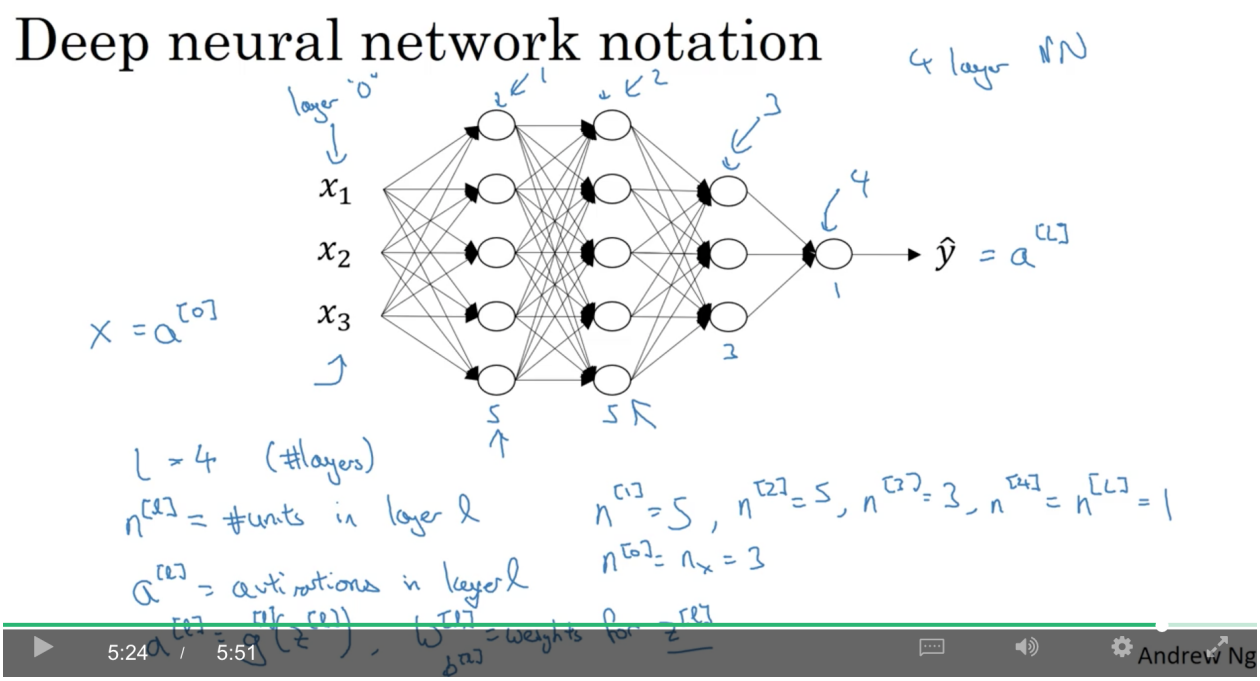
contents

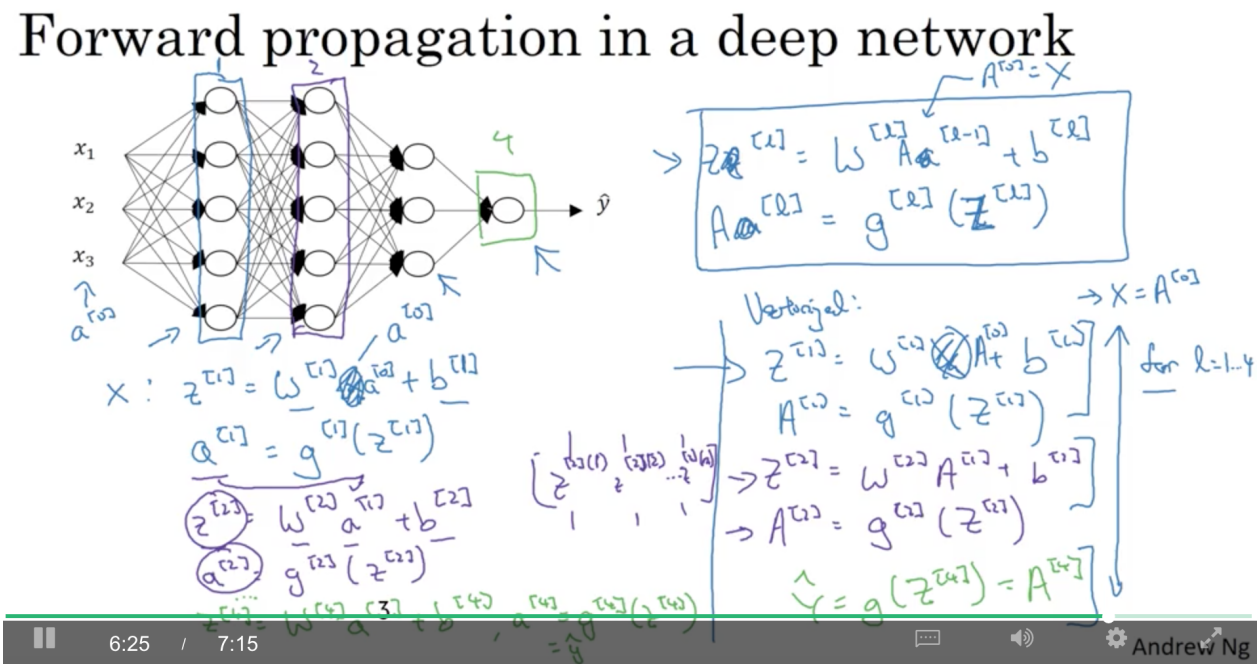
deep L-layer neural network



