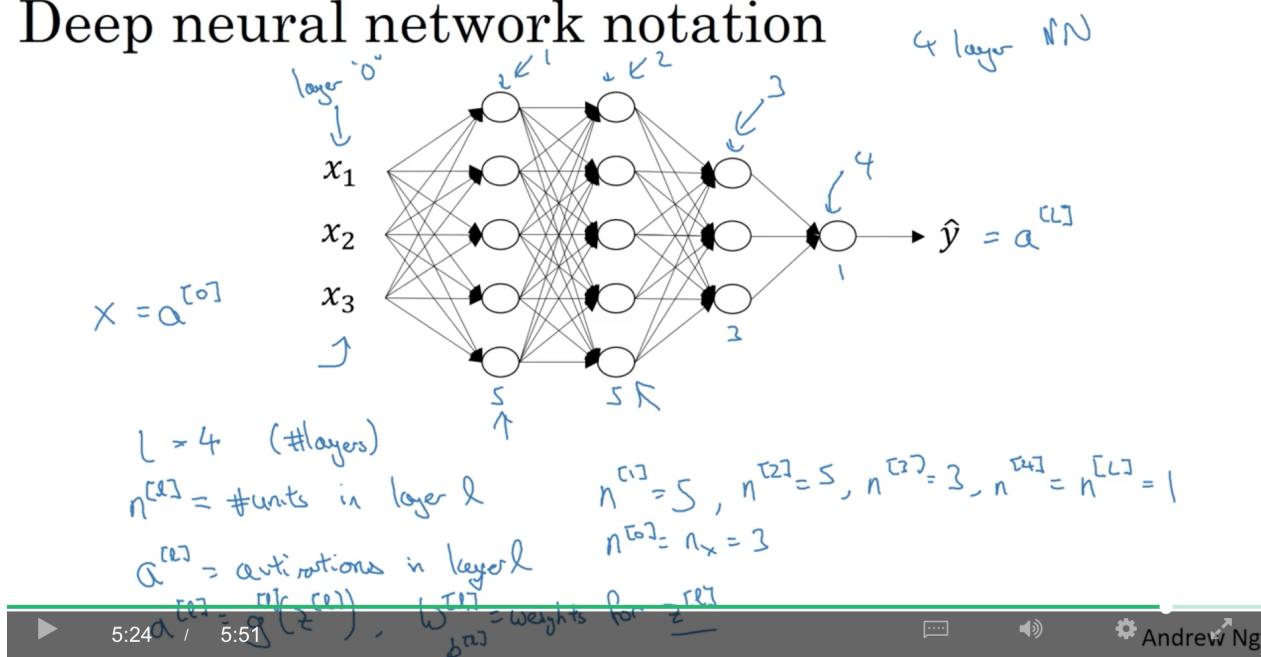


83 lines (46 sloc) 4.15 KB

contents

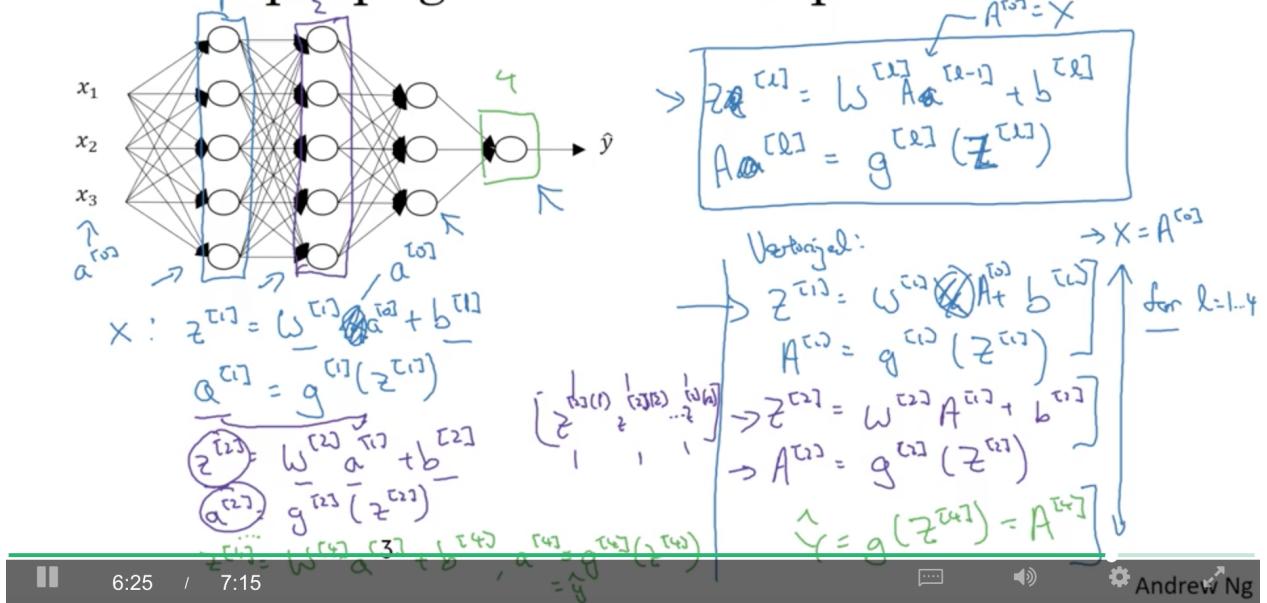
- deep L-layer neural network
- forward propagation in a deep network
- getting your matrix dimensions right
- why deep representations?
- building blocks of deep neural networks
- forward and backward propagation
- parameters vs hyperparameters
- what does this have to do with the brain?

deep L-layer neural network**Deep neural network notation****forward propagation in a deep network**

$z^{[l](i)}$ 表示第 l 层的第 i 个训练样本(列向量), $Z^{[l]}$ 表示将第 l 层的这 m 个训练样本全部水平地放在一起形成的矩阵。以此 vectorization 的方法, 可以避免从 $1 \rightarrow m$ 的这个 for-loop。

