Deep Neural Networks

Week 3

Content

- Deep Neural Networks - Shallow NN's
- Neural Network Notation • $n^{[l]} =$ • $W^{[l]} =$
- Forward Propagation
- Backward Propagation
- Hyperparameters
- Universal Approximation Theorem

Deep Neural Networks

Deep Neural Networks

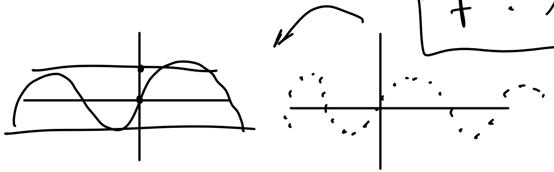
$f : X \rightarrow y$

$f_3(f_2(f_1(x)))$

first l_1

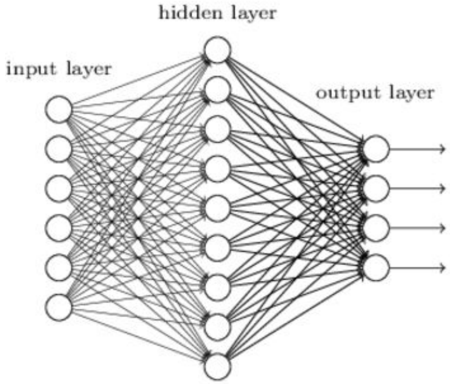
second

third

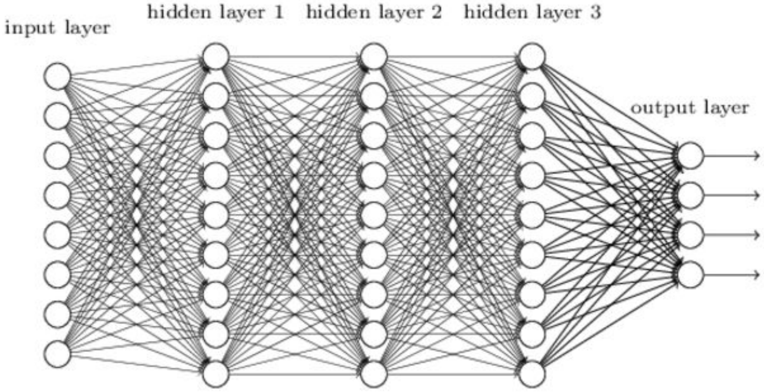


- Shallow neural networks is a term used to describe NN that usually have only one hidden layer while the term deep neural networks is used to describe NN that have several hidden layers.
- The deep NN with the right architectures achieve better results than shallow ones that have the same computational power.