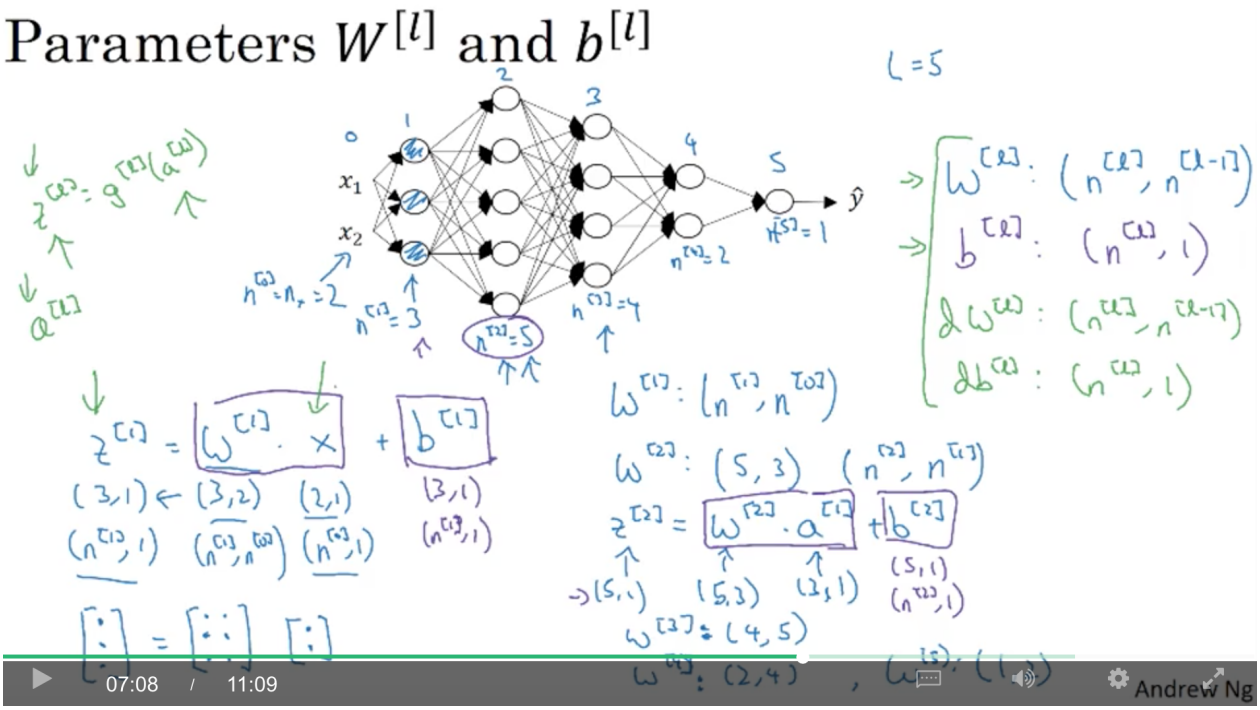
forward propagation in a deep network

$z^{[l](i)}$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->m的这个for-loop。