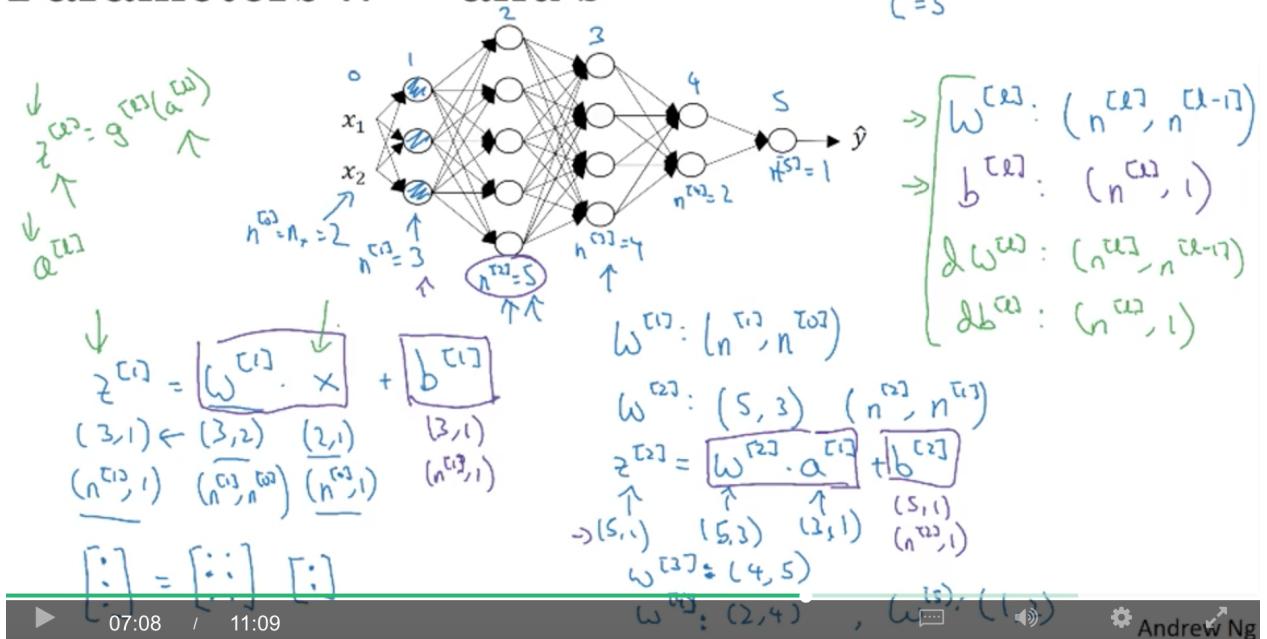
另外, 从第1层到第4层这个for-loop是无法避免的。

Forward propagation in a deep network



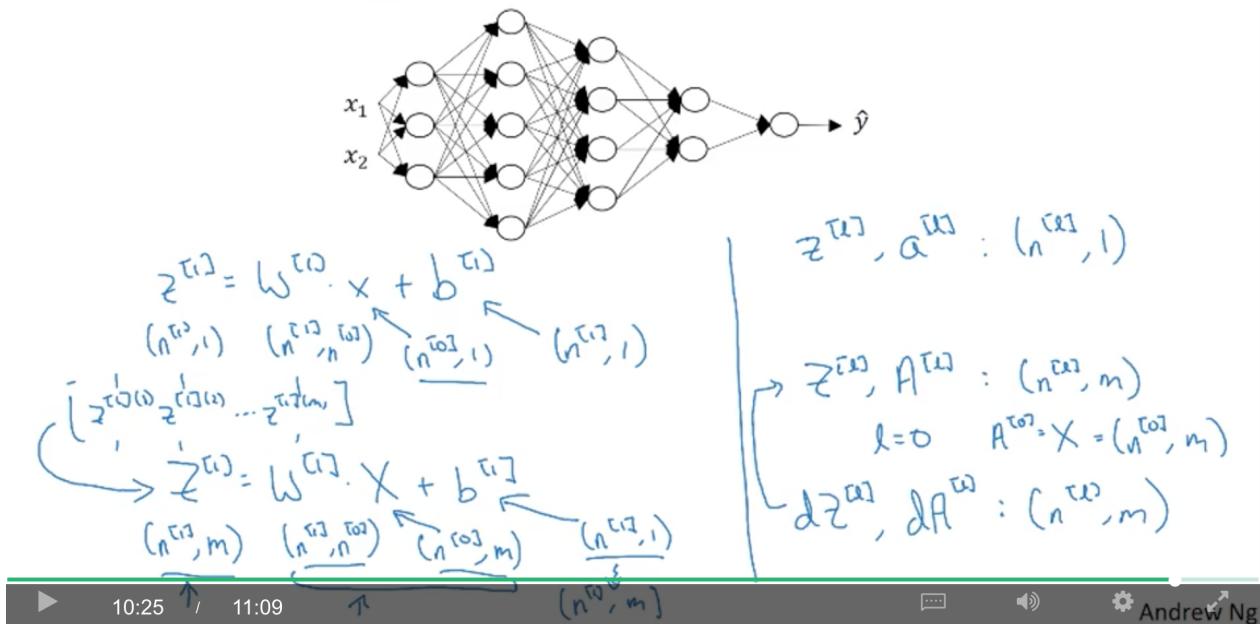
getting your matrix dimensions right

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