"Non-deep" feedforward neural network



Deep neural network



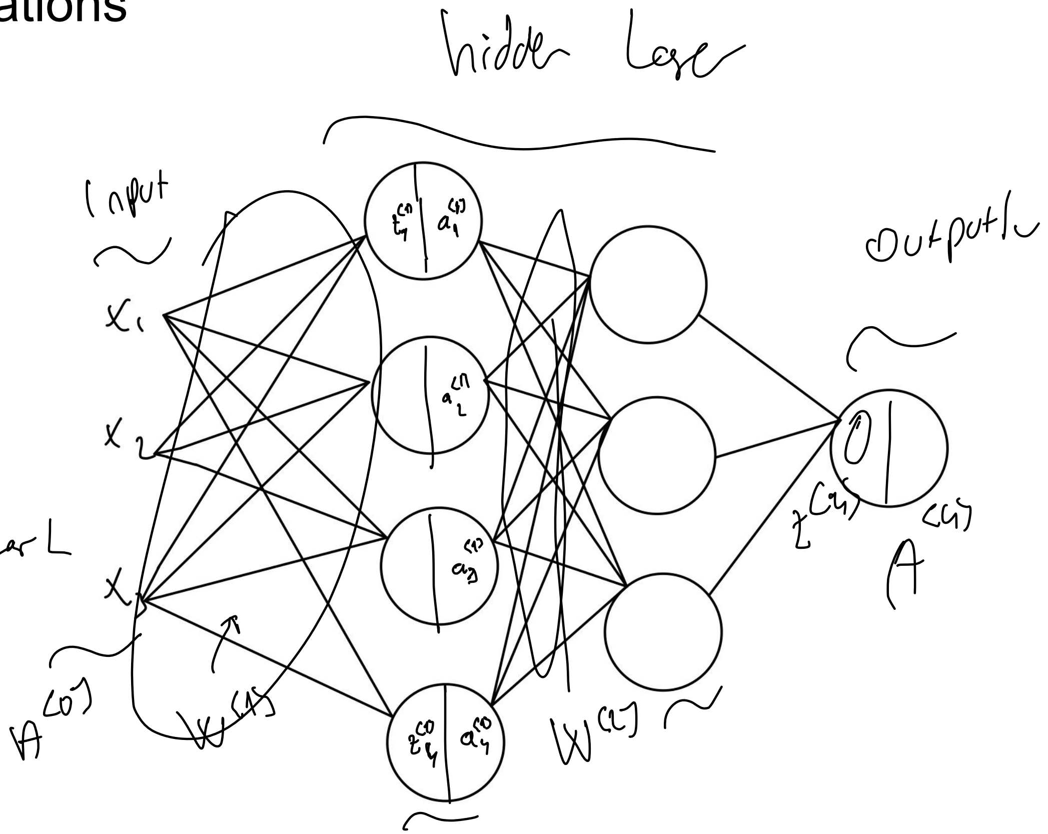
Neural Network Notations

$n^{[0]} = 3$

$n^{[1]} = 4$

$n^{[2]} = 1$

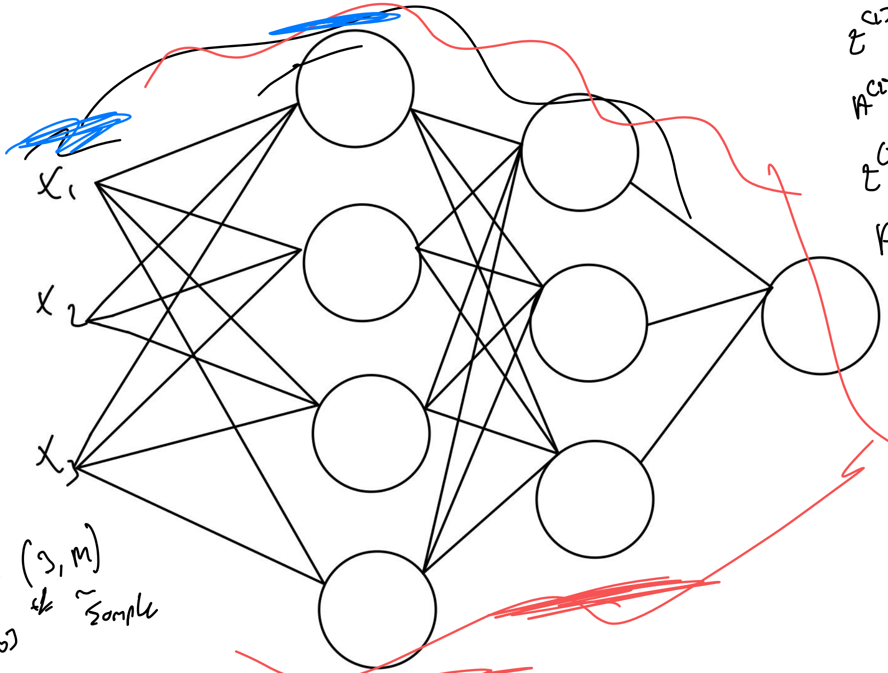
$n^{[L]} =$ # of units in Layer L



Forward Propagation

$g^{[l]}$: action function in layer l

$$z^{[1]} = W^{[1]} A^{[0]} + b^{[1]}$$
$$A^{[1]} = g(z^{[1]})$$
$$z^{[2]} = W^{[2]} A^{[1]} + b^{[2]}$$
$$A^{[2]} = g^{[2]}(A^{[2]})$$
$$z^{[3]} = W^{[3]} A^{[2]} + b^{[3]}$$
$$A^{[3]} = g^{[3]}(z^{[3]})$$