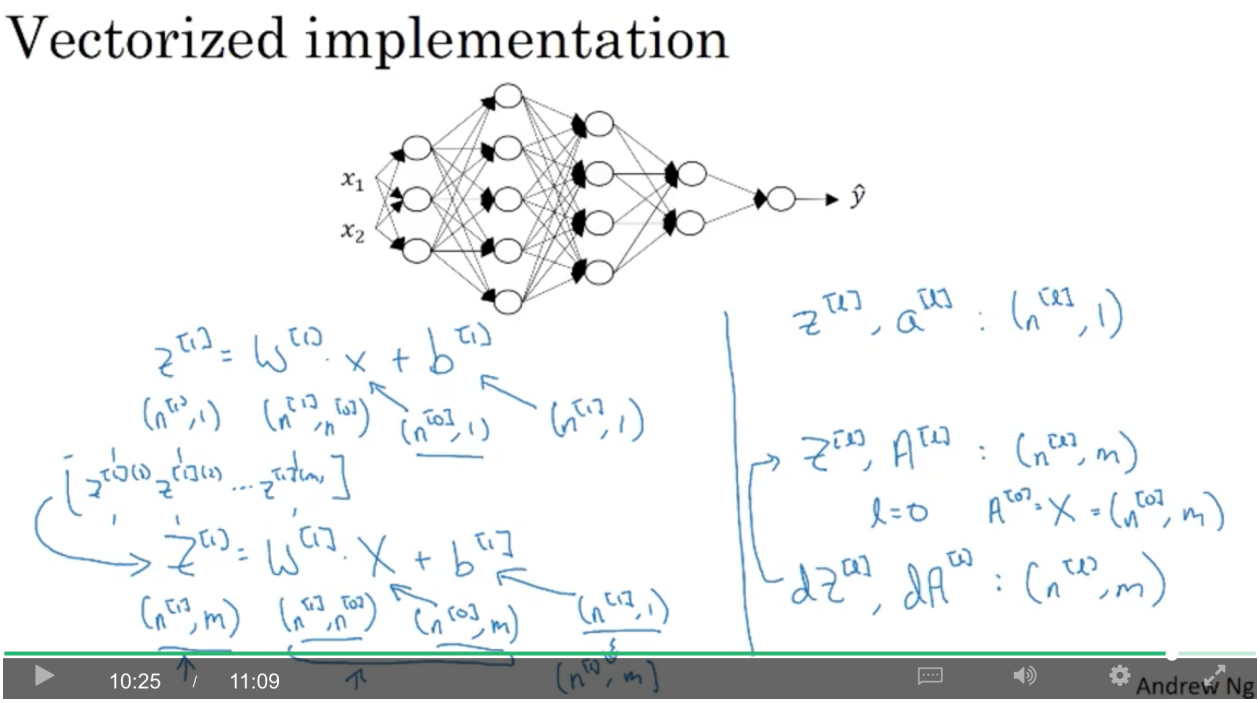
另外，从第1层到第4层这个for-loop是无法避免的。



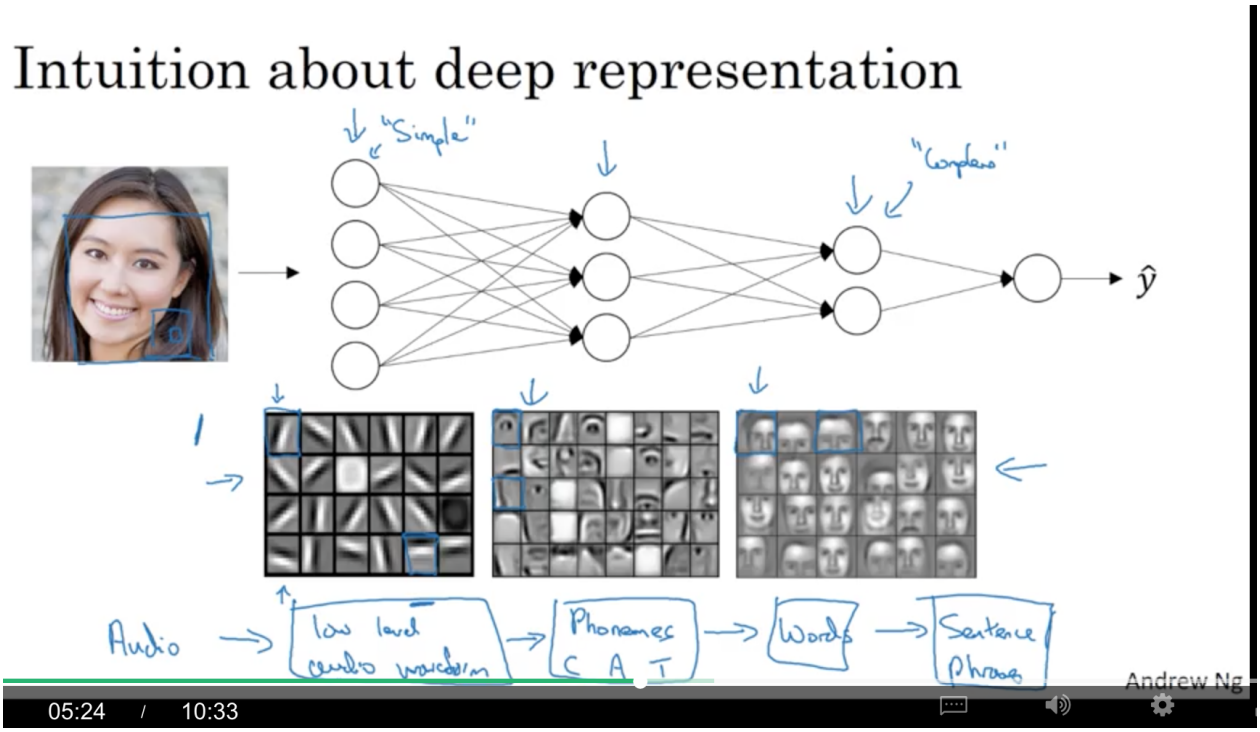
getting your matrix dimensions right



