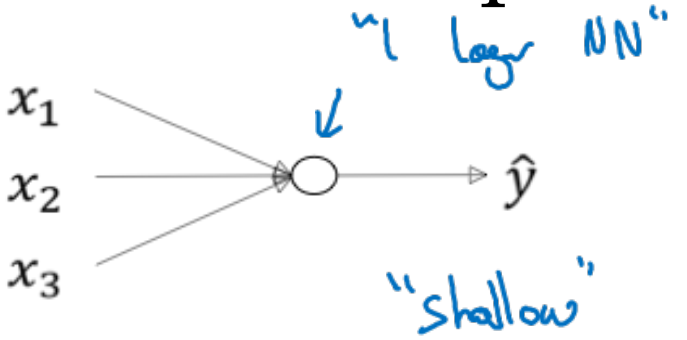
Copyright Notice

deeplearning.ai

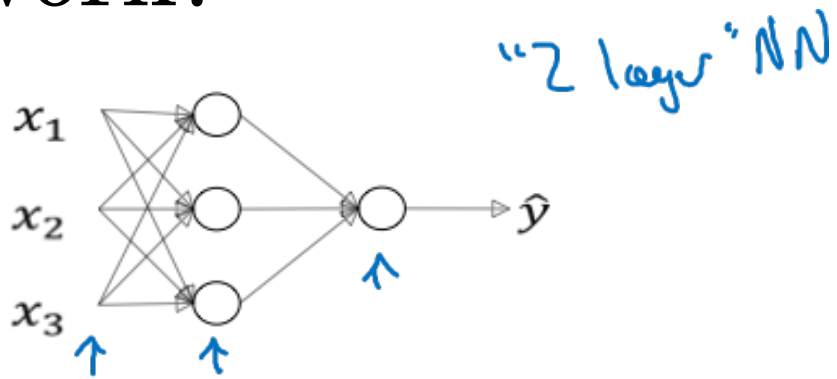
Deep Neural Networks

Deep L-layer Neural network

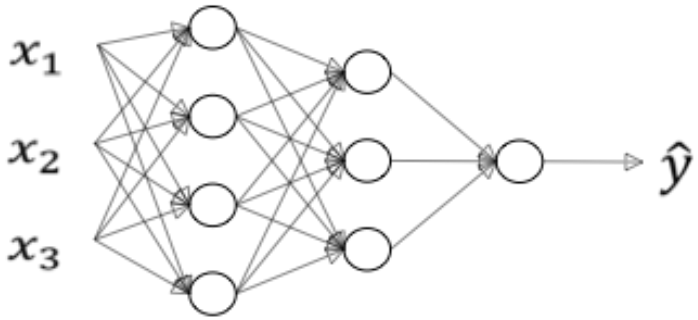
What is a deep neural network?



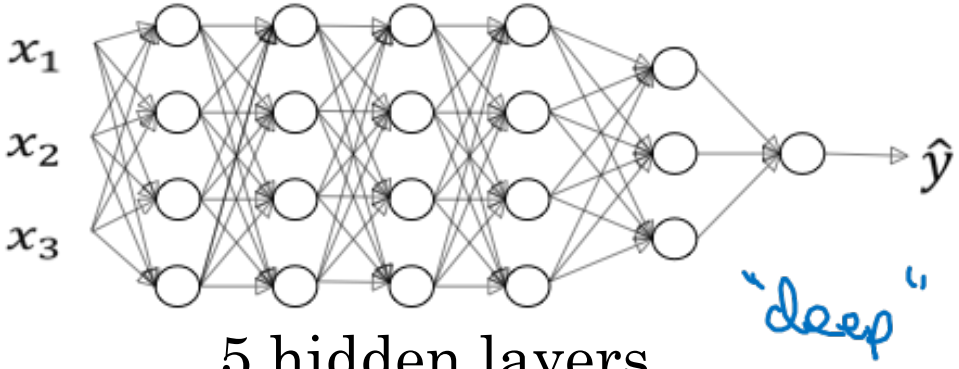
logistic regression



1 hidden layer



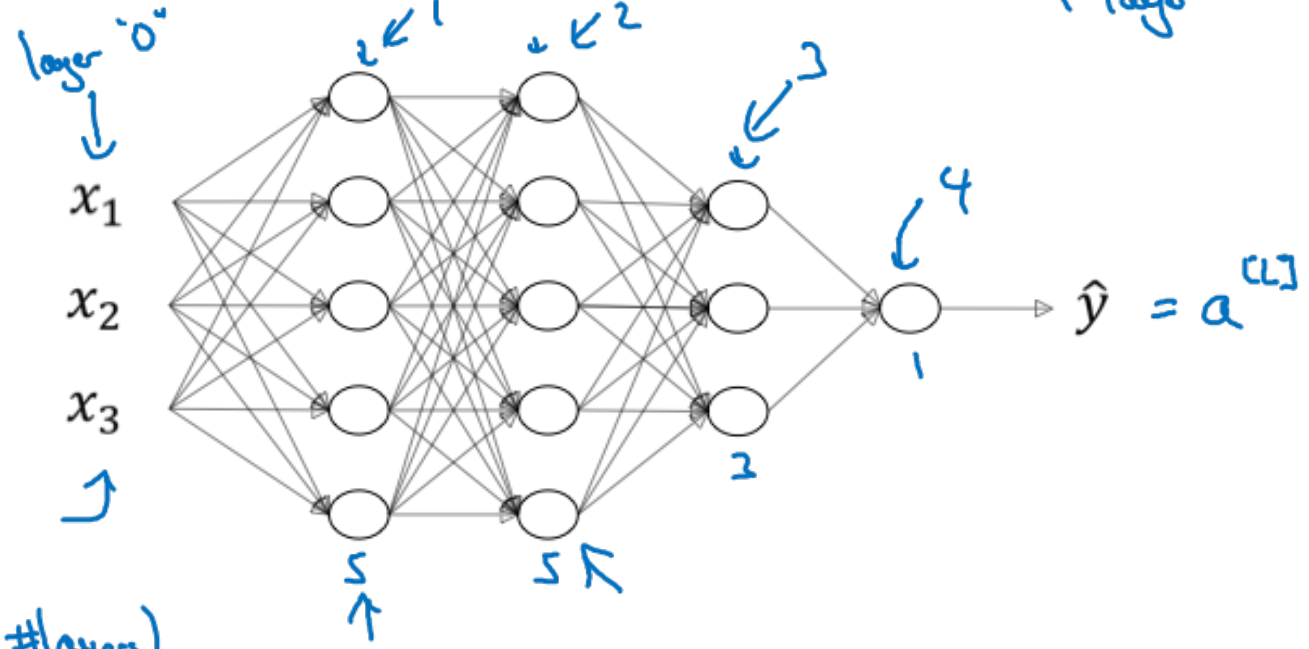
2 hidden layers



5 hidden layers

Andrew

Deep neural network notation



$L = 4$ (#layers)