$A^{[0]} = (3, m)$

$n^{[0]}$ sample

$$z^{[1]} = \underset{4\times 3}{W^{[1]}}\ \underset{3\times 1}{A^{[0]}} + \underset{4\times 1}{b^{[1]}}$$

$$\boxed{4\times 1} = 4\times 1 + 4\times 1 =$$

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

4×1

$$z^{[2]} = \underset{3\times 4}{W^{[2]}}\ \underset{4\times 1}{A^{[1]}} + \underset{3\times 1}{b^{[1]}}$$

$$\underset{3\times 1}{\downarrow} = 3\times 1 + 3\times 1$$

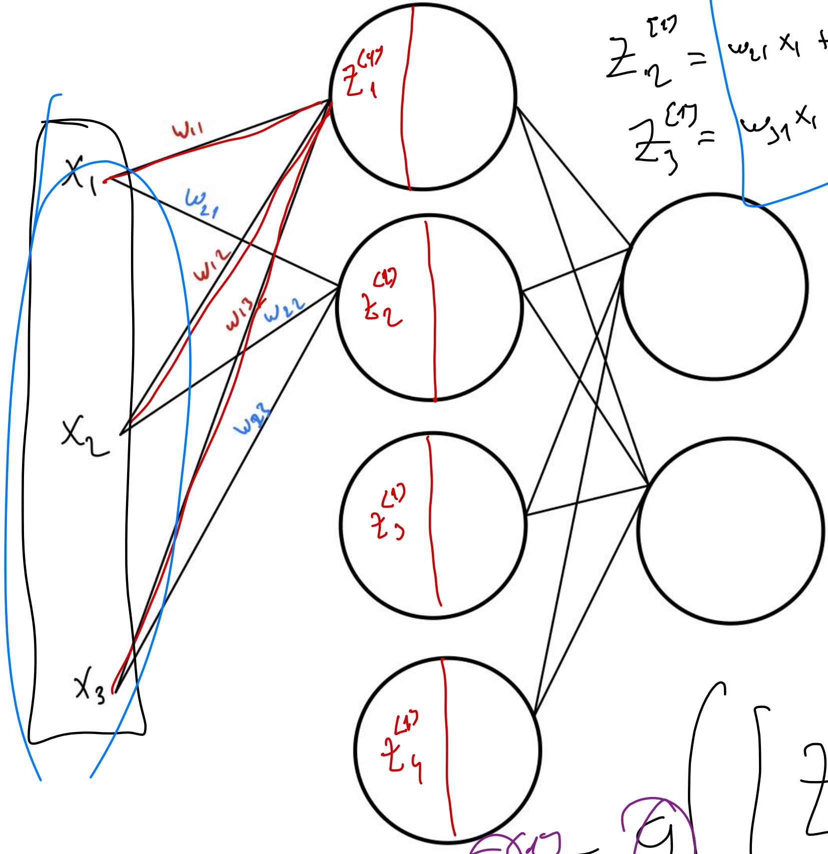
$W^{[l]}$.shape $= (n^{[l]}, n^{[l-1]})$

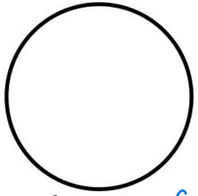
$b^{[l]}$.shape $= (n^{[l]}, 1)$

$z^{[l]}$.shape $=$

$A^{[l]}$.shape $= (n$

Forward Propagation





$$Z_1^{[1]} = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1^{[1]}$$

$$Z_2^{[1]} = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2^{[1]}$$

$$Z_3^{[1]} = w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + b_3^{[1]}$$

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1m} \\ X_{21} & X_{22} & \cdots & X_{1m} \\ X_{31} & \cdots & \cdots & X_{3m} \end{bmatrix}$$

$(3,1)$

$w^{[1]}.\text{shape} = (4,3)$

$b^{[1]}.\text{shape} = (4,1)$

$z^{[1]}.\text{shape} = (4,m)$

$$w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b_1^{[1]} = z^{[1]}$$

$$w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b_2^{[1]} = z_2^{[1]}$$

$$w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + b_3^{[1]} = z_3^{[1]}$$

