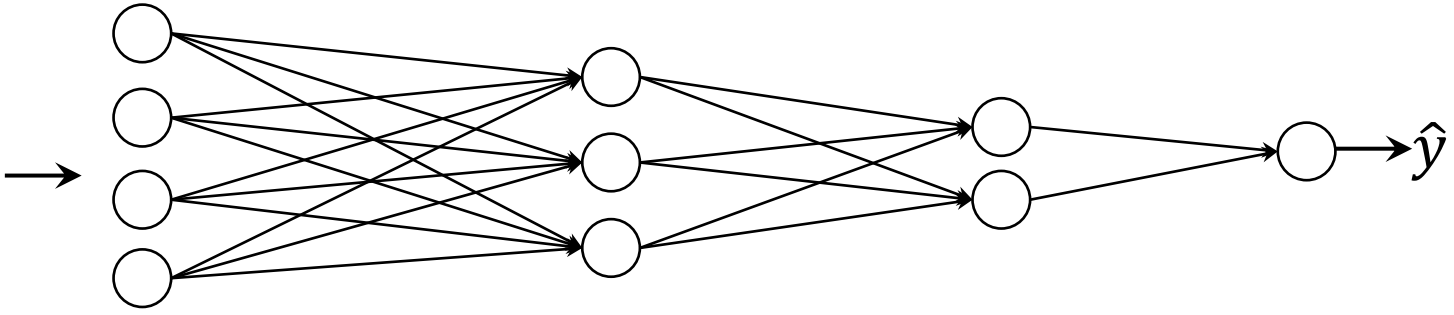
$(n^{[1]}, m)$ broadcasting

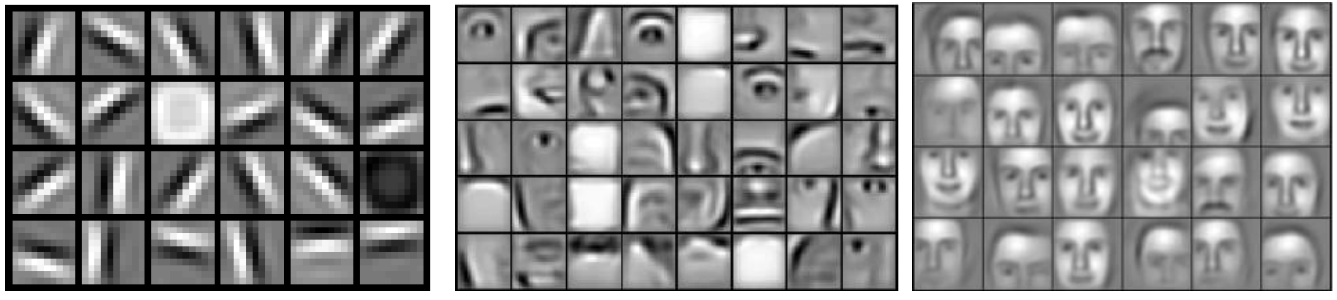
- $z^{[l]}, a^{[l]}: (n^{[l]}, 1)$
- $dz^{[l]}, da^{[l]}: (n^{[l]}, 1)$

- $Z^{[l]}, A^{[l]}: (n^{[l]}, m)$

  $l = 0$: $A^{[0]}: (n^{[0]}, m)$
- $dZ^{[l]}, dA^{[l]}: (n^{[l]}, m)$

Intuition about deep representation





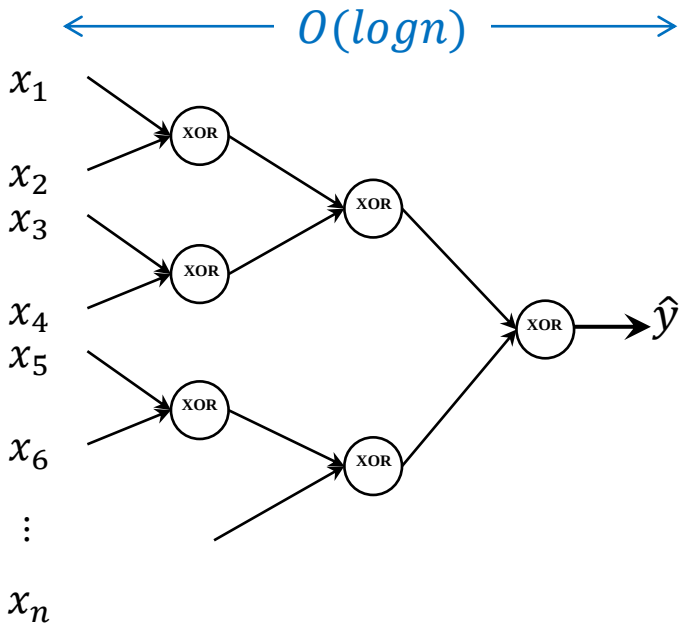
speech → Low level audio wave features → Phonemes C A T → Words → Sentence phrases