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

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

$n^{[0]} = n_x = 3$

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

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

Andrew

1단계: 순전파 (Forward Propagation) 단계
: 이 단계에서는 입력된 데이터가 각자의
신경망 층을 지나가며,
결과적으로 데이터에 대한 기계의 예상치인
Y-hat 을 산출하는 과정입니다.

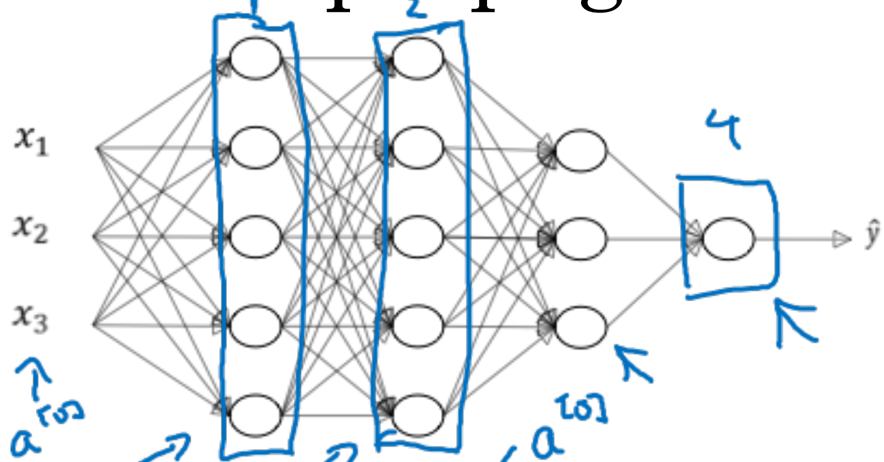
2단계: 역전파 (Backward Propagation) 단계
이 단계는, 1단계에서 산출된 Y-hat 을 통해서
나온 (결과와 예상의 차이를 의미하는) 손실 (loss)
값을 통해서, 손실 (loss)을 최소화하는 좋은
방법인 경사하강법 (Gradient Descent)을
사용해서 매개변수 w, b 를 수정하는 단계입니다.
경사하강법을 통해서 매개변수를 수정함으로써,
손실 (loss)은 점점
작아지게 된다.

Deep Neural Networks

Forward Propagation in a Deep Network

deeplearning.ai

Forward propagation in a deep network



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

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

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

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

$z^{[4]} = w^{[4]} a^{[3]} + b^{[4]}$, $a^{[4]} = g^{[4]}(z^{[4]}) = \hat{y}$

$\left[z^{[1](1)}, z^{[1](2)}, \ldots, z^{[1](m)} \right]$

$A^{[0]} = X$

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

Vectorized:

$X = A^{[0]}$

$z^{[1]} = w^{[1]} A^{[0]} + b^{[1]}$

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

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

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

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

for $l = 1..4$

z[1]
: 활성화 함수의 입력값

W[1]
: 입력데이터와 1층 사이 가중치 w를 모아놓은 벡터 (=배열, 행렬)

a[0]
: 입력데이터 x를 모아놓은 벡터

b[1]:
1층의 퍼셉트론들의 편향 b를 모아놓은 벡터

A[1]
: 1층에서 활성화 함수 결과값들을 모아놓은 벡터 (=배열, 행렬)