$$w_{41}x_1 + w_{42}x_2 + w_{43}x_3 + b_4^{[1]} = z_4^{[1]}$$

$$\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} Z \end{bmatrix}$$

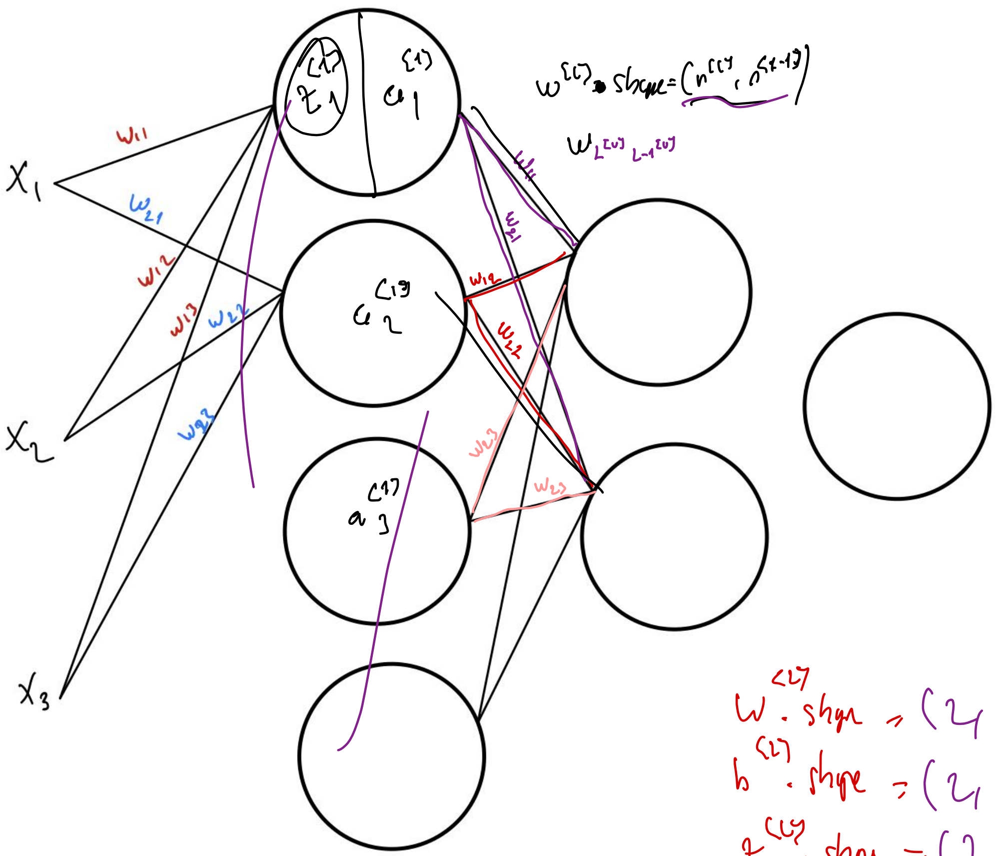
$(4,3)\ (3,m) + (4,1) \quad (4,m)$

$(4,m) + (4,1)$

$(4,m) \leftarrow A^{[1]} = g\left(\begin{bmatrix} Z \end{bmatrix}\right)$

Forward Propagation

$L^{[u]}$ = u th unit in Layer L



$w^{[l]}$. shape $= (n^{[l]}, n^{[l-1]})$

$w_{L^{[u]}\, L-1^{[u]}}$

$$w_{11} a_1^{[1]} + w_{12} a_2^{[1]} + w_{13} a_3^{[1]} + w_{14} a_4^{[1]} = z_1^{[2]}$$

$$w_{21} a_1^{[1]} + w_{22} a_2^{[1]} + w_{23} a_3^{[1]} + w_{24} a_4^{[1]} = z_1^{[2]}$$

$$\begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \end{bmatrix} \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \\ a_4^{[1]} \end{bmatrix} + \begin{bmatrix} b_1^{[2]} \\ b_2^{[2]} \end{bmatrix} = \begin{bmatrix} Z \end{bmatrix}$$