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。

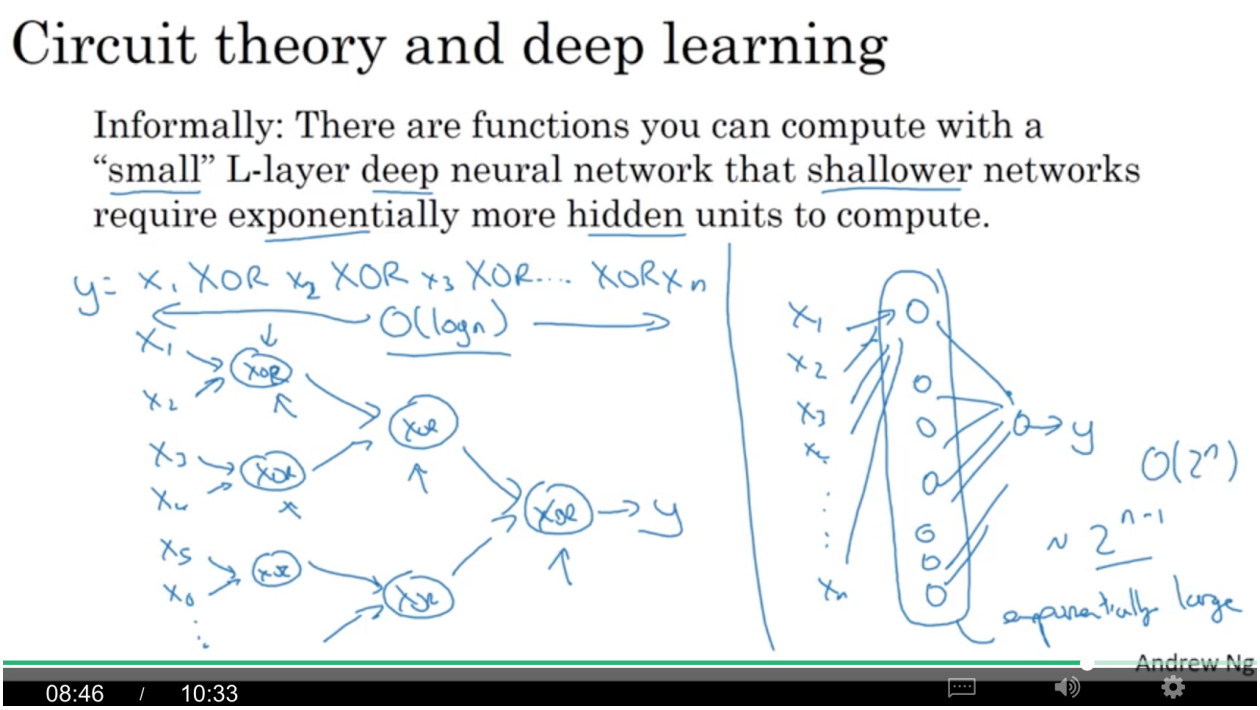


why deep representations?