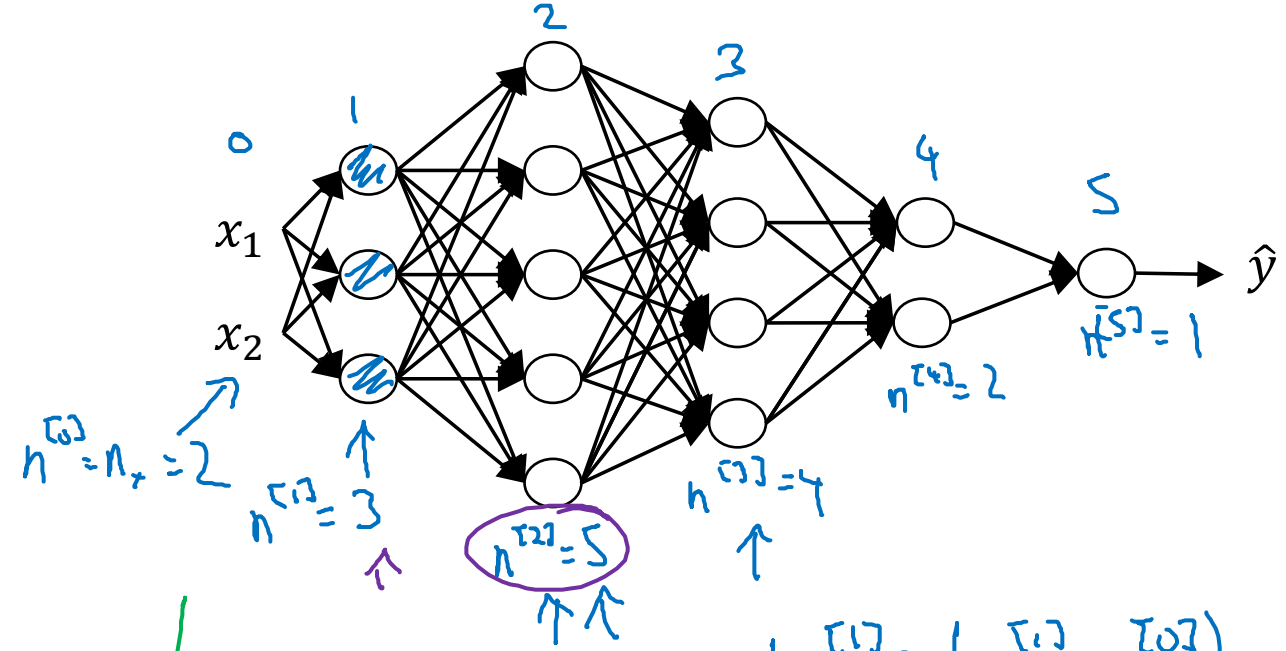
Andrew N

deeplearning.ai

Deep Neural Networks

Getting your matrix dimensions right

Parameters $W^{[l]}$ and $b^{[l]}$



$L = 5$

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

$a^{[l]}$

- $w^{[l]} : (n^{[l]}, n^{[l-1]})$
- $b^{[l]} : (n^{[l]}, 1)$
- $dw^{[l]} : (n^{[l]}, n^{[l-1]})$
- $db^{[l]} : (n^{[l]}, 1)$

$z^{[1]} = w^{[1]} \cdot x + b^{[1]}$

$(3,1) \leftarrow (3,2)\ (2,1) \quad (3,1)$

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

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

$w^{[2]} : (5,3) \quad (n^{[2]}, n^{[1]})$

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

$(5,1) \leftarrow (5,3)\ (3,1) \quad (5,1)$

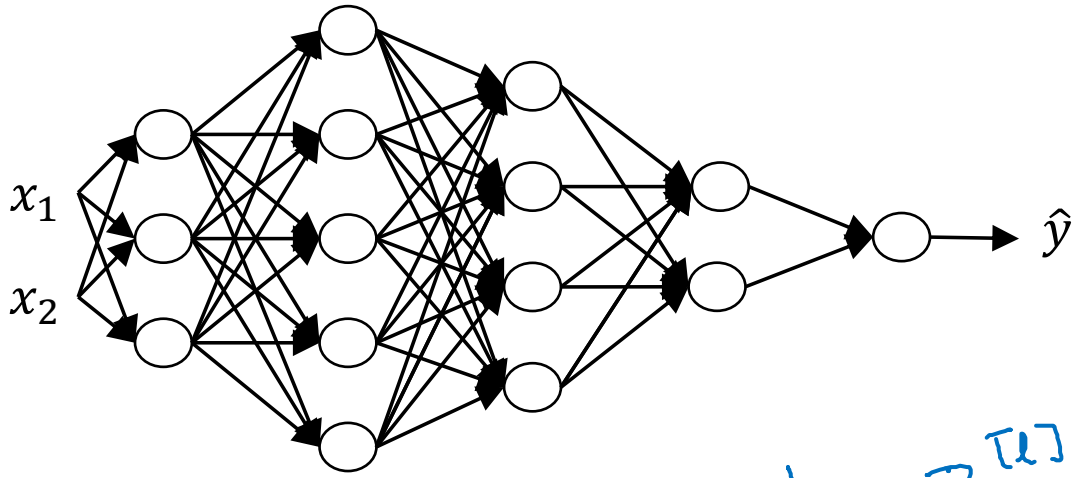
$(n^{[2]}, 1) \quad (n^{[2]}, n^{[1]})\ (n^{[1]}, 1) \quad (n^{[2]}, 1)$