$(2,4) \times (4,m)$

$(2,m) + (2,1)$

$(2,m) \to [Z]$.shape

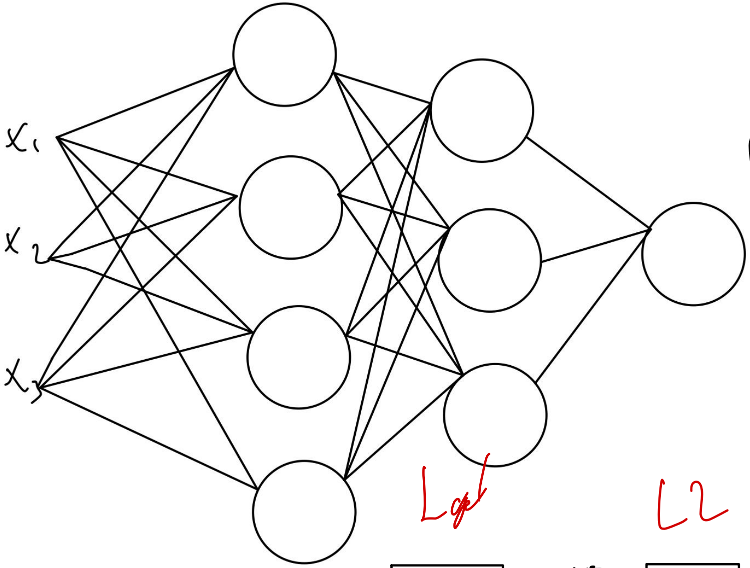
$W^{[2]}$.shape $= (2, 4)$

$b^{[2]}$.shape $= (2, 1)$

$z^{[2]}$.shape $= (2, m)$

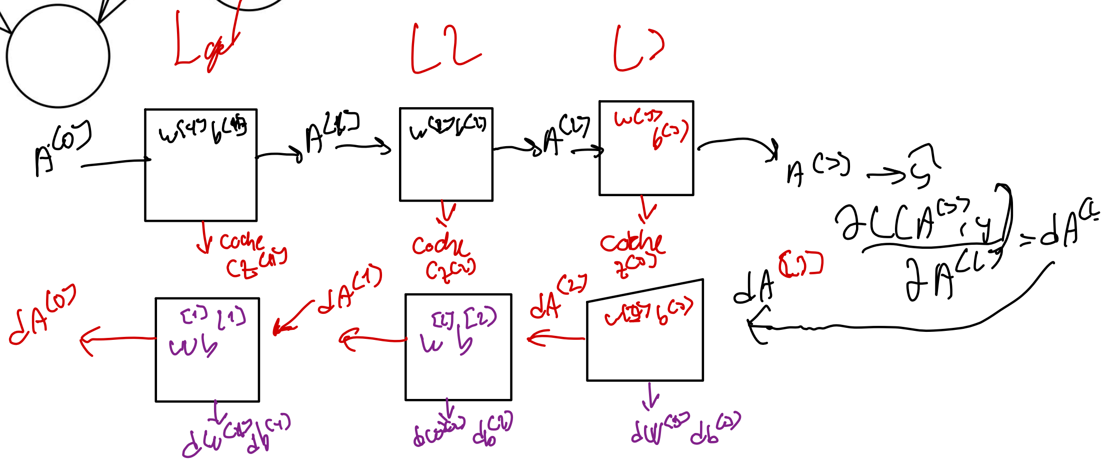
Backward Propagation

Backward Propagation



forward = input $A^{[l-1]}$ → $A^{[l]}$, $cache(Z^{[l]})$

backward = input $dA^{[l]}$, $cache(Z^{[l]})$ Output $dA^{[l-1]}$, $dW^{[l]}$, $db^{[l]}$