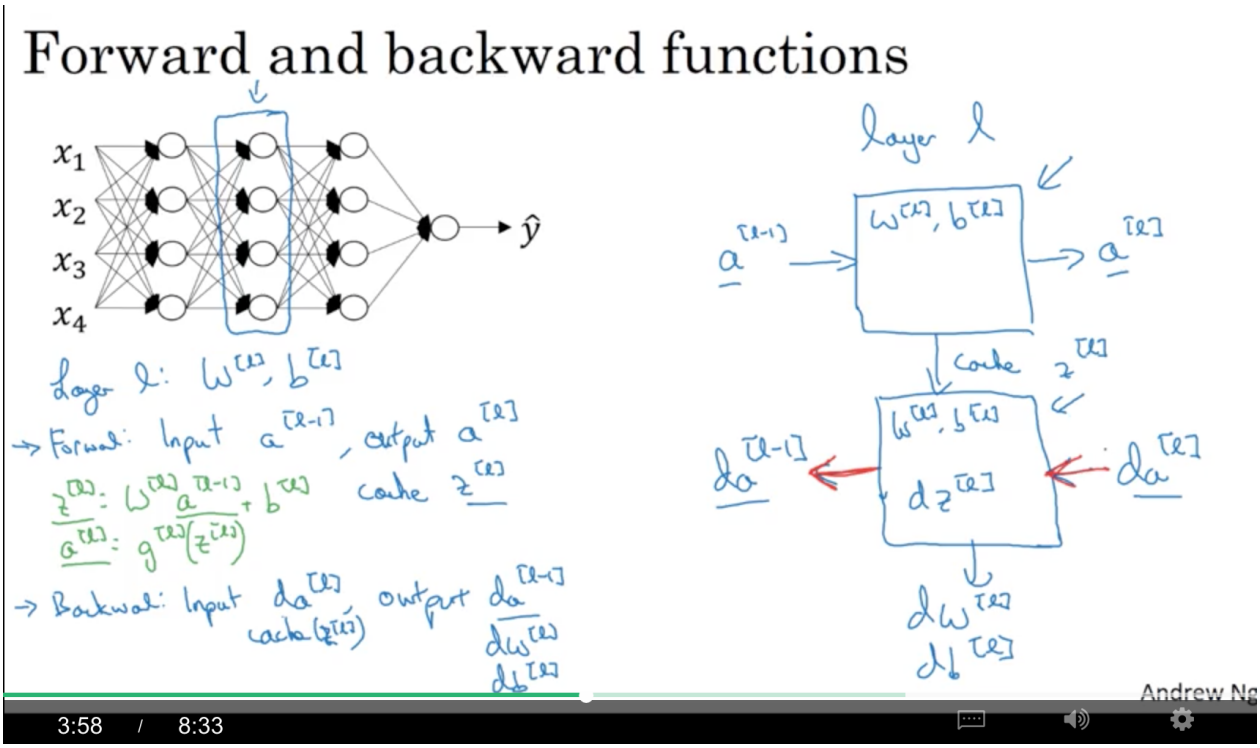
对于$(x_1, x_2, \dots x_n)$的异或（XOR）操作，如果用树型结构，n个叶子节点，则树深度是$log_2 n + 1$(深度为k的满二叉树的第i层上有2^{i-1}个结点,总共至多有$2^k - 1$个结点)，即只需要$O(log_2 n)$层的树就能完成。

而如果采用单隐层，则需要$2^{(n-1)}$个节点

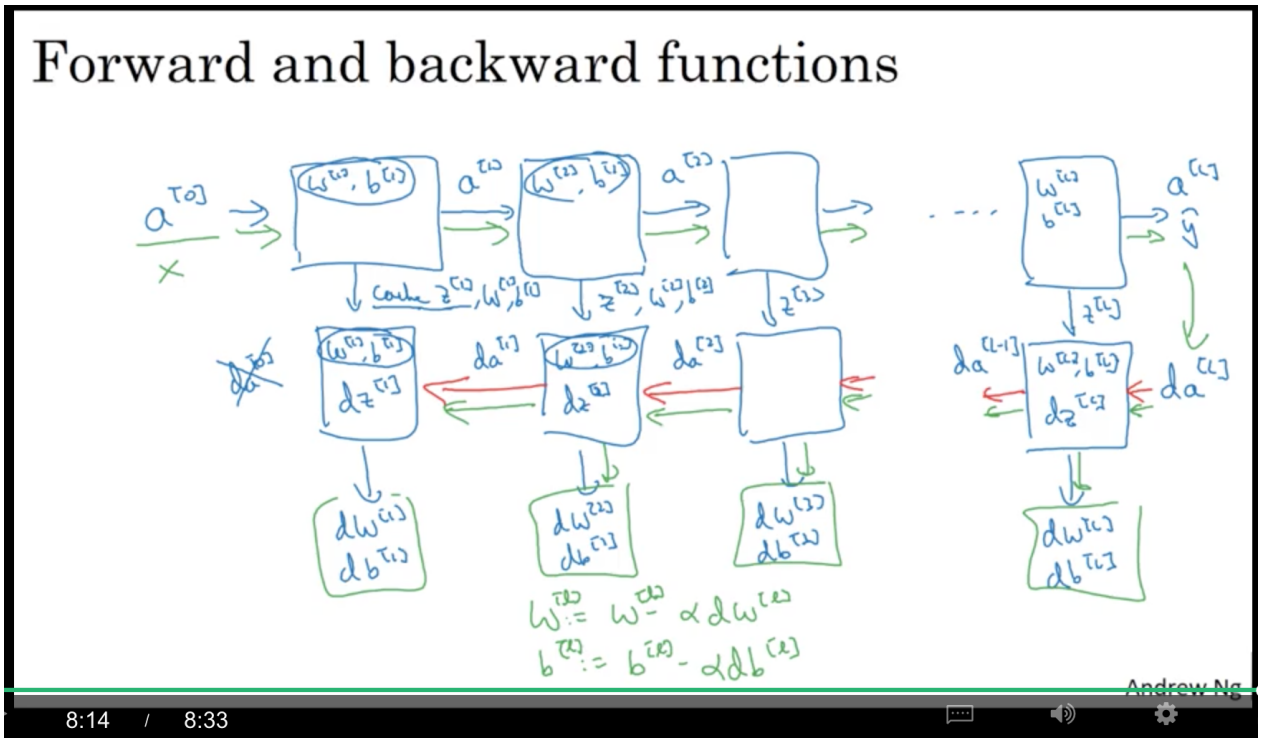


building blocks of deep neural networks

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

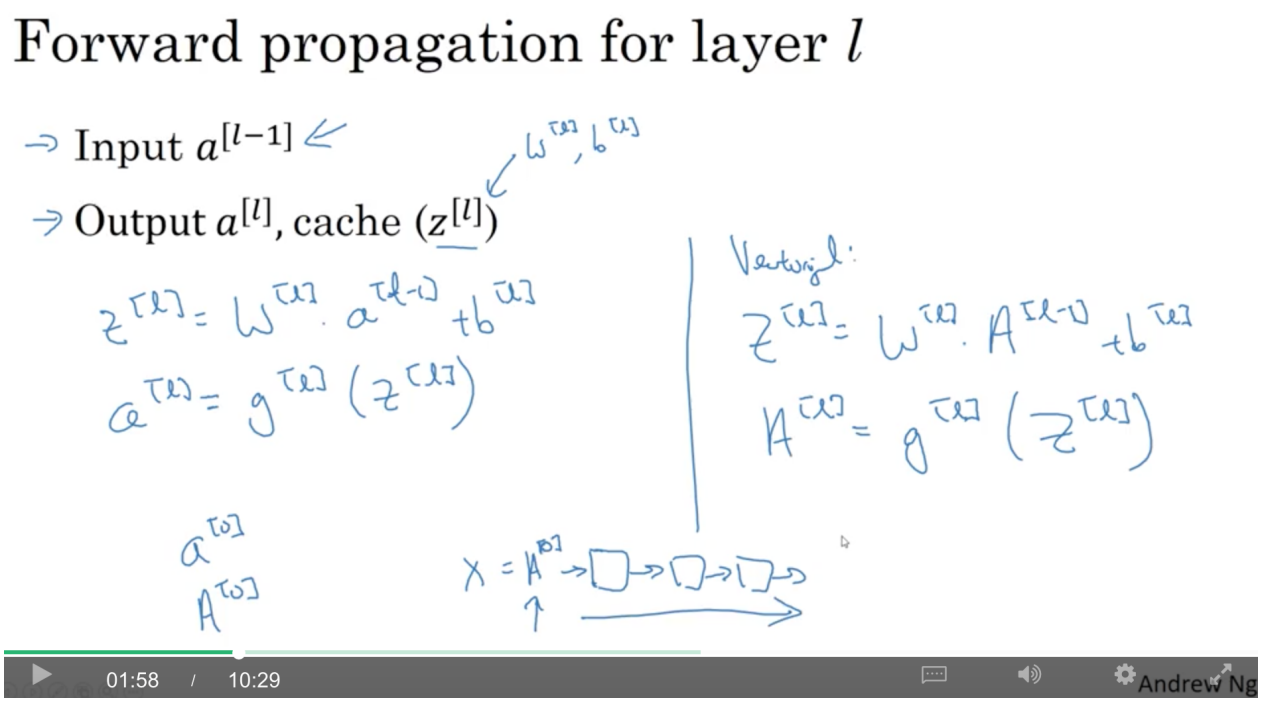


为了计算backward，其实需要cache的有$Z^{[l]}$ 、$W^{[l]}$ 以及$b^{[l]}$:



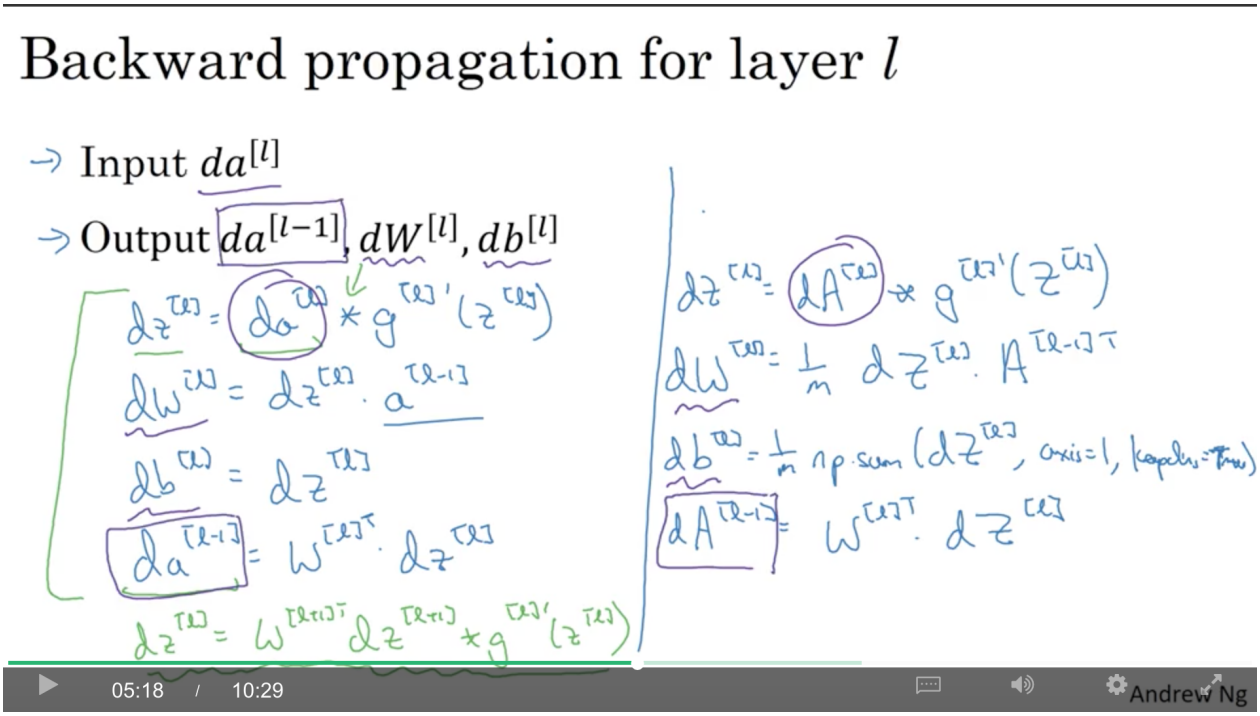
forward and backward propagation

forward propagation for layer l:

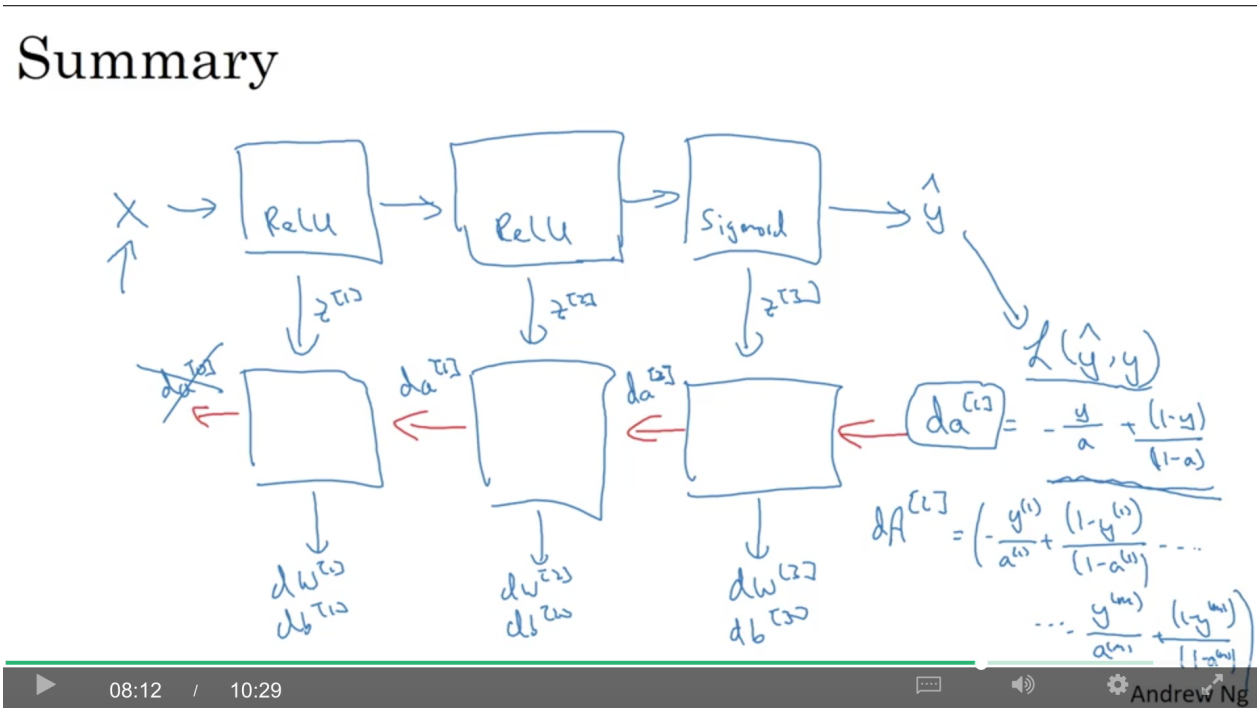


backward propagation for layer l:

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

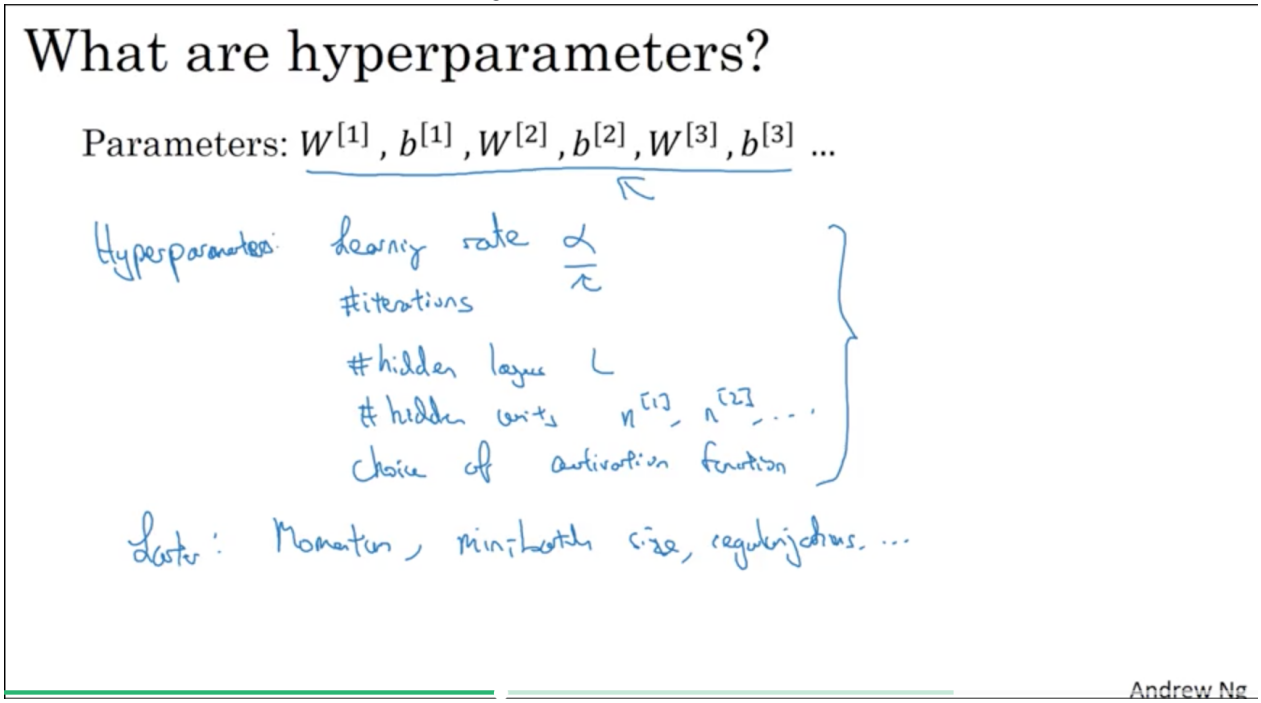


