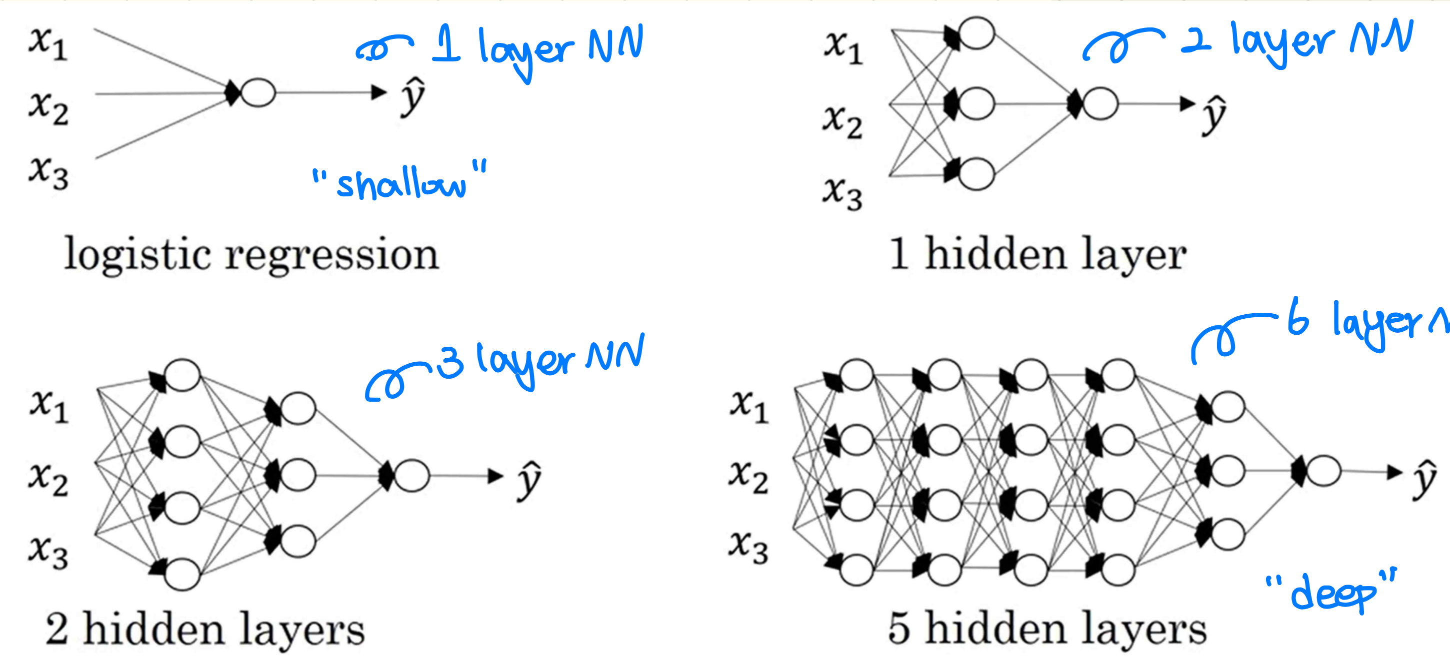
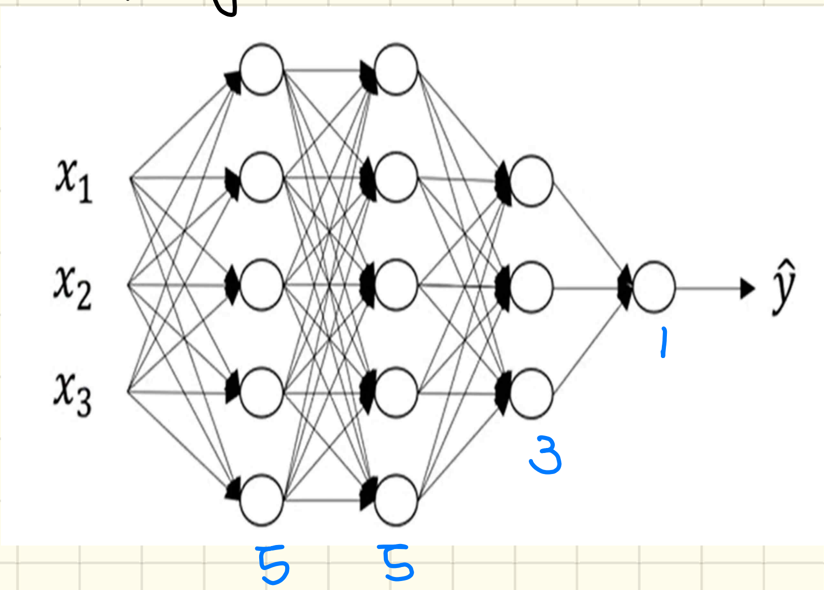
# [4-1. Deep Neural Network]

## < Deep L-layer Neural Network >




* 얼마나 깊은 신경망을 사용해야 하는지는 예측하기 어려우며, 하이퍼파라미터를 조정하면서 깊이를 결정해야 한다.

ex) 4 layer NN




$L =$ #layers $= 4$

$n^{[l]} =$ # units in layer $l$

$n^{[0]} = 3$, $n^{[1]} = 5$, $n^{[2]} = 5$

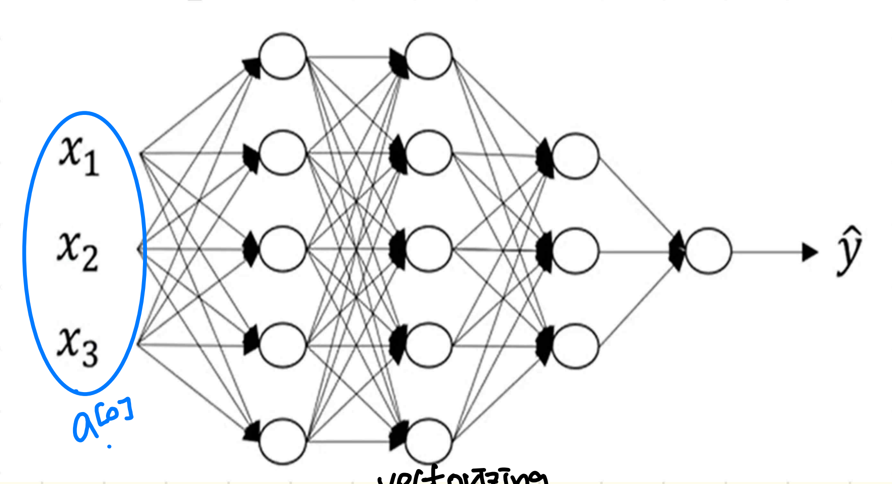
$n^{[3]} = 3$, $n^{[4]} = n^{[L]} = 1$

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

$a^{[0]} = X$, $a^{[L]} = \hat{y}$

$w^{[l]}, b^{[l]}$ : weighs for $z^{(l)}$

# <Forward Propagation in a Deep Network>




vectorizing

$z^{[1]} = w^{[1]} \cancel{x}^{\,a^{[0]}} + b^{[1]}$

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

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

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

...

$\Rightarrow z^{[l]} = w^{[l]} a^{[l-1]} + b^{[l]}$

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

$Z^{[1]} = W^{[1]} A^{[0]} + b^{[1]}$

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

$Z^{[2]} = W^{[2]} A^{[1]} + b^{[2]}$

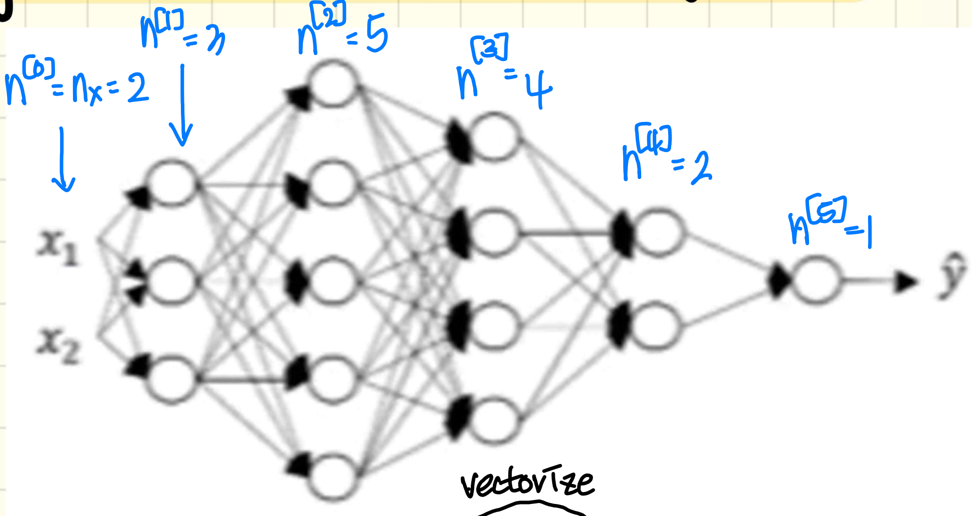
$A^{[2]} = g^{[2]}(Z^{[2]})$

...

$\hat{Y} = g(Z^{[4]}) = A^{[4]}$

for $l=1$ to $4$

# <Getting your Matrix Dimensions Right>




vectorize

$\underset{(3,1)}{z^{[1]}} = \underset{(3,2)}{w^{[1]}} \underset{(2,1)}{x} + \underset{(3,1)}{b^{[1]}} \Rightarrow z^{[l]} = (n^{[l]}, 1)$

$\underset{(5,1)}{z^{[2]}} = \underset{(5,3)}{w^{[2]}} \underset{(3,1)}{a^{[1]}} + \underset{(3,1)}{b^{[2]}}$

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

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

$dw^{[l]} = (n^{[l]}, n^{[l-1]})$

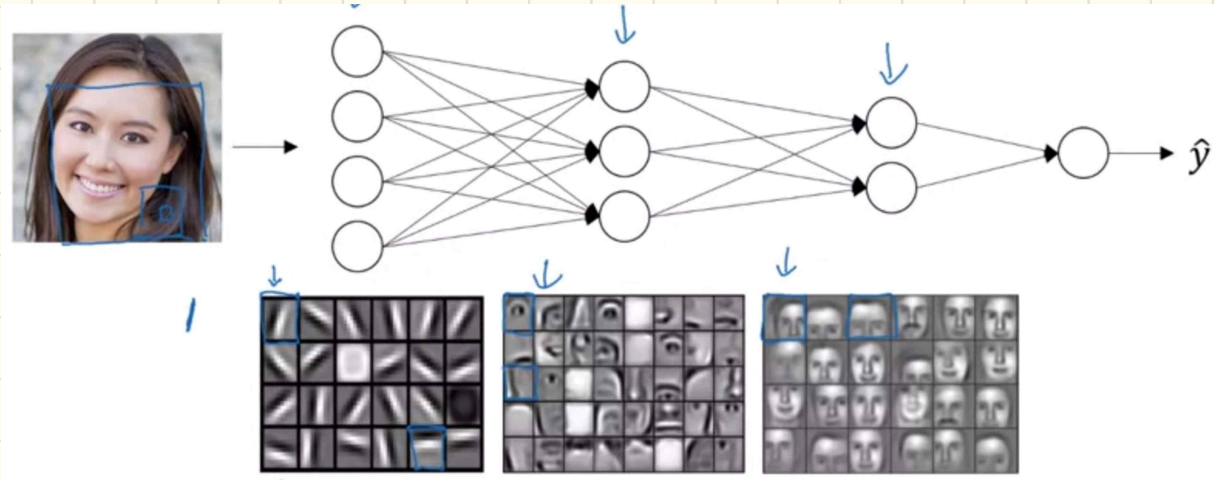
$db^{[l]} = (n^{[l]}, 1)$

$\underset{(3,m)}{Z^{[1]}} = \underset{(3,2)}{W^{[1]}} \underset{(2,m)}{X} + \underset{(3,1)}{b^{[1]}}$

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

$dZ^{[l]}, dA^{[l]} : (n^{[l]}, m)$

# <Why Deep Representations?>



직관 1) 네트워크가 깊어질수록 더 많은 특징을 잡아낼 수 있다. 낮은 층에서는 간단한 특징을 찾고 깊은 층에서는 낮은 층에서 탐지한 간단한 것들을 종합해서 더 복잡한 특징을 찾아낼 수 있다.

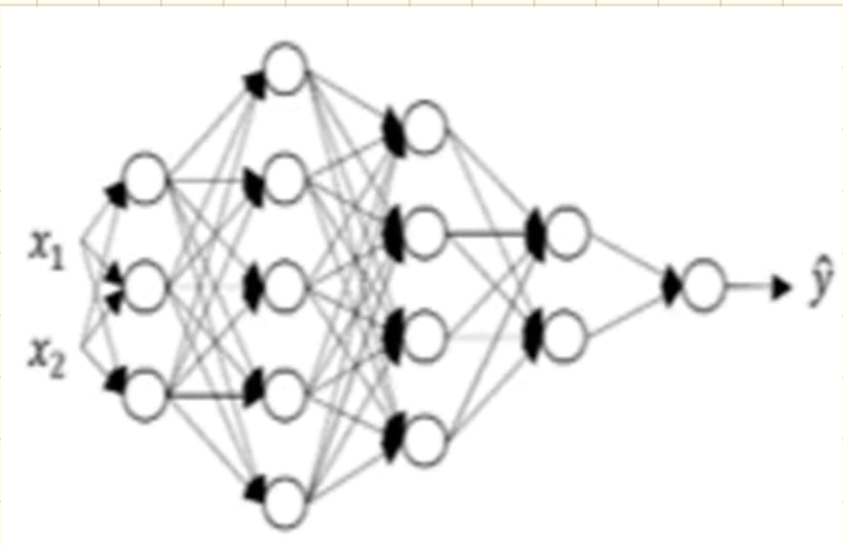
## Circuit theory and deep learning

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.

직관 2) 순환이론에 따르면, 얕은 네트워크보다 깊은 네트워크에서 더 계산하기 쉬운 함수가 있다.

# <Building Blocks of Deep Neural Networks>

* Forward and Backward functions



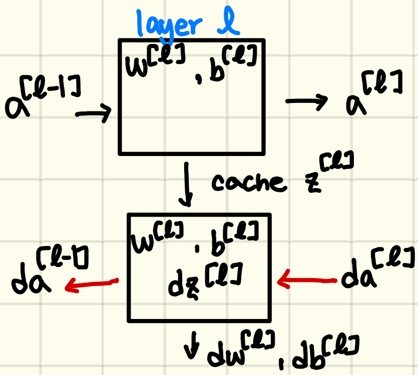

Layer $l$ : $W^{[l]}, b^{[l]}$

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

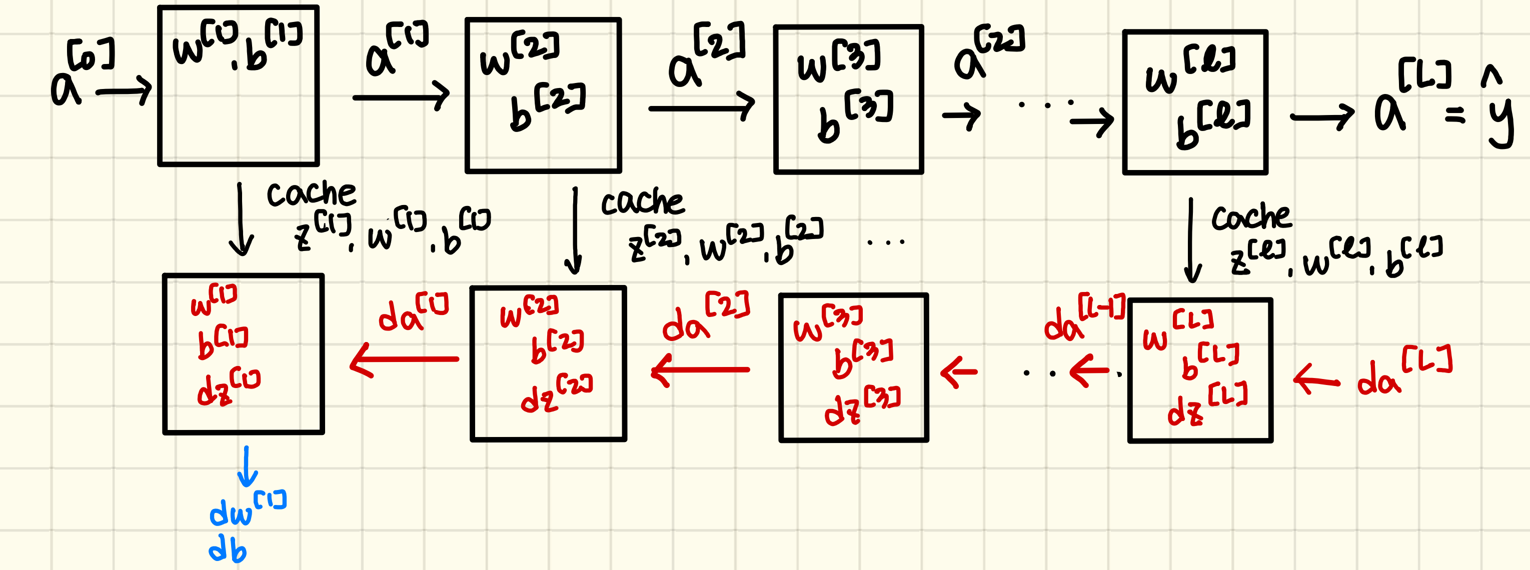
cache $z^{[l]}$

Backward : Input $da^{[l]}$

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



확장해보면,




# < Foward and Backward Propagation >

## * Forward propagation for layer $l$

- Input : $a^{[l-1]}$
- Output : $a^{[l]}$, cache $(z^{[l]})$ ⊕ coding 관점에서 cache $w^{[l]}, b^{[l]}$

$$z^{[l]} = w^{[l]} \cdot a^{[l-1]} + b^{[l]}$$
$$a^{[l]} = g^{[l]}(z^{[l]})$$

vectorize

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

## * Backward propagation for layer $l$

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

$$dz^{[l]} = da^{[l]} * g^{[l]\prime}(z^{[l]})$$
$$dw^{[l]} = dz^{[l]} \cdot a^{[l-1]T}$$
$$db^{[l]} = dz^{[l]}$$
$$da^{[l]} = W^{[l]T} \cdot dz^{[l]}$$
$$\rightarrow dz^{[l]} = W^{[l+1]T} \cdot dz^{[l+1]} * g^{[l]\prime}(z^{[l]})$$

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

# < Parameters vs Hyperparameters >

* What are hyperparameters?

  - parameters : $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, \dots$
  - Hyperparameters : learning rate $\alpha$
    # iterations
    # hidden layer $L$
    # hidden units $n^{[1]}, n^{[2]}, \dots$
    Choice of activation functions
    ...