$$\boxed{dA^{[L]} = \frac{\partial J}{\partial A^{[L]}}}$$

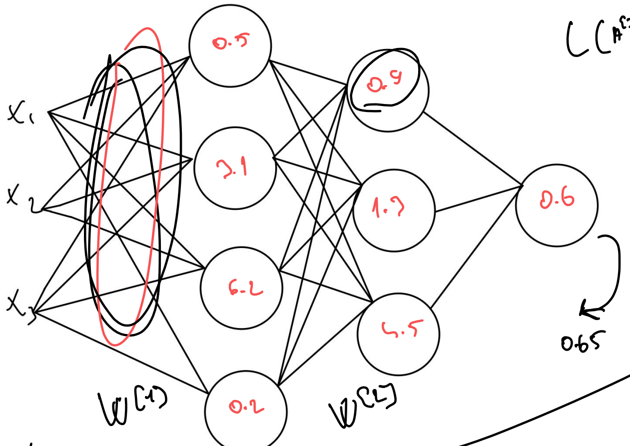
Backward Propagation

$A^{[3]} = g(z^{[3]})$

$z^{[3]} = W^{[3]} A^{[2]} + b^{[3]}$

$L(A^{[3]}) = \boxed{y \log A^{[3]} + (1-y) \log A^{[3]}}$

binary cross-entropy



$A^{[3]} = \hat{y}$

$A^{[3]} = g(z^{[3]})$

$L(\hat{y}) = 1 \cdot \log^{0.6} = \log^{0.6}$

$$\frac{\partial \log A^{[3]}}{\partial W^{[3]}} = \frac{\partial L}{\partial A^{[3]}} \cdot \frac{\partial A^{[3]}}{\partial z^{[3]}} \cdot \frac{\partial z^{[3]}}{\partial W^{[3]}}$$

$\frac{1}{0.6}$

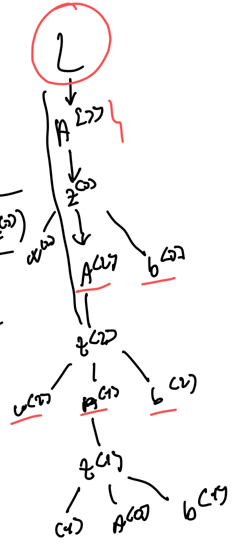
$$\frac{\partial L}{\partial A^{[3]}} = \frac{\partial(\log A^{[3]})}{\partial W^{[3]}} = \frac{1}{A^{[3]}} =$$

$$\frac{\partial A^{[3]}}{\partial z^{[3]}} = \frac{\partial(\sigma(z^{[3]}))}{\partial z^{[3]}} = \sigma(z^{[3]})(1-\sigma(z^{[3]}))$$

$$\frac{\partial z^{[3]}}{\partial W^{[3]}} = \frac{\partial(W^{[3]}A^{[2]} + b^{[3]})}{\partial W^{[3]}} = A^{[2]}$$

$W^{[3]}A^{[2]} + b^{[3]}$

$z^{[3]} = WA + b$



$$\int f(\tau) \cdot h(t-\tau)\, d\tau$$

$$\frac{\partial L}{\partial W^{[2]}} = \frac{\partial L}{\partial A^{[3]}} \cdot \frac{\partial A^{[3]}}{\partial z^{[3]}} \cdot \frac{\partial z^{[3]}}{\partial A^{[2]}} \cdot \frac{\partial A^{[2]}}{\partial z^{[2]}} \cdot \frac{\partial z^{[2]}}{\partial W^{[2]}}$$