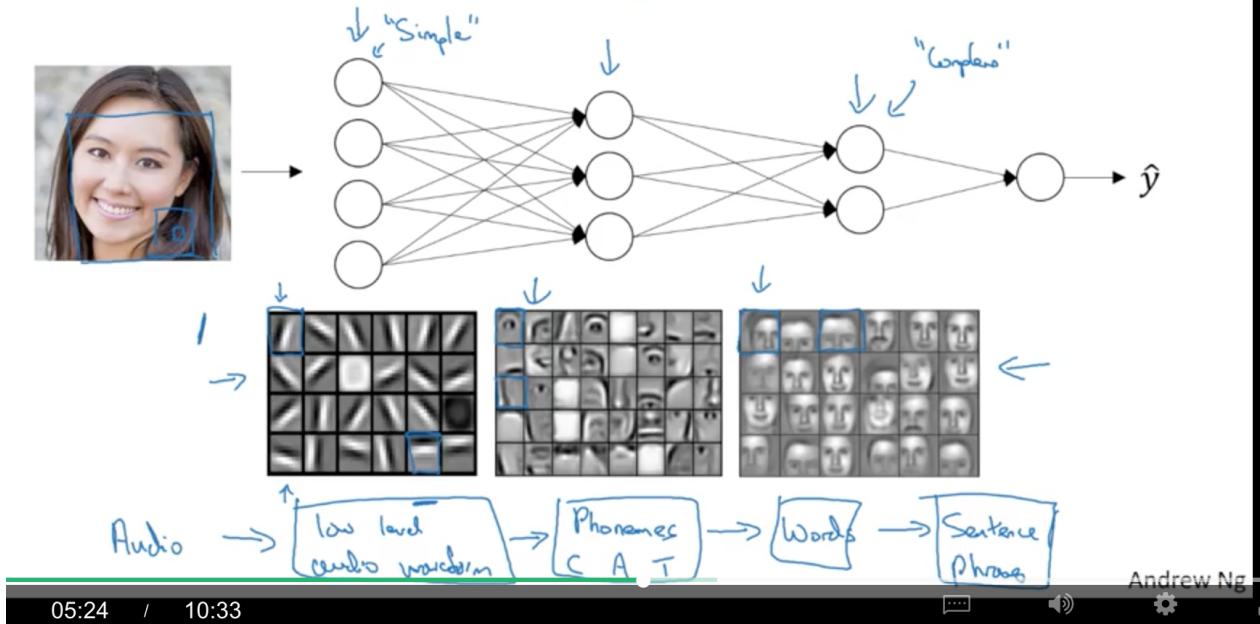
vectorized之后, $b^{[l]}$ 仍然是 $(n^{[l]}, 1)$ 维, 只是因为broadcasting, 才复制m遍, 变成了 $(n^{[l]}, m)$ 维。

Vectorized implementation



why deep representations?

Intuition about deep representation

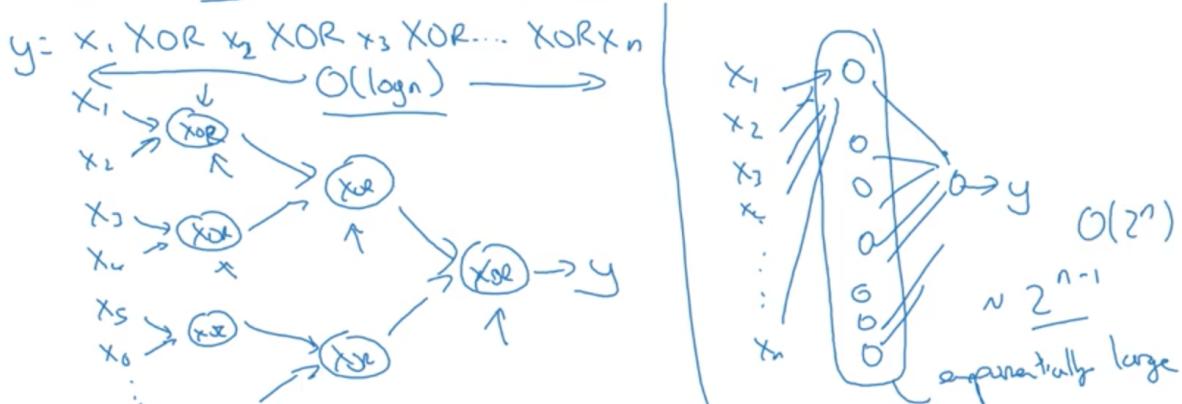


对于 (x_1, x_2, \dots, x_n) 的异或(XOR)操作，如果用树型结构， n 个叶子节点，则树深度是 $\log_2 n + 1$ (深度为 k 的满二叉树的第 i 层上有 2^{i-1} 个结点，总共至多有 $2^k - 1$ 个结点)，即只需要 $O(\log_2 n)$ 层的树就能完成。

而如果采用单隐层，则需要 $2^{(n-1)}$ 个节点

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.



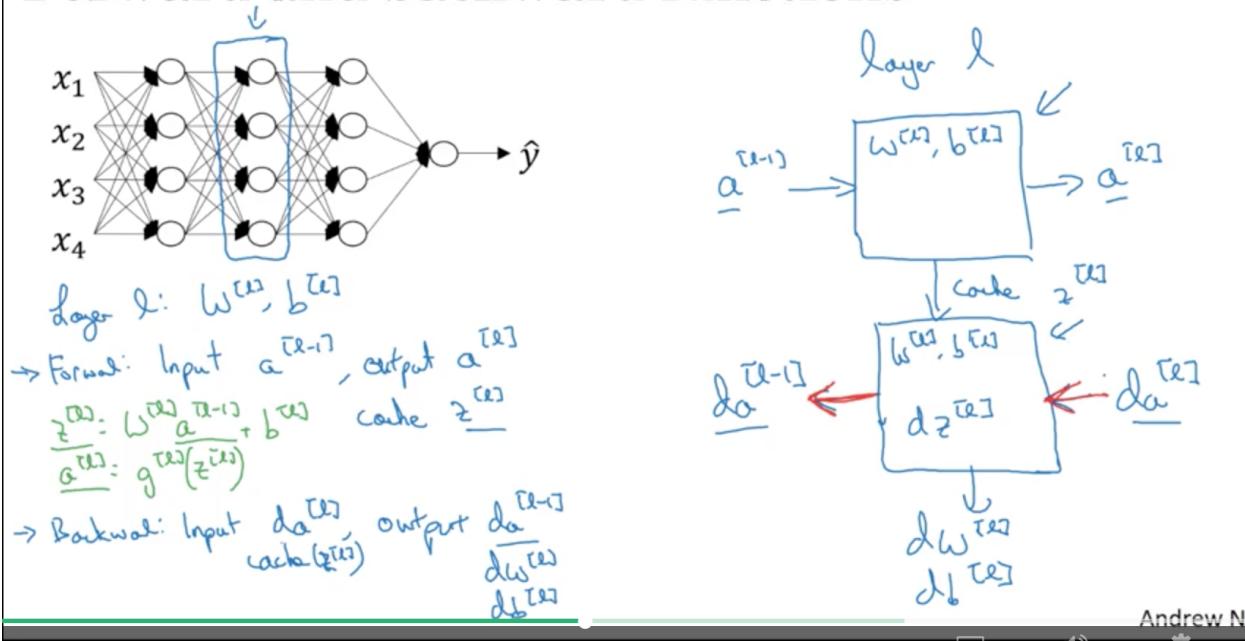
08:46 / 10:33

Andrew Ng

building blocks of deep neural networks

forward时，需要cache $Z^{[l]}$ 以供backward使用。

Forward and backward functions

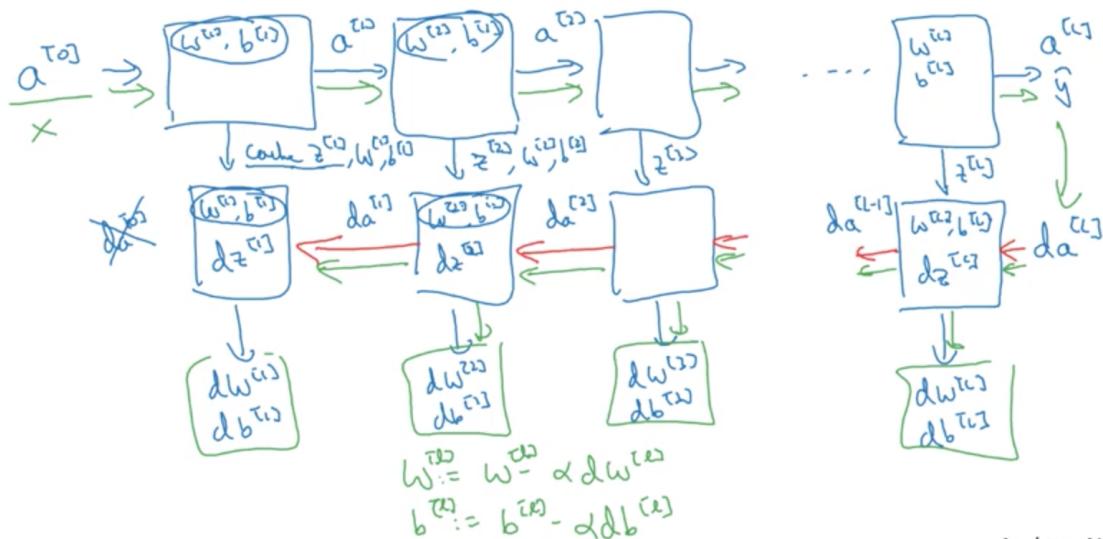


3:58 / 8:33

Andrew Ng

为了计算backward，其实需要cache的有 $Z^{[l]}$ 、 $W^{[l]}$ 以及 $b^{[l]}$ ：

Forward and backward functions



Andrew Ng

8:14 / 8.33



forward and backward propagation

forward propagation for layer l :

Forward propagation for layer l

\rightarrow Input $a^{[l-1]}$

\rightarrow Output $a^{[l]}$, cache ($z^{[l]}$)

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

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

$$\begin{matrix} a^{[0]} \\ A^{[0]} \end{matrix}$$

$$x = A^{[0]} \rightarrow \square \rightarrow \square \rightarrow \square \rightarrow$$

Vertwijf:

$$z^{[l]} = W^{[l]} \cdot A^{[l-1]} + b^{[l]}$$

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

backward propagation for layer l :

参考C1W2的backward propagation intuition部分：注意： $da^{[l]} * g^{[l]}'(z^{[l]})$ 是element-wise product。

01:58 / 10:29



Andrew Ng

Backward propagation for layer l

→ Input $da^{[l]}$

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

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

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

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

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

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

$$dz^{[l]} = dA^{[l]} \times g^{[l]}(z^{[l]})$$

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

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

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

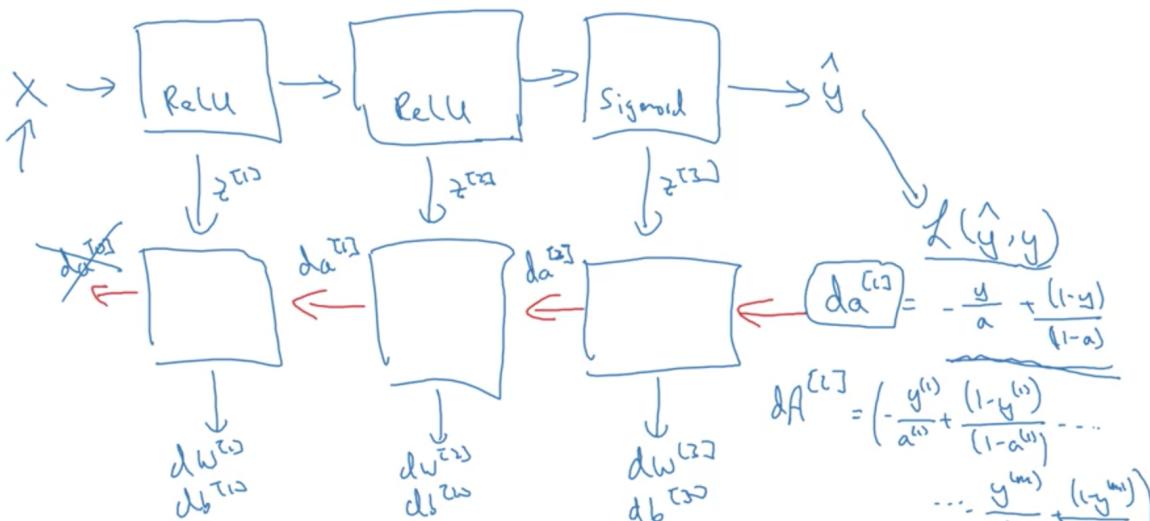
05:18 / 10:29



Andrew Ng

对于最后一层L，如果是sigmoid并采用logloss，那么：

Summary



08:12 / 10:29



Andrew Ng

parameters vs hyperparameters

What are hyperparameters?

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

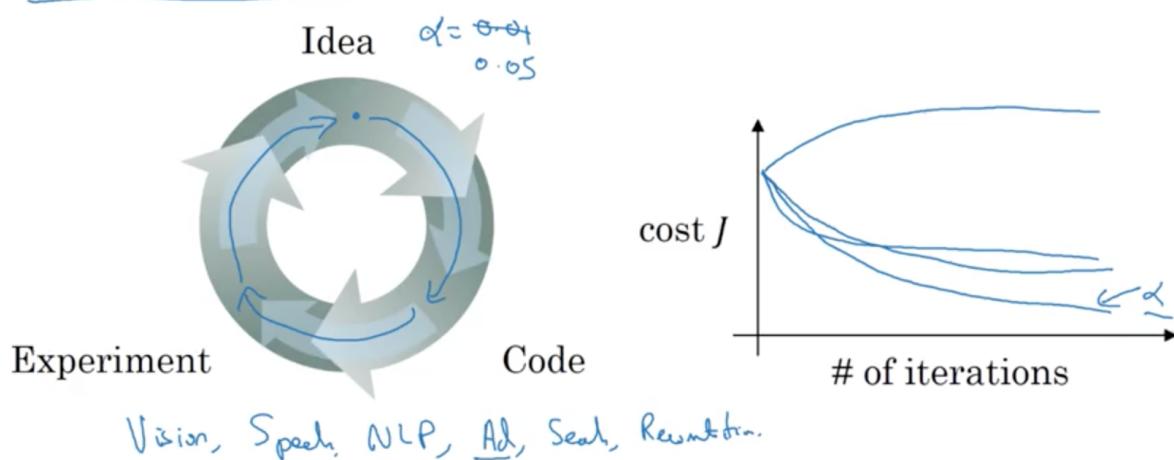
Hyperparameters:

- learning rate $\frac{\alpha}{\pi}$
- #iterations
- #hidden layers L
- #hidden units $n^{[1]}, n^{[2]}, \dots$
- choice of activation function

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

Andrew Ng

Applied deep learning is a very empirical process



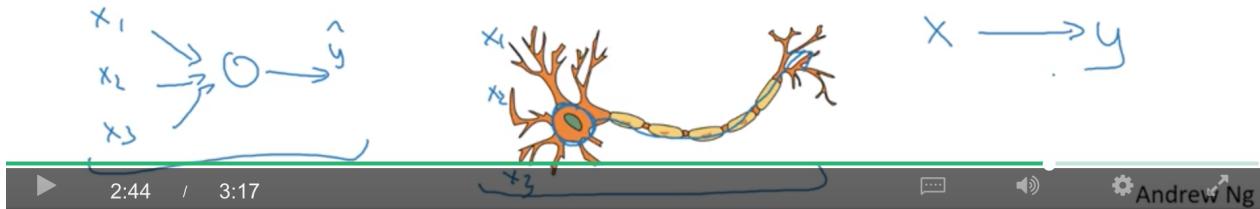
what does this have to do with the brain?

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"

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



others

Thus for example if the size of our input X is (12288, 209) (with $m = 209$ examples) then:

	Shape of W	Shape of b	Activation	Shape of Activation
Layer 1	$(n^{[1]}, 12288)$	$(n^{[1]}, 1)$	$Z^{[1]} = W^{[1]}X + b^{[1]}$	$(n^{[1]}, 209)$
Layer 2	$(n^{[2]}, n^{[1]})$	$(n^{[2]}, 1)$	$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$	$(n^{[2]}, 209)$
\vdots	\vdots	\vdots	\vdots	\vdots
Layer L-1	$(n^{[L-1]}, n^{[L-2]})$	$(n^{[L-1]}, 1)$	$Z^{[L-1]} = W^{[L-1]}A^{[L-2]} + b^{[L-1]}$	$(n^{[L-1]}, 209)$
Layer L	$(n^{[L]}, n^{[L-1]})$	$(n^{[L]}, 1)$	$Z^{[L]} = W^{[L]}A^{[L-1]} + b^{[L]}$	$(n^{[L]}, 209)$