Deep Networks

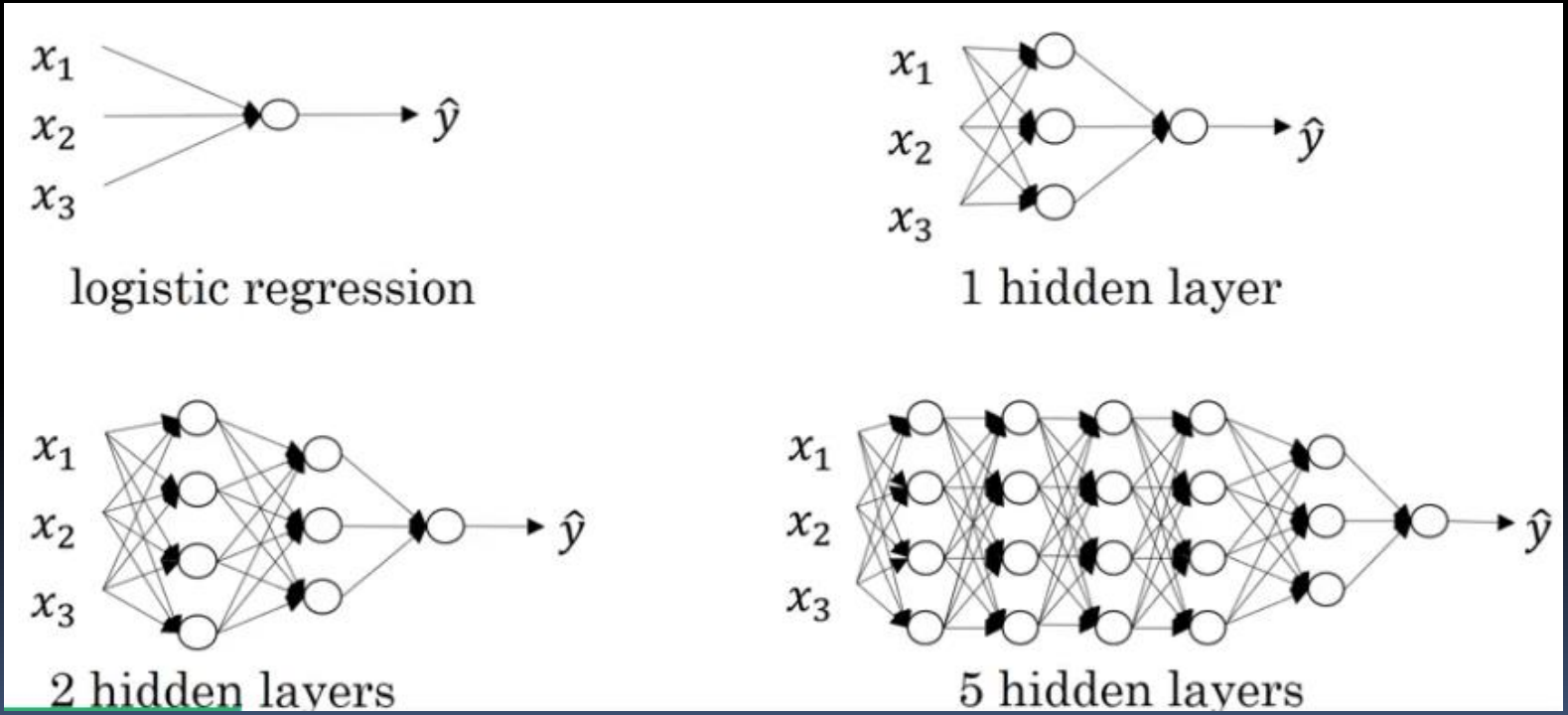
CSCI 447/547 MACHINE LEARNING

Outline

- Deep L-Layer Network
- Forward Propagation
- Matrix Dimensions
- Why Deep Representation?
- Building Blocks
- Back Propagation
- Parameters vs. Hyperparameters
- Brain Analogy

What is a Deep Network?

- A matter of degree



How Many Layers?

- Hard to predict how deep your network needs to be
  - Determine through experimentation

Notation

L = Number of layers

$n^{[l]}$ = number of nodes in layer l

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

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

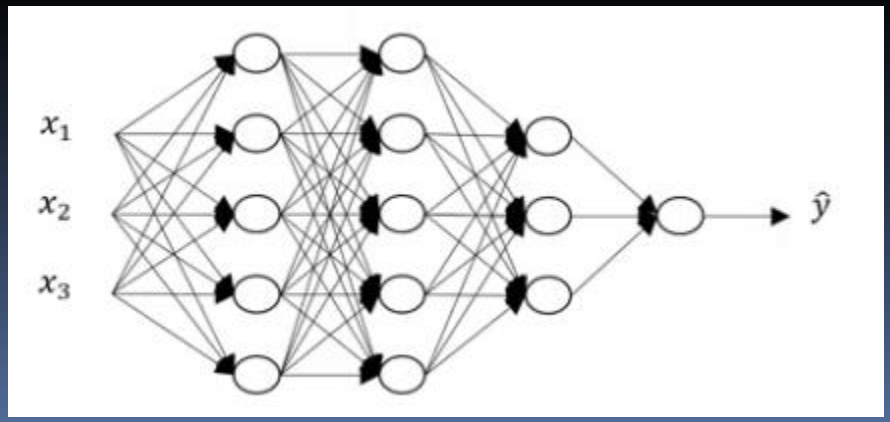
$a^{[0]} = x$ = input

$a^{[L]} = \hat{y}$ = output

Forward Propagation

- Single example

$$
\begin{aligned}
&x = a^{[0]}, \text{ so} \\
&z^{[1]} = W^{[1]} a^{[0]} + b^{[1]} \\
&a^{[1]} = g^{[1]}(z^{[1]}) \\
&z^{[2]} = W^{[2]} a^{[1]} + b^{[2]} \\
&a^{[2]} = g^{[2]}(z^{[2]}) \\
&\cdots \\
&z^{[L]} = W^{[L]} a^{[L-1]} + b^{[L]} \\
&a^{[L]} = g^{[L]}(z^{[L]})
\end{aligned}
$$



General case:

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

Vectorization

- Essentially replace lower case z and a with vectorized uppercase Z and A

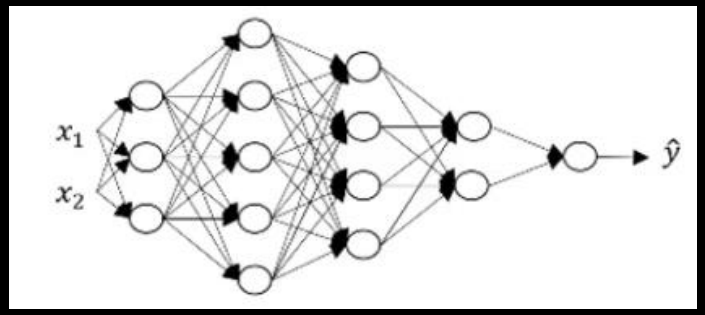
$$
\begin{aligned}
Z^{[1]} &= W^{[1]} A^{[0]} + b^{[1]} \\
A^{[1]} &= g^{[1]}(Z^{[1]}) \\
Z^{[2]} &= W^{[2]} A^{[1]} + b^{[2]} \\
A^{[2]} &= g^{[2]}(Z^{[2]}) \\
&\text{and} \\
\hat{Y} &= g(Z^{[L]}) = A^{[L]}
\end{aligned}
$$

Forward Propagation

- Will need to do run the general equations in a loop, once for each layer (can't vectorize this part)

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

Matrix Dimensions



- Looking at the example above, not vectorized…
- $n^{[1]} = 3$, $n^{[2]} = 5$, $n^{[3]} = 4$, $n^{[4]} = 2$, $n^{[5]} = 1$
- $z^{[1]} = W^{[1]}x + b^{[1]}$
- $z^{[1]}$: (3 x 1), or more generally, ($n^{[1]}$ x 1)
- $x=a^{[0]}$: (2x1), or more generally, ($n^{[0]}$ x 1)
- $W^{[1]}$: (3 x 2), or more generally, ($n^{[1]}$ x $n^{[0]}$)
- $b^{[1]}$: (3 x 1), same as $z^{[1]}$

Matrix Dimensions

- In general, not vectorized:

- $W^{[l]}$: $(n^{[l]} \times n^{[l-1]})$
- $b^{[l]}$: $(n^{[l]} \times 1)$
- $a^{[l]}$: $(n^{[l]} \times 1)$
- z[l]: $(n^{[l]} \times 1)$

- $dW^{[l]}$: $(n^{[l]} \times n^{[l-1]})$
- $db^{[l]}$: $(n^{[l]} \times 1)$

Matrix Dimensions

- In general, vectorized:

  m = number of training examples

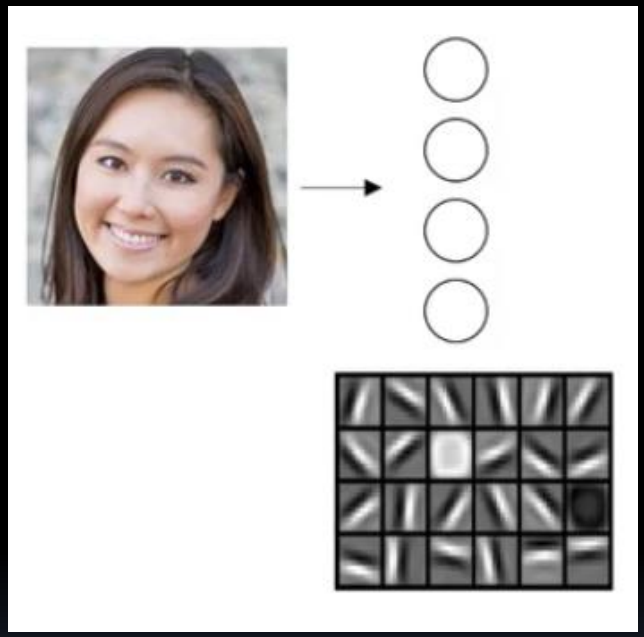
- $A^{[l]}$: $(n^{[l]} \times m)$
- Z[l]: $(n^{[l]} \times m)$

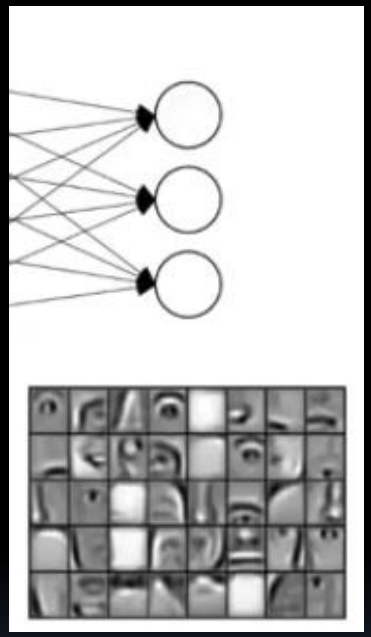
- $dZ^{[l]}$, $dA^{[l]}$: $(n^{[l]} \times m)$

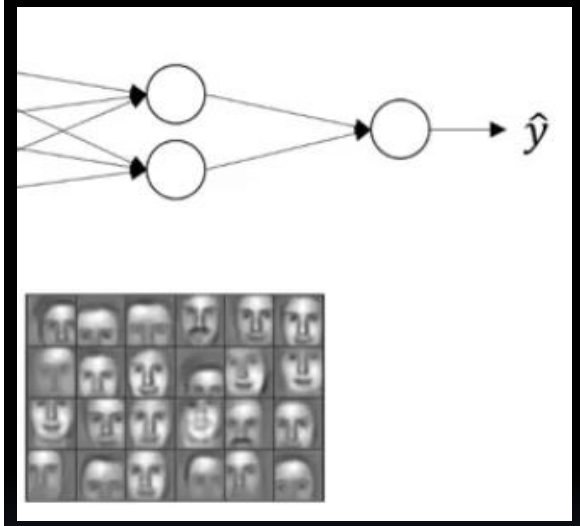
Why Deep Networks?

- Intuition about deep representation
  - Image input for example
    - First layer might be an edge detector
    - Second layer might group edges to find features
    - Third layer might put image features together
  - Earlier layers might be looking at simpler features while later layers looking at more complex functions, composing them

Why Deep Networks?

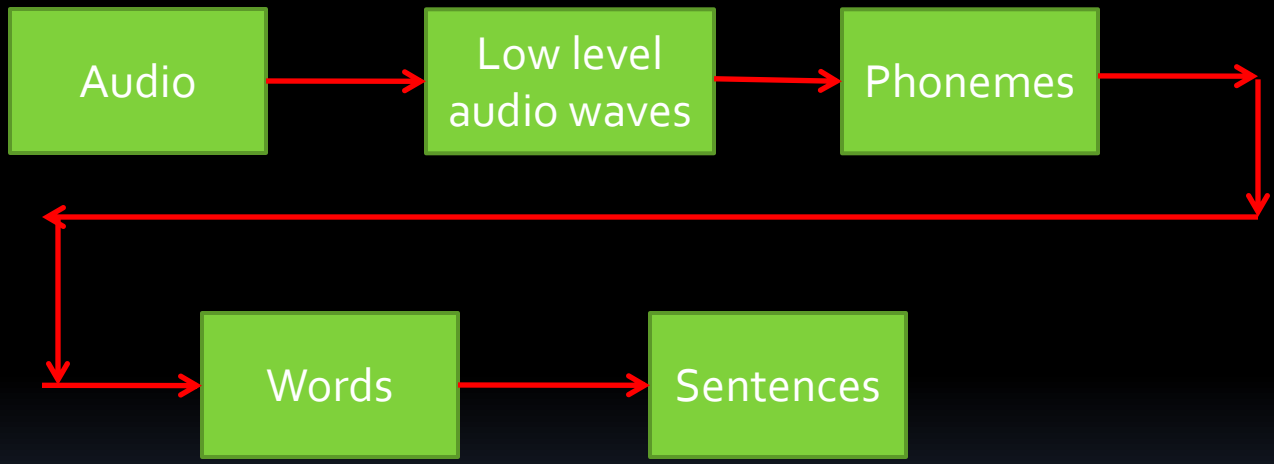






Why Deep Networks?

- Could be similar process for speech recognition



Why Deep Networks?

- Hypothesis that human brain works in a similar manner
  - Detect simple features first and compose them into more complex features

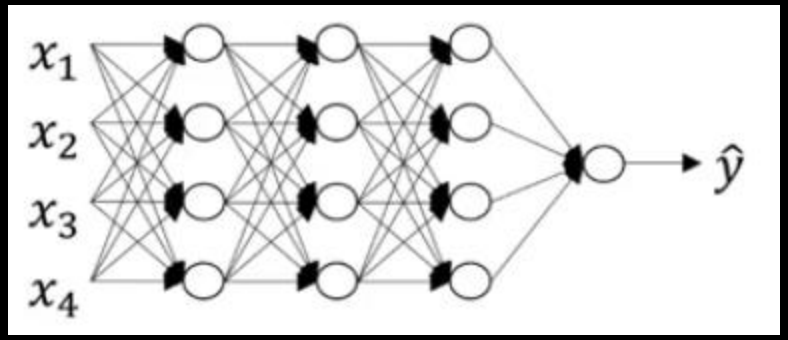
Why Deep Networks?

- Another view – circuit theory
  - 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

Why Deep Networks?

- Really just re-branding
  - Used to be called "neural networks"
    - With many layers
  - But "deep learning" sounds better…

Building Blocks

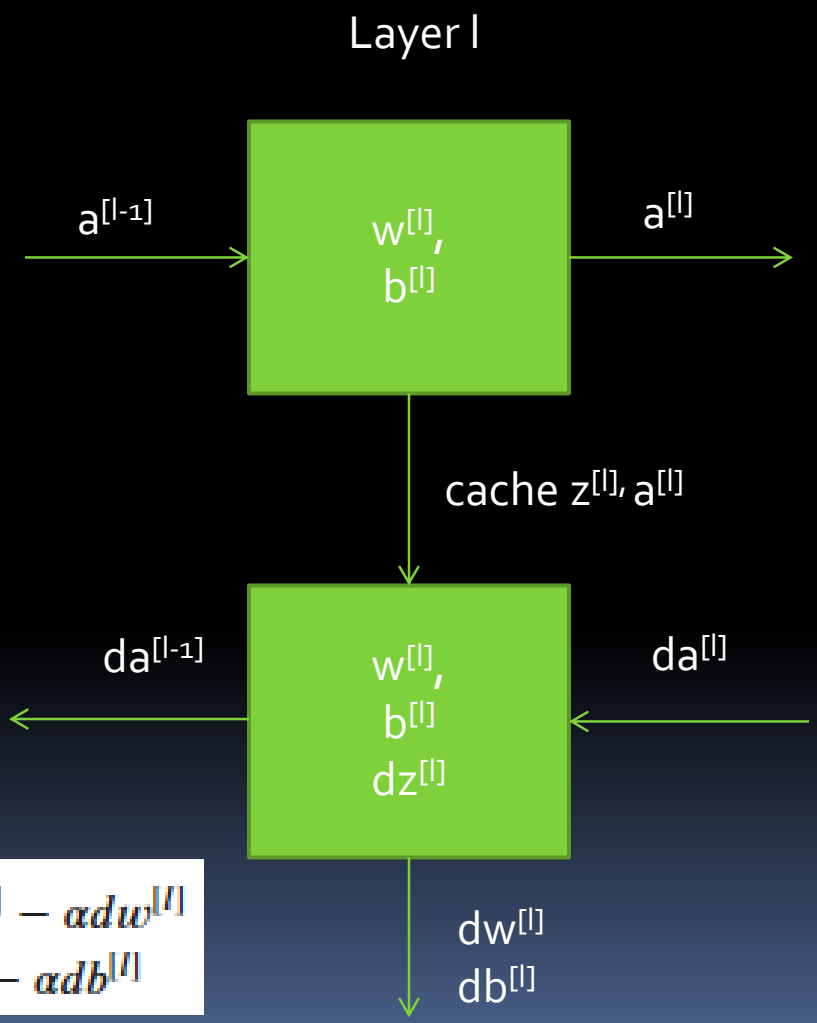


- Looking at computations for just one layer
- Layer l: $W^{[l]}$, $b^{[l]}$
- Forward: Input $a^{[l-1]}$, output $a^{[l]}$
  - Caching $z^{[l]}$ and $a^{[l]}$ will be useful for backward propagation step

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

- Backward: Input $da^{[l]}$ (cached $z^{[l]}$ and $a^{[l]}$), outputs $da^{[l-1]}$ ($dw^{[l]}$, $db^{[l]}$)

Building Blocks



$$w^{[l]} = w^{[l]} - \alpha dw^{[l]}$$
$$b^{[l]} = b^{[l]} - \alpha db^{[l]}$$

Forward and Backward Propagation

- Forward propagation for layer l
  - Input $a^{[l-1]}$
  - Output $a^{[l]}$, cache $a^{[l]}$, $z^{[l]}$, (save $w^{[l]}$ and $b^{[l]}$ also)

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

Processing a single example

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

Processing all examples

Forward and Backward Propagation

- Backward propagation for layer l
  - Input $da^{[l]}$
  - Output $da^{[l-1]}$, $dw^{[l]}$ and $db^{[l]}$

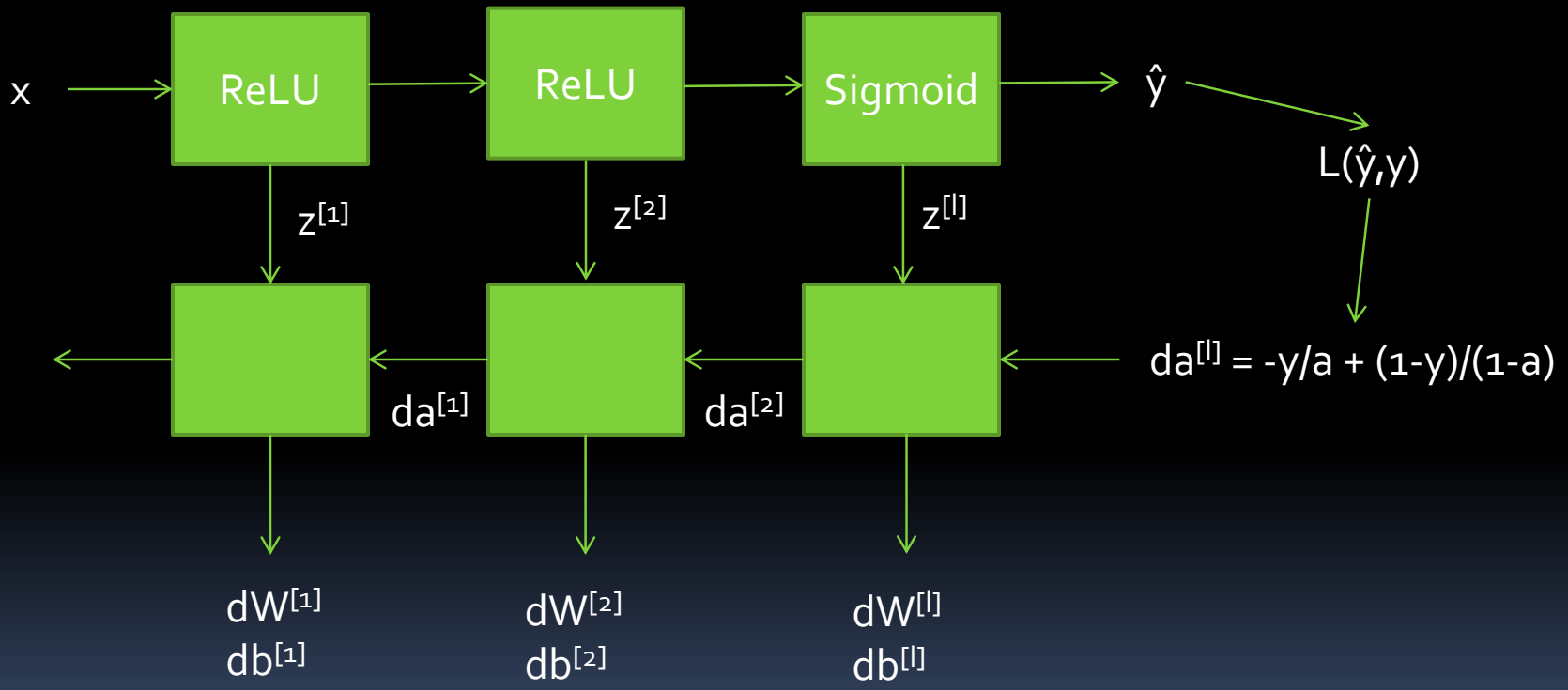
$$dz^{[l]} = da^{[l]} * g^{[l]'}(z^{[l]})$$
$$dW^{[l]} = dz^{[l]} . a^{[l-1]}$$
$$db^{[l]} = dz^{[l]}$$
$$da^{[l-1]} = W^{[l]T}(dz^{[l]})$$
Putting it all together,
$$dz^{[l]} = W^{[l-1T]}dz^{[l-1]} * g^{[l]'}(z^{[l]})$$

Processing a single example

$$dZ^{[l]} = dA^{[l]} * g^{[l]'}(Z^{[l]})$$
$$dW^{[l]} = \frac{1}{m} dZ^{[l]} . A^{[l-1]T}$$
$$db^{[l]} = \frac{1}{m} \text{npsum}(dz^{[l]}, \text{axis=1, keepdims=True})$$
$$dA^{[l-1]} = W^{[l]T}(dZ^{[l]})$$

Processing all examples

Forward and Backward Propagation



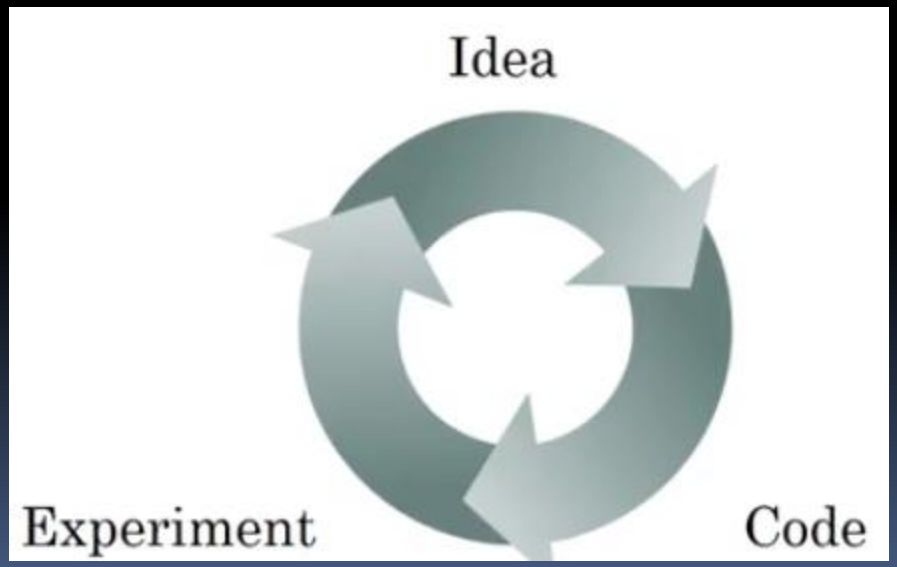
Parameters and Hyperparameters

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

- Hyperparameters
  - Learning rate alpha
  - Number of iterations
  - Number of hidden layers
  - Number of units per layer
  - Choice of activation functions

Parameters and Hyperparameters

- Empirical process
  - Can vary for different domains
  - Can change over time



Braaaaiiiinnnzzz

- Simplistic analogy between logistic regression and a single neuron, but neurons are really much more complex

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

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

Summary

- Deep L-Layer Network
- Forward Propagation
- Matrix Dimensions
- Why Deep Representation?
- Building Blocks
- Back Propagation
- Parameters vs. Hyperparameters
- Brain Analogy