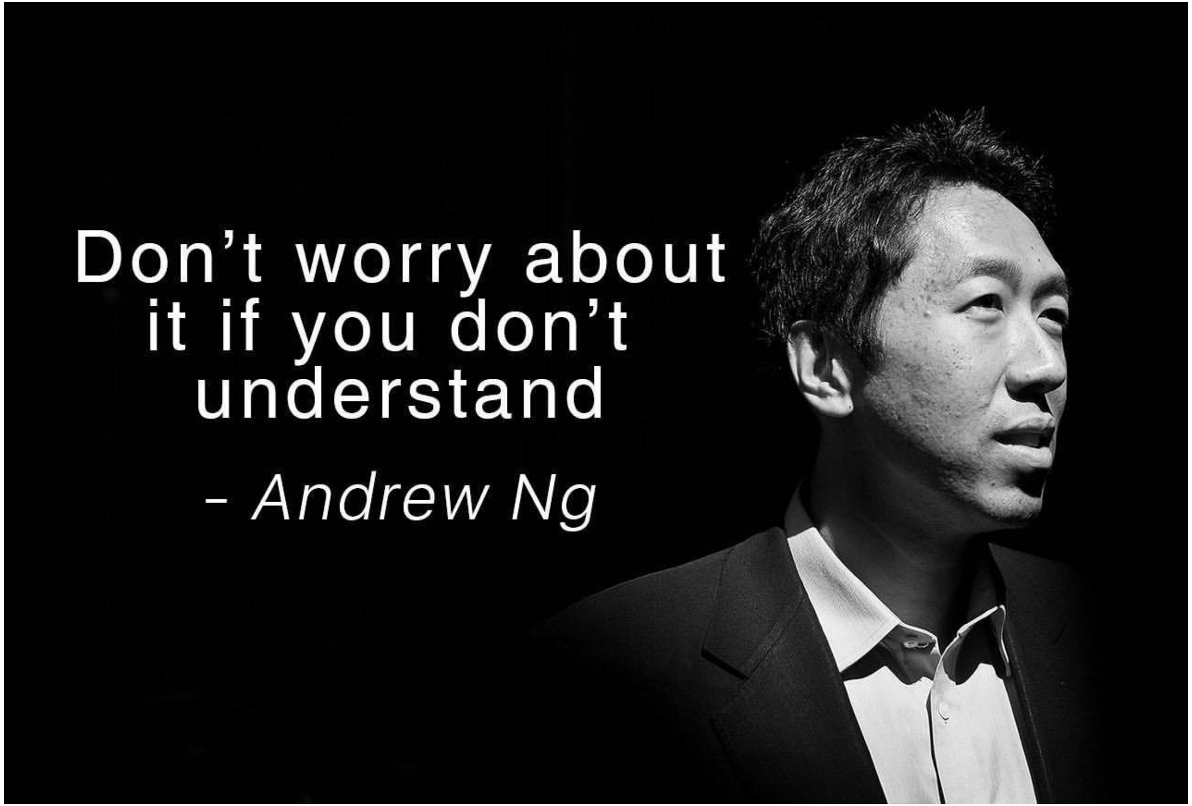
$$\frac{1}{A^{[3]}} \cdot \sigma'(z^{[3]}) \cdot W^{[3]} \cdot \sigma'(z^{[2]}) \cdot A^{[1]}$$

$$\frac{1}{0.6} \cdot \frac{\partial L}{\partial W^{[3]}} = \frac{1}{A^{[3]}} \cdot \sigma'(z^{[3]}) \cdot A^{[2]}$$

$$W^{[3]} = W^{[3]} - \alpha\, dW^{[3]}$$

$$\frac{A^{[3]}}{z^{[3]}} = \frac{\sigma(z^{[3]})}{z^{[3]}} = \frac{\sigma(W^{[3]}A^{[2]} + b^{[3]})}{z^{[3]}} = \left[\sigma(z^{[3]}) \cdot (1-\sigma(z^{[3]}))\right] = \frac{\partial A^{[3]}}{\partial z^{[3]}}$$

Hyperparameters



Hyperparameters

Hyperparameters effect parameters

Hyperparameter examples:

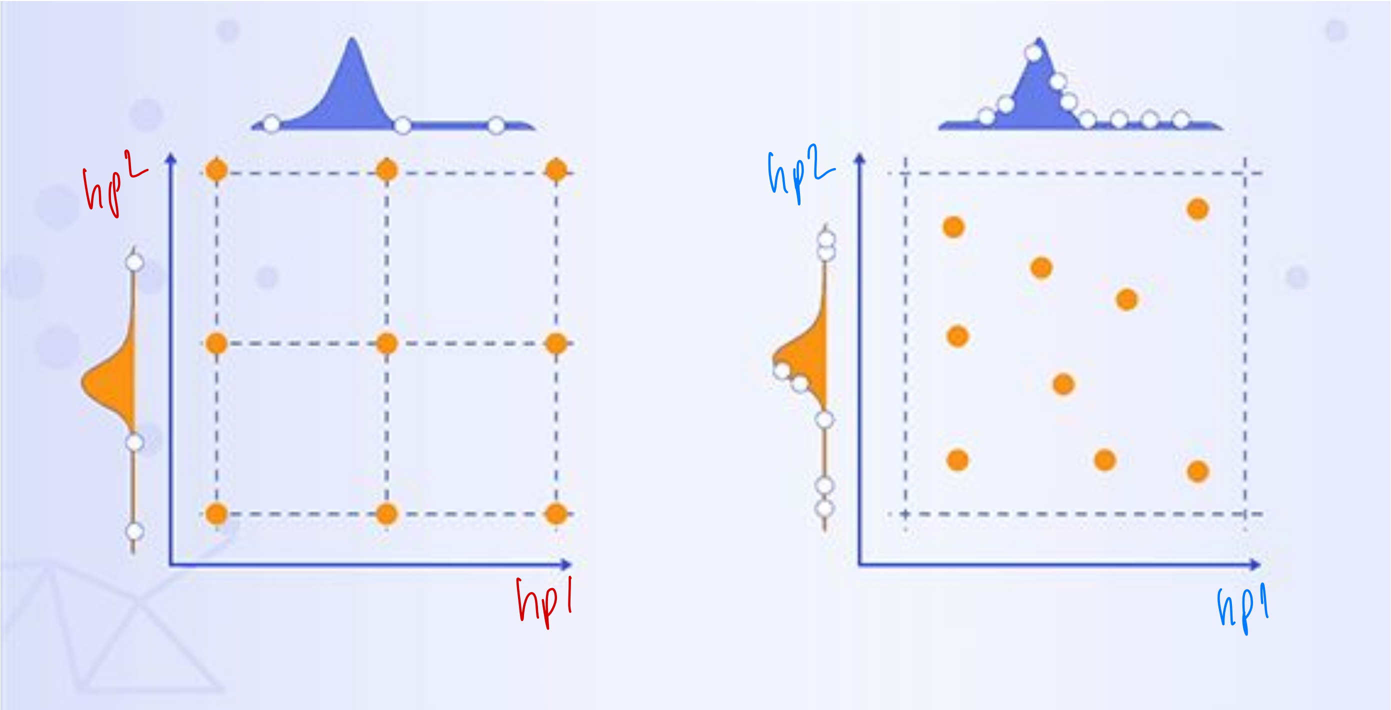
- Learning Rate
- #Units
- #Iterations
- #Layers
- Batch size