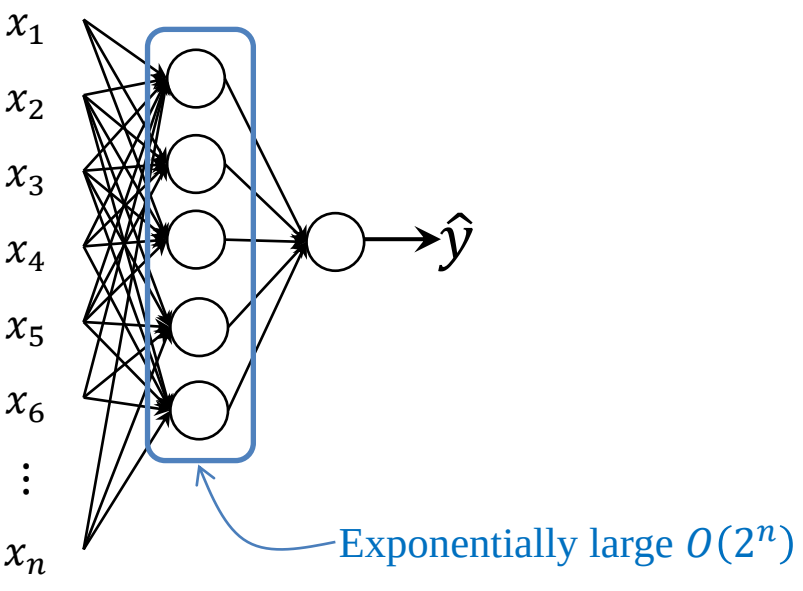
(*) Note that the image in the above example is generated by the algorithm from : https://this-person-does-not-exist.com

Circuit theory and deep learning

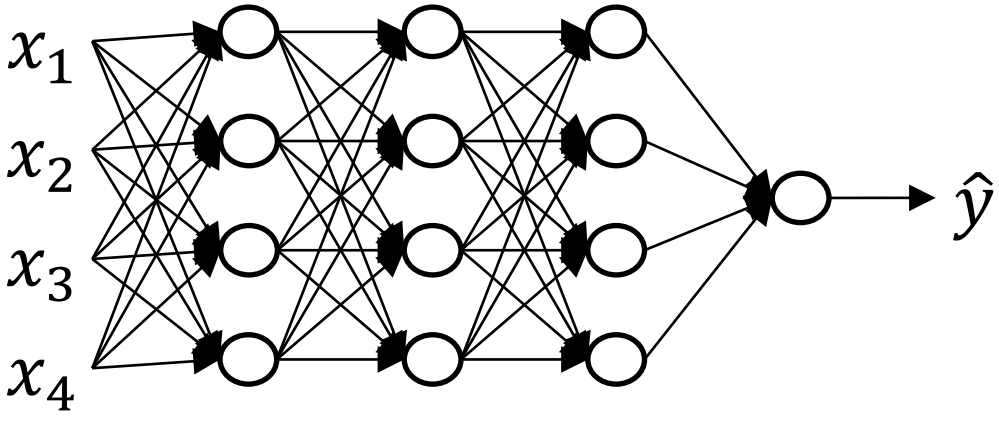
Informally: There are functions you can compute with a "small" L-layer deep neural network that shallower networks require exponentially more hidden units to compute.

$$y = x_1 XOR\ x_2 XOR\ x_3 XOR \cdots XOR\ x_n$$





Forward and backward functions



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

- **Forward:**

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

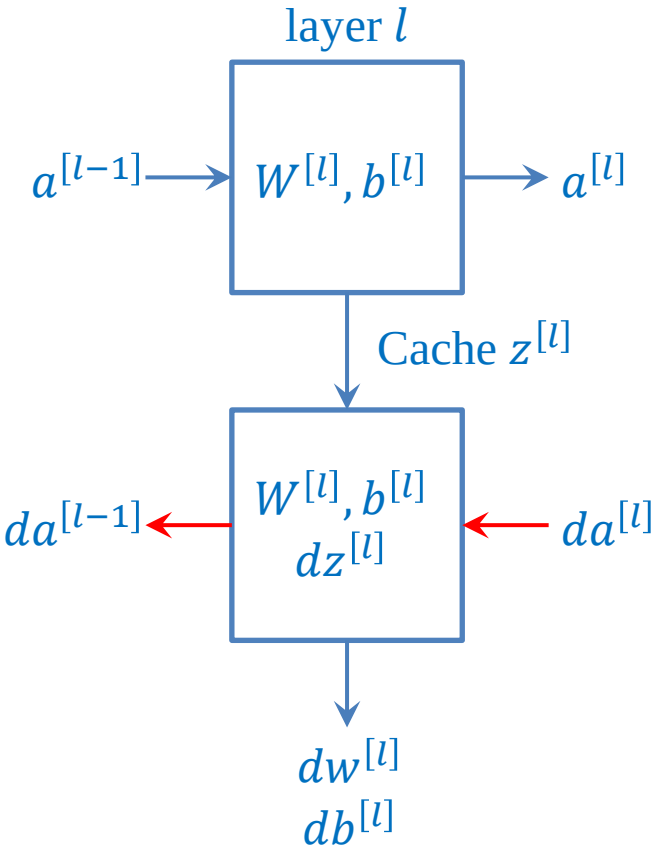
  $z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$ (Cache $z^{[l]}$)

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

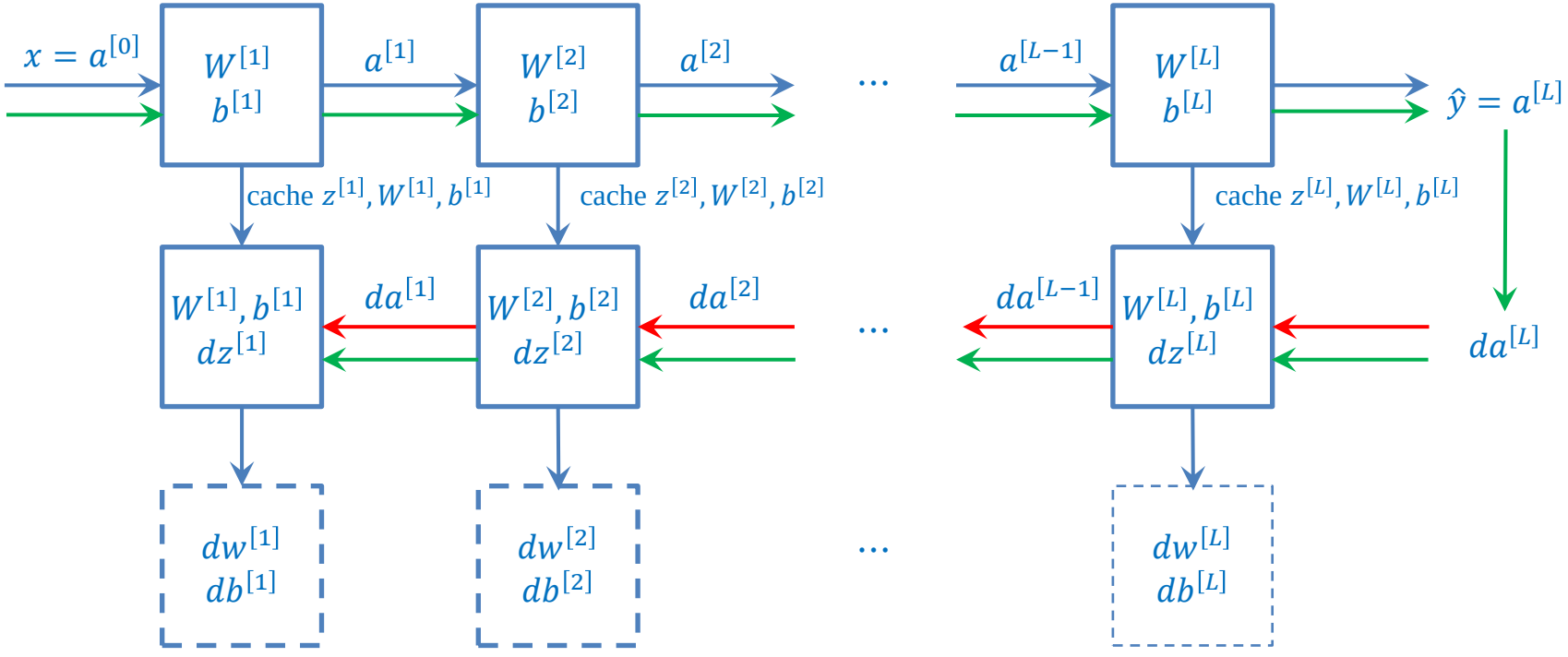
- **Backward:**

  Input: $da^{[l]}$, cache ($z^{[l]}$)

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



Forward and backward functions



$$W^{[l]} \leftarrow W^{[l]} - \alpha dw^{[l]}$$
$$b^{[l]} \leftarrow b^{[l]} - \alpha db^{[l]}$$

Forward propagation for layer l

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

Output: $a^{[l]}$,Cache($Z^{[l]}$)

$$z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]}$$
$$a^{[l]} = g^{[l]}\left(z^{[l]}\right)$$

Vectorized:

$$Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}$$
$$A^{[l]} = g^{[l]}\left(Z^{[l]}\right)$$

Backward propagation for layer l

Input: $da^{[l]}$

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

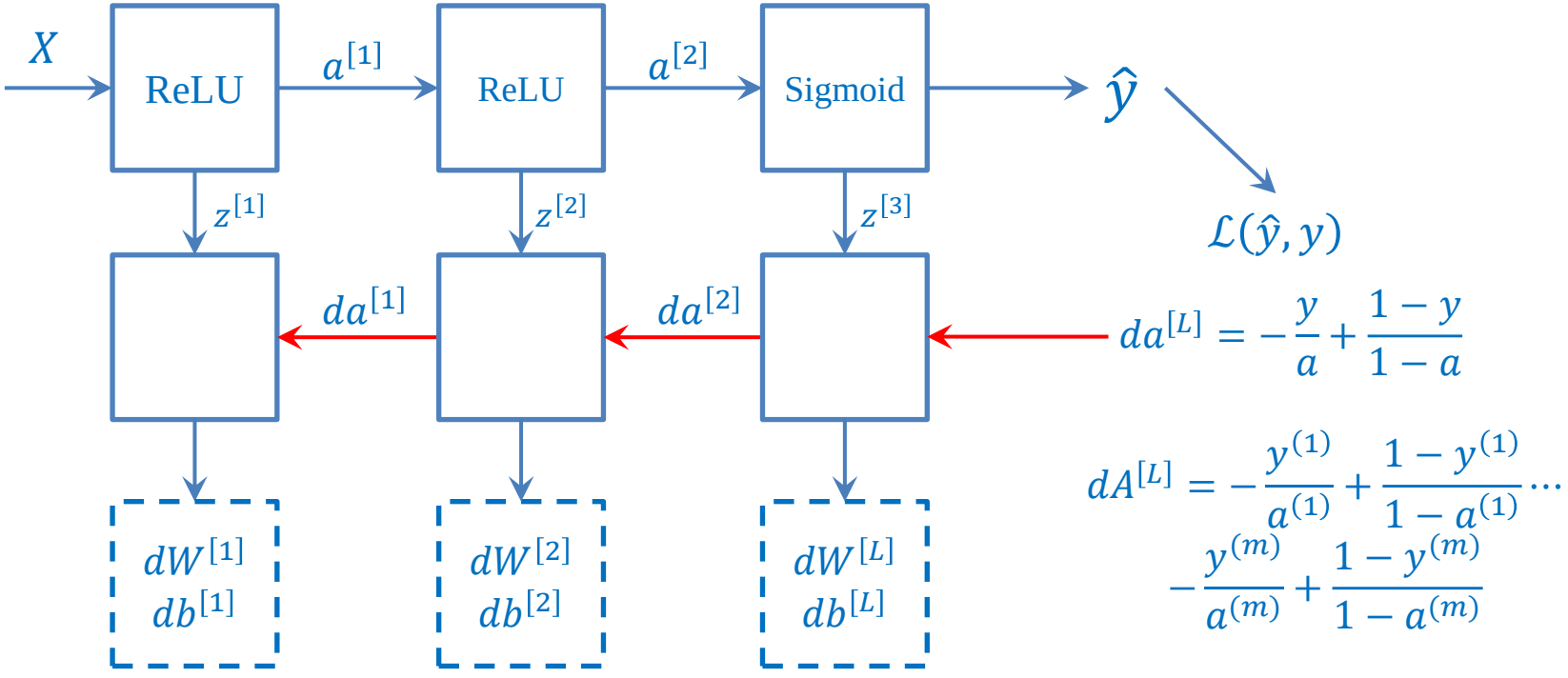
$$dz^{[l]} = da^{[l]} * g^{[l]\prime}\left(z^{[l]}\right)$$
$$dW^{[l]} = dz^{[l]}.a^{[l-1]}$$
$$db^{[l]} = dz^{[l]}$$
$$da^{[l-1]} = W^{[l]T}.dz^{[l]}$$

If we *substitute* $da^{[l]}$ *to* $dz^{[l]}$*:*

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

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

Summary

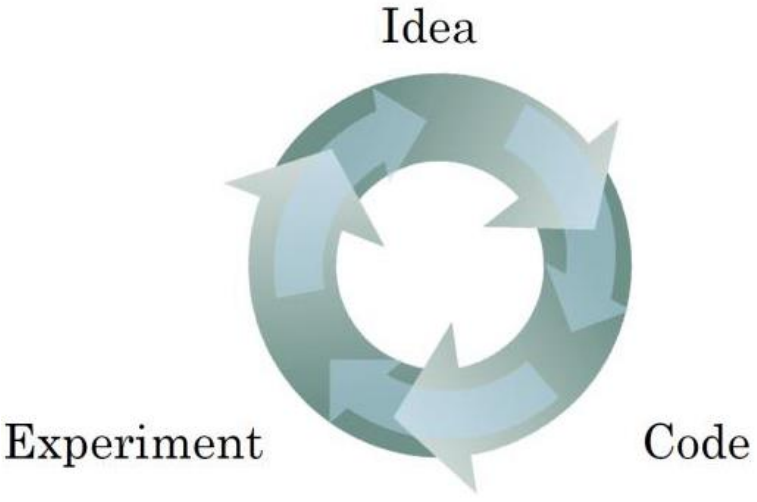

X
ReLU
$a^{[1]}$
ReLU
$a^{[2]}$
Sigmoid
$\hat{y}$
$z^{[1]}$
$z^{[2]}$
$z^{[3]}$
$\mathcal{L}(\hat{y}, y)$
$da^{[1]}$
$da^{[2]}$
$da^{[L]} = -\frac{y}{a} + \frac{1-y}{1-a}$
$dA^{[L]} = -\frac{y^{(1)}}{a^{(1)}} + \frac{1-y^{(1)}}{1-a^{(1)}} \cdots$
$-\frac{y^{(m)}}{a^{(m)}} + \frac{1-y^{(m)}}{1-a^{(m)}}$
$dW^{[1]}$
$db^{[1]}$
$dW^{[2]}$
$db^{[2]}$
$dW^{[L]}$
$db^{[L]}$

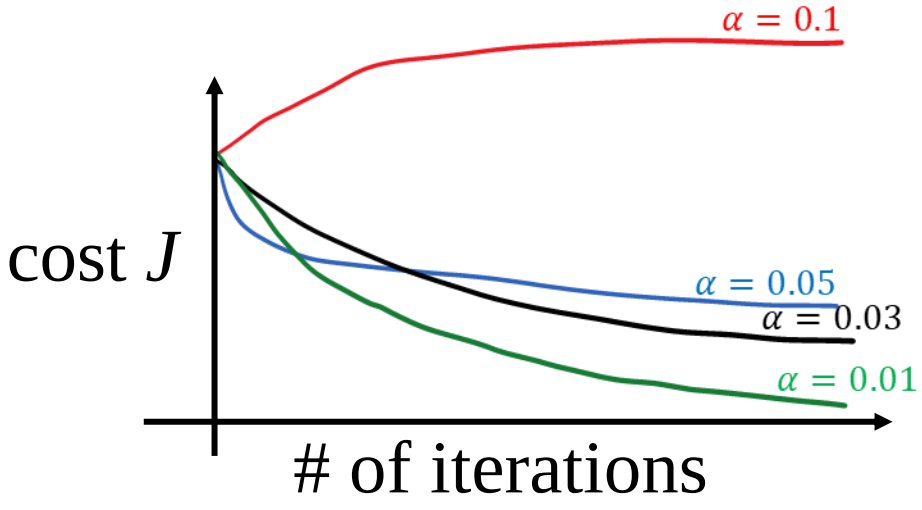
Parameters vs Hyperparameters

- **Parameters**: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, W^{[3]}, b^{[3]}, \dots$
- **Hyperparameters**:
  - Learning rate: α
  - Number of iterations
  - Number of hidden layers: L
  - Number of hidden units: $n^{[1]}, n^{[2]}, n^{[3]}, \dots, n^{[L]}$
  - Choice of activation function

  Later: descent gradient optimization method, momentum, mini-batch size, regularization, … etc.

Parameters: Applied deep learning is a very empirical process

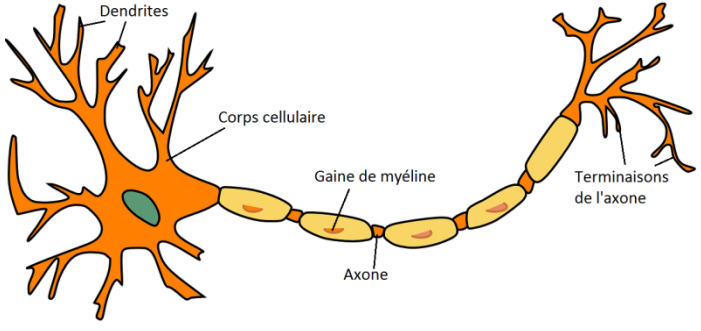




What does this have to do with the brain?

- Forward and backward propagation:

Forward propagation	Backward propagation
$Z^{[1]} = W^{[1]}X + b^{[1]}$ $A^{[1]} = g^{[1]}(Z^{[1]})$ $Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$ $A^{[2]} = g^{[2]}(Z^{[2]})$ $\vdots$ $Z^{[L]} = W^{[L]}A^{[L-1]} + b^{[L]}$ $A^{[L]} = g^{[L]}(Z^{[L]}) = \hat{Y}$	$dZ^{[L]} = A^{[L]} - Y$ $dW^{[L]} = \frac{1}{m} dZ^{[L]} A^{[L]T}$ $db^{[L]} = \frac{1}{m} np.sum(dZ^{[L]}, axis = 1, keepdims = True)$ $dZ^{[L-1]} = dW^{[L]T} dZ^{[L]} g^{[L]\prime}(Z^{[L-]})$ $\vdots$ $dZ^{[1]} = dW^{[1]T} dZ^{[2]} g^{[1]\prime}(Z^{[1]})$ $dW^{[1]} = \frac{1}{m} dZ^{[1]} A^{[1]T}$ $db^{[1]} = \frac{1}{m} np.sum(dZ^{[1]}, axis = 1, keepdims = True)$



References

- Andrew Ng. Deep learning. Coursera.
- Geoffrey Hinton. Neural Networks for Machine Learning.
- Kevin P. Murphy. Probabilistic Machine Learning An Introduction. MIT Press, 2022.
- MIT Deep Learning 6.S191 (http://introtodeeplearning.com/)