$w^{[3]} : (4,5)$

$w^{[4]} : (2,4)$, $w^{[5]} : (1,2)$

Andrew Ng

Vectorized implementation



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

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

$$\begin{bmatrix} z^{[1](1)} & z^{[1](2)} & \cdots & z^{[1](m)} \end{bmatrix}$$

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

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

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

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

$l = 0 \quad A^{[0]} = X = (n^{[0]}, m)$

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

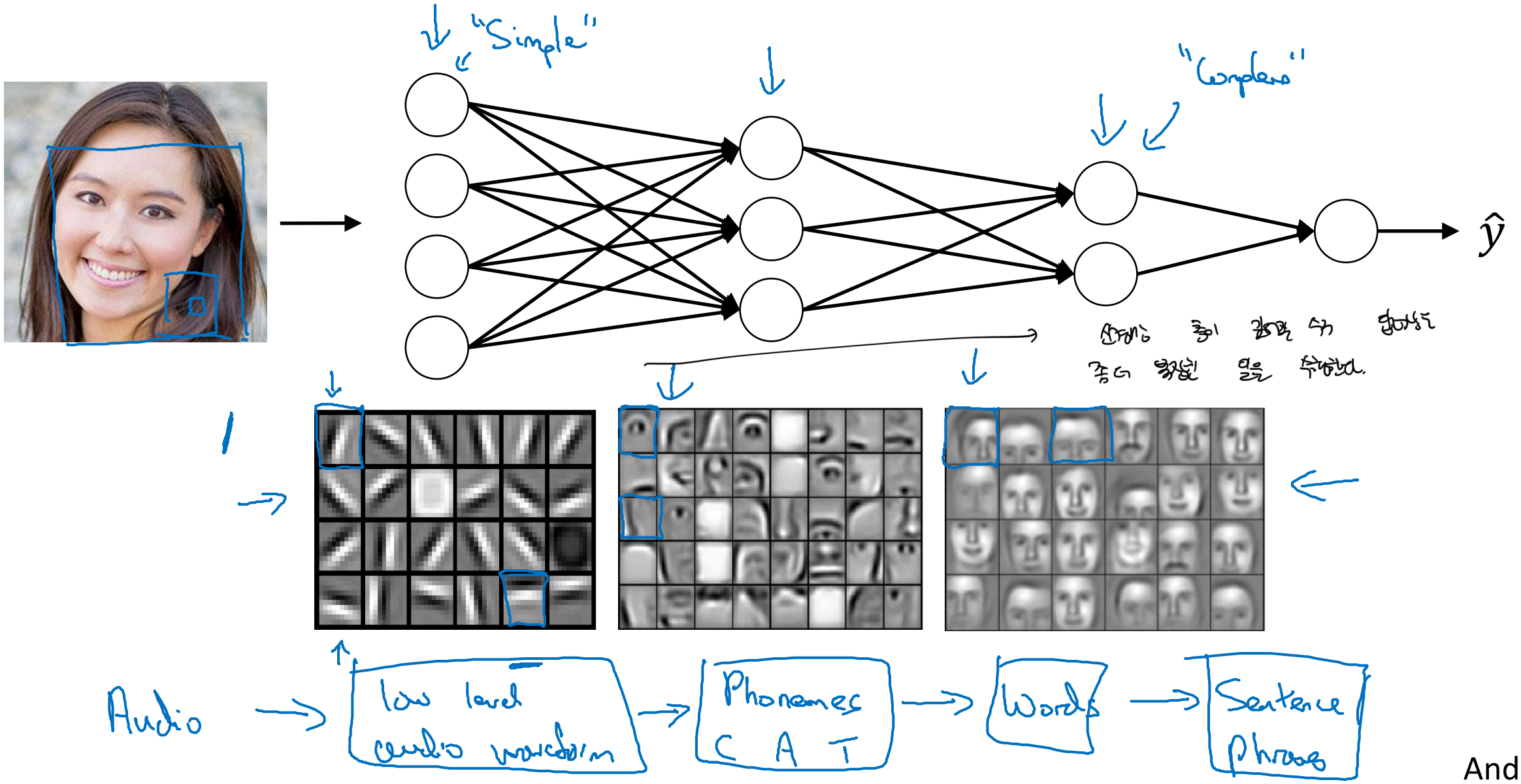
Andrew Ng

deeplearning.ai

Deep Neural Networks

Why deep representations?

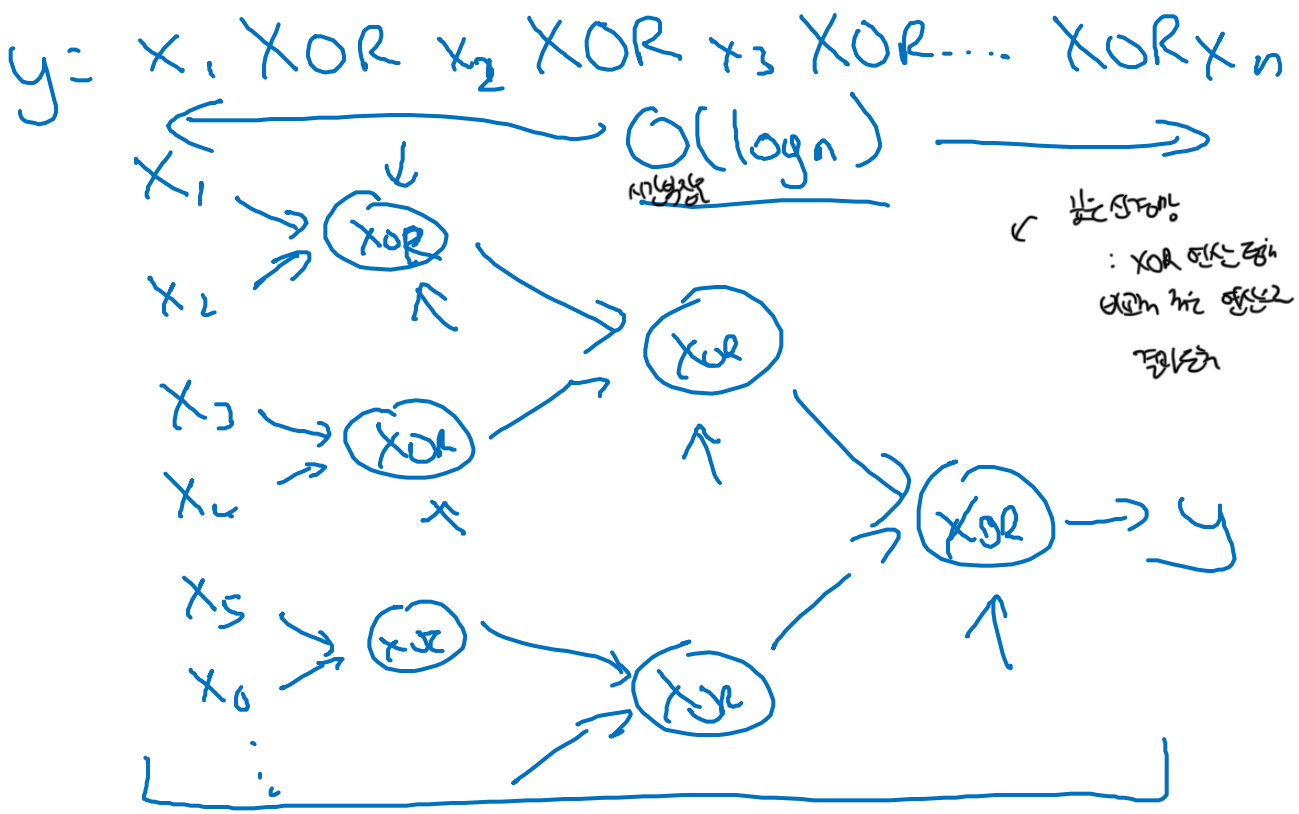
Intuition about deep representation

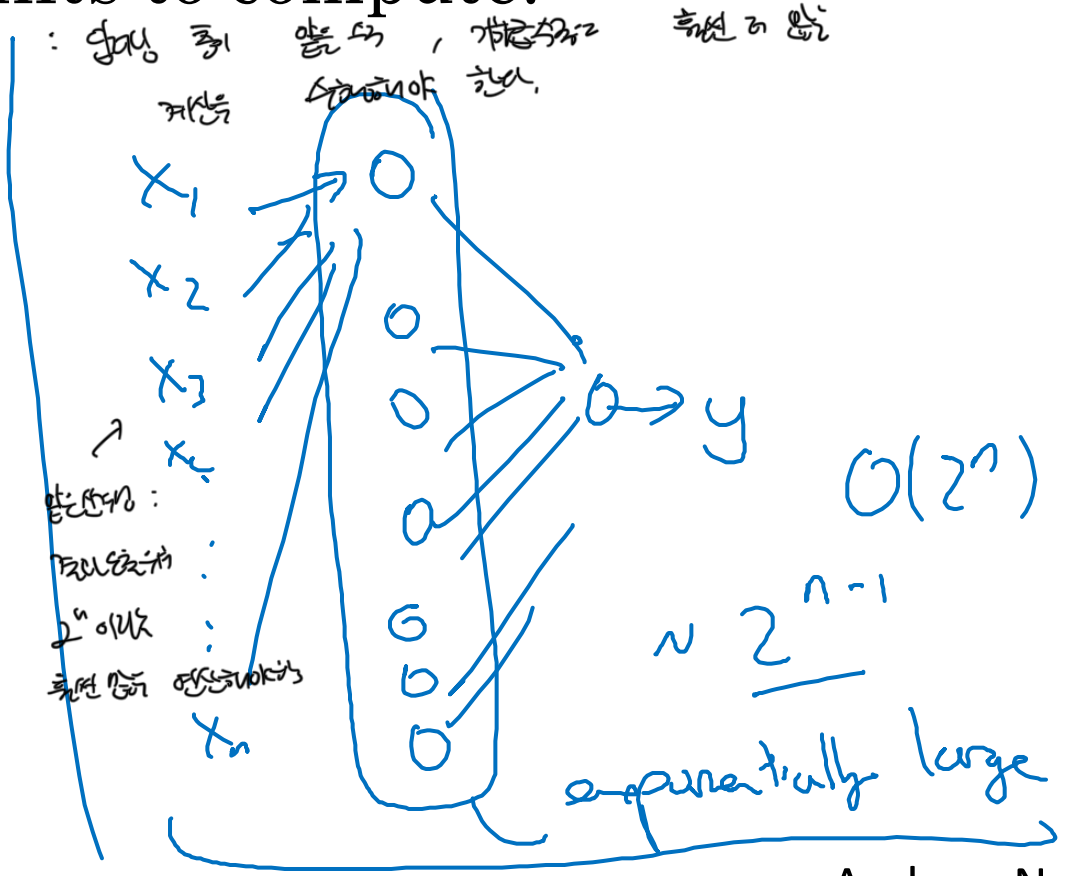


Andrew Ng

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.



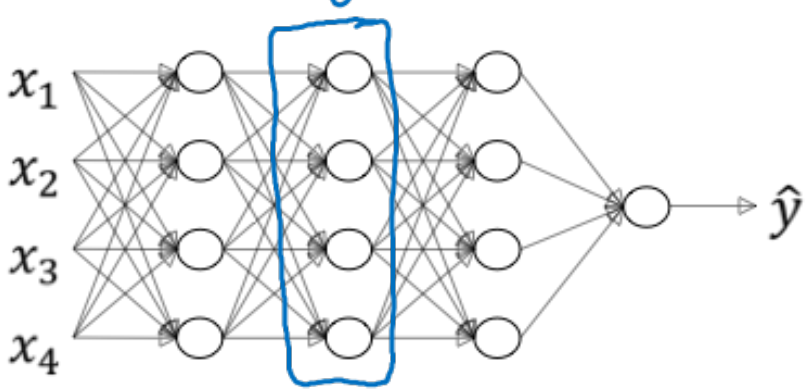


deeplearning.ai

Deep Neural Networks

Building blocks of deep neural networks

Forward and backward functions



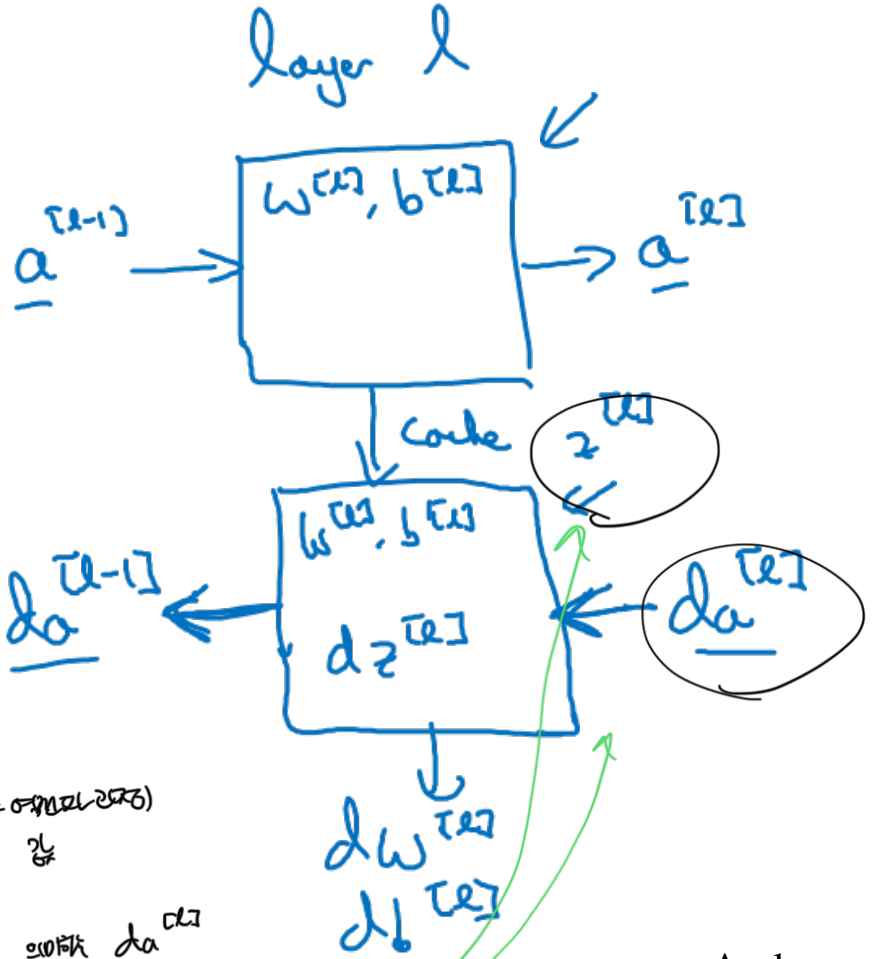
Layer l: $W^{[l]}, b^{[l]}$

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

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

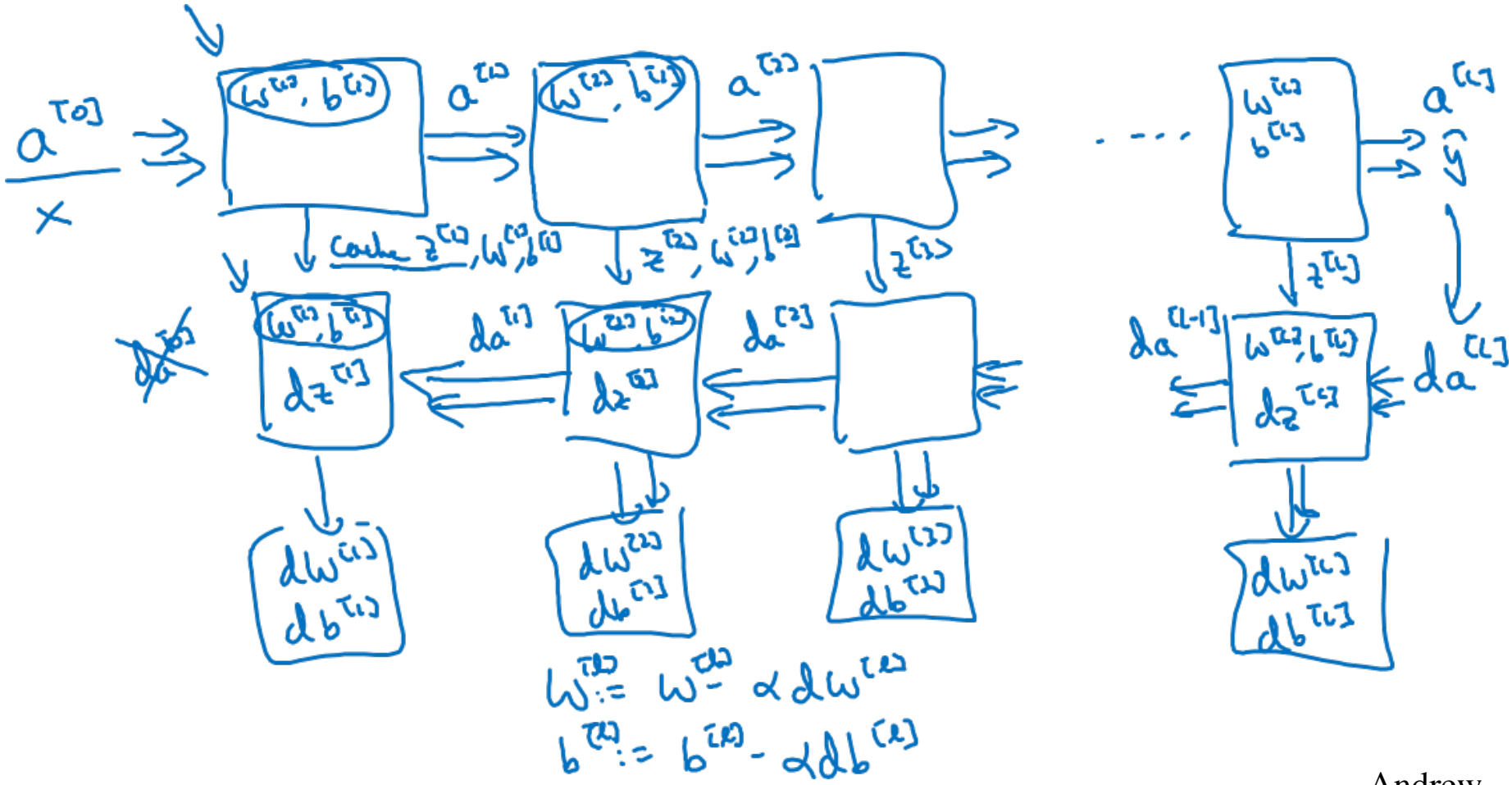
$a^{[l]} = g^{[l]}(z^{[l]})$

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



Andrew

Forward and backward functions



Andrew

deeplearning.ai

Deep Neural Networks

Forward and backward propagation

Backward propagation for layer l

→ Input $da^{[l]}$

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

"$da^{[l]}$를 입력하면 $da^{[l-1]}$을 출력하고 매개변수 w, b를 업데이트를 출력하는 것"

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

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

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

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

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

역전파 과정에서 L층의 $w^{[l]}$, $b^{[l]}$ 값을 업데이트하는 방식은 도함수 값 $dw^{[l]}$, $db^{[l]}$을 구하여 업데이트 해주는 방식

ex) 새로운 $w^{[l]}$ = 원래 $w^{[l]}$ − $dw^{[l]}$

그다음 신경망을 출력으로 역전파 과정을 수행하기 위해 $da^{[l-1]}$ 값도 구해준다.

구하고자 하는 $da^{[l-1]}$, $dw^{[l]}$, $db^{[l]}$ 값을 구하기 위해서는 "$dz^{[l]}$" 값이 꼭 필요하기에 순전파 과정에서 저장했던 $z^{[l]}$와 순전파 계산 과정에서 구한 $da^{[l]}$을 통해 $dz^{[l]}$ 값을 구한다.

← 훈련 데이터 1개일 때

$$dZ^{[l]} = dA^{[l]} * g^{[l]\prime}(Z^{[l]})$$

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

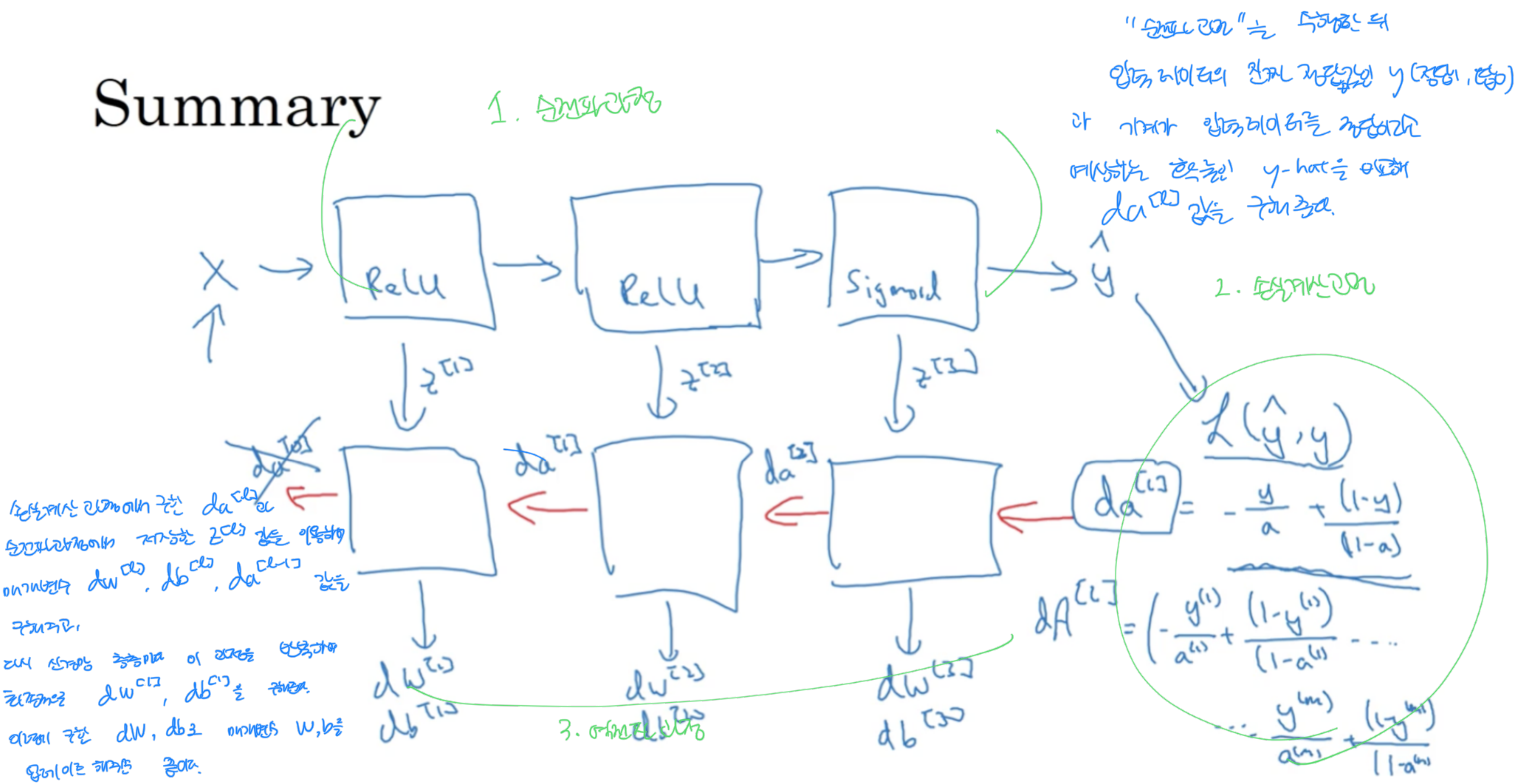
$$db^{[l]} = \frac{1}{m} np.sum(dZ^{[l]}, axis=1, keepdims=True)$$

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

훈련 데이터가 m(여러 개)일 때 행렬 · 벡터 적용

m을 나눠주는 이유: $Z^{[l]}$을 구하는 과정에서 각각의 데이터의 손실 값을 평균하여 구하기 때문.

Summary


1. 순전파 과정
2. 손실계산 과정
3. 역전파 과정
"순전파 과정"을 수행한 뒤
입력 레이어의 진짜 정답값인 y (정답, 레이블)
와 기계가 입력레이어를 참고하고
예상하는 결과값인 y-hat을 비교해
da[L] 값을 구해준다.
X
ReLU
ReLU
Sigmoid
ŷ
z[1]
z[2]
z[3]
da[0]
da[1]
da[2]
da[L]
dw[1]
db[1]
dw[2]
dw[3]
db[3]
$\mathcal{L}(\hat{y}, y)$
$da^{[L]} = -\frac{y}{a} + \frac{(1-y)}{(1-a)}$
$dA^{[L]} = \left(-\frac{y^{(1)}}{a^{(1)}} + \frac{(1-y^{(1)})}{(1-a^{(1)})} \cdots -\frac{y^{(m)}}{a^{(m)}} + \frac{(1-y^{(m)})}{(1-a^{(m)})}\right)$
손실계산 과정에서 구한 da[L]와
순전파 과정에서 저장한 z[L] 값을 이용하여
매개변수 dw[L], db[L], da[L-1] 값을
구해주고,
다시 신경망 층층마다 이 과정을 반복하여
최종적으로 dw[1], db[1]을 구해준다.
이렇게 구한 dW, db로 매개변수 W, b를
업데이트 해주면 끝이다.

deeplearning.ai

Deep Neural Networks

Parameters vs Hyperparameters

What are hyperparameters?

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

Hyperparameters: learning rate α

#iterations

#hidden layers L

#hidden units $n^{[1]}, n^{[2]}, \ldots$

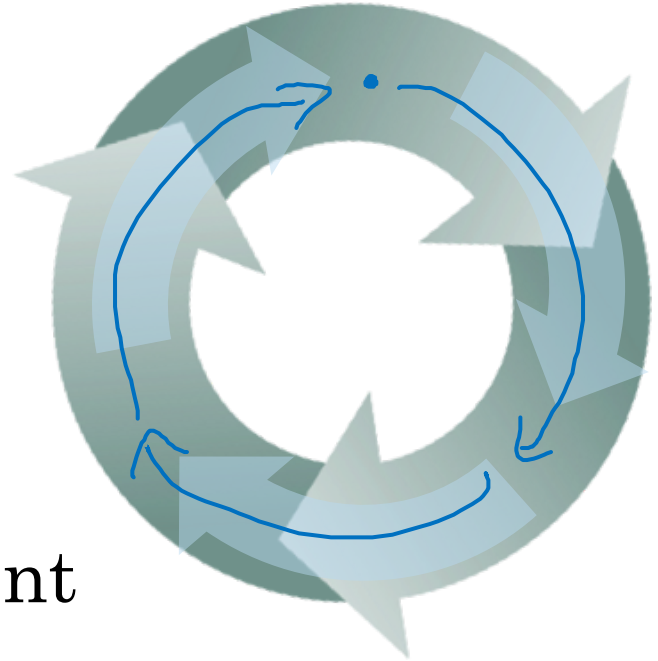
choice of activation function

Later: Momentum, mini-batch size, regularizations, ...

Applied deep learning is a very empirical process

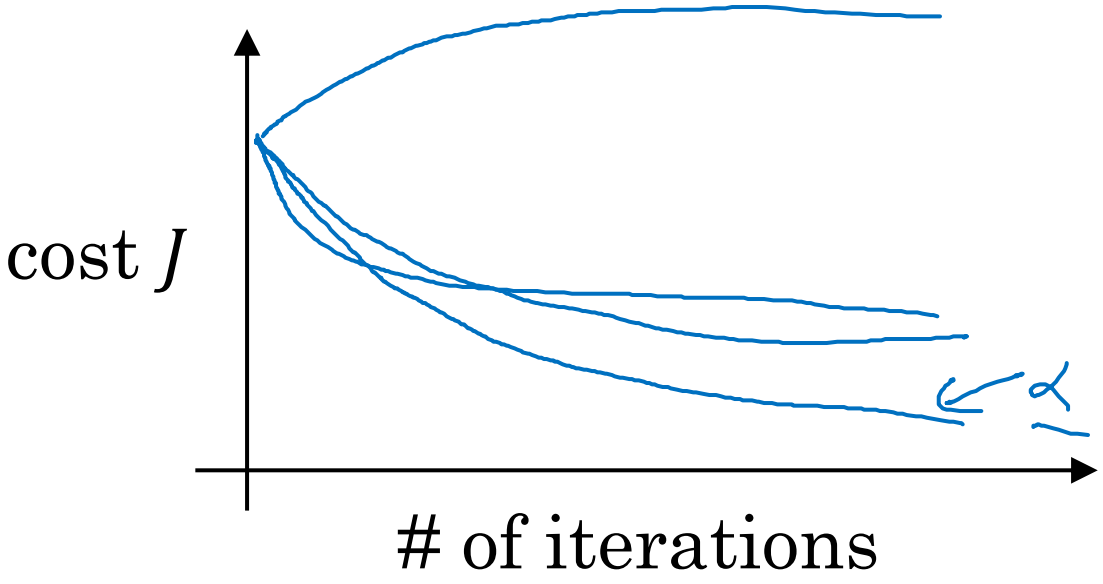
Idea

$\alpha = 0.04$
0.05



Experiment

Code



Vision, Speech, NLP, Ad, Search, Recommendation.

deeplearning.ai

Deep Neural Networks

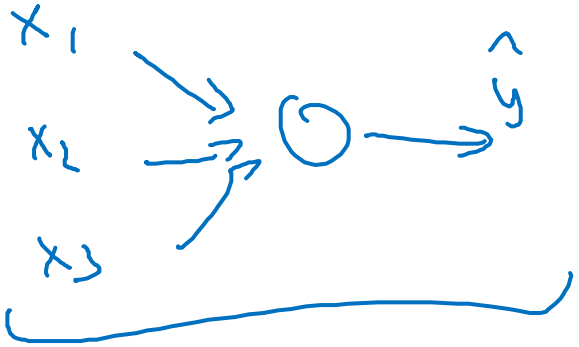
What does this have to do with the brain?

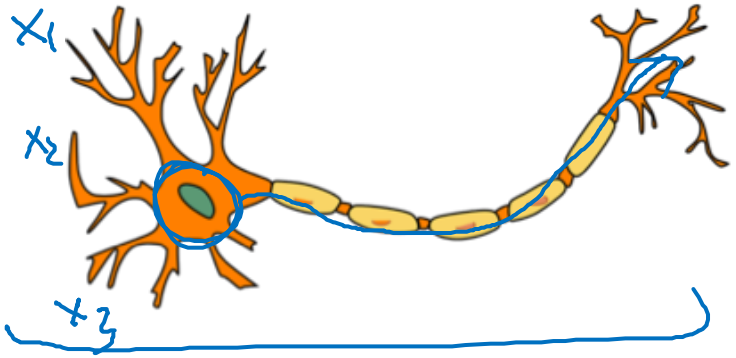
Forward and backward propagation

$$
\begin{aligned}
Z^{[1]} &= W^{[1]}X + b^{[1]} \\
A^{[1]} &= g^{[1]}(Z^{[1]}) \\
Z^{[2]} &= W^{[2]}A^{[1]} + b^{[2]} \\
A^{[2]} &= g^{[2]}(Z^{[2]}) \\
&\vdots \\
A^{[L]} &= g^{[L]}(Z^{[L]}) = \hat{Y}
\end{aligned}
$$

"It's like the brain."

$$
\begin{aligned}
dZ^{[L]} &= A^{[L]} - Y \\
dW^{[L]} &= \frac{1}{m} dZ^{[L]} A^{[L]T} \\
db^{[L]} &= \frac{1}{m} np.\mathrm{sum}(\mathrm{d}Z^{[L]}, axis = 1, keepdims = True) \\
dZ^{[L-1]} &= dW^{[L]T} dZ^{[L]} g'^{[L]}(Z^{[L-1]}) \\
&\vdots \\
dZ^{[1]} &= dW^{[L]T} dZ^{[2]} g'^{[1]}(Z^{[1]}) \\
dW^{[1]} &= \frac{1}{m} dZ^{[1]} A^{[1]T} \\
db^{[1]} &= \frac{1}{m} np.\mathrm{sum}(\mathrm{d}Z^{[1]}, axis = 1, keepdims = True)
\end{aligned}
$$





Andrew Ng