We can select hyperparameters using several methods

Hyperparameter Tuning

Grid Search

Random Search



Universal Approximation Theorem

The Universal Approximation Theorem tells us that Neural Networks has a kind of universality no matter what f(x) is, there is a network that can approximately approach the result.

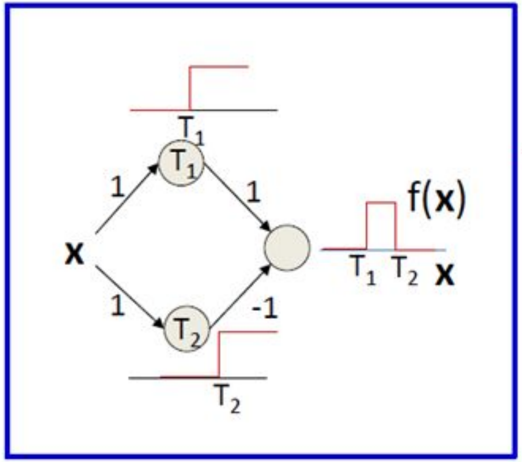
NN's can approximate any f.

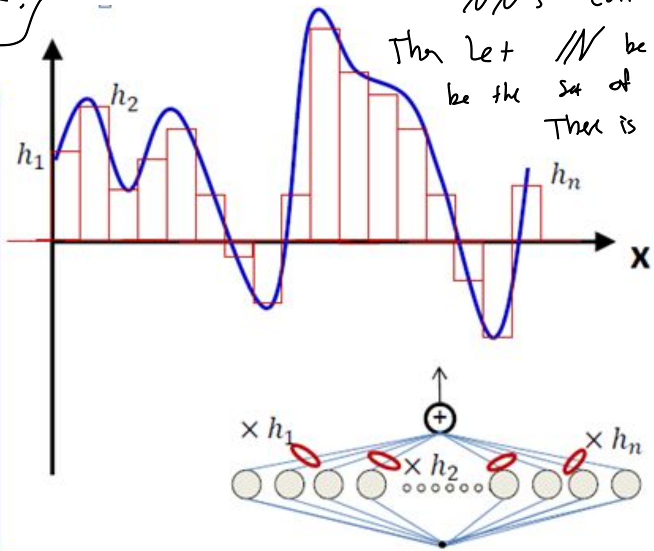
Expl

NN's can be represented as functions. Then let $\mathbb{N}$ be the set of NN's and F be the set of functions on cont. space. Then There is a function such that

$G : \mathbb{N} \longrightarrow F$

G is onto or surjective





inzva: *brings the AI fellows together*

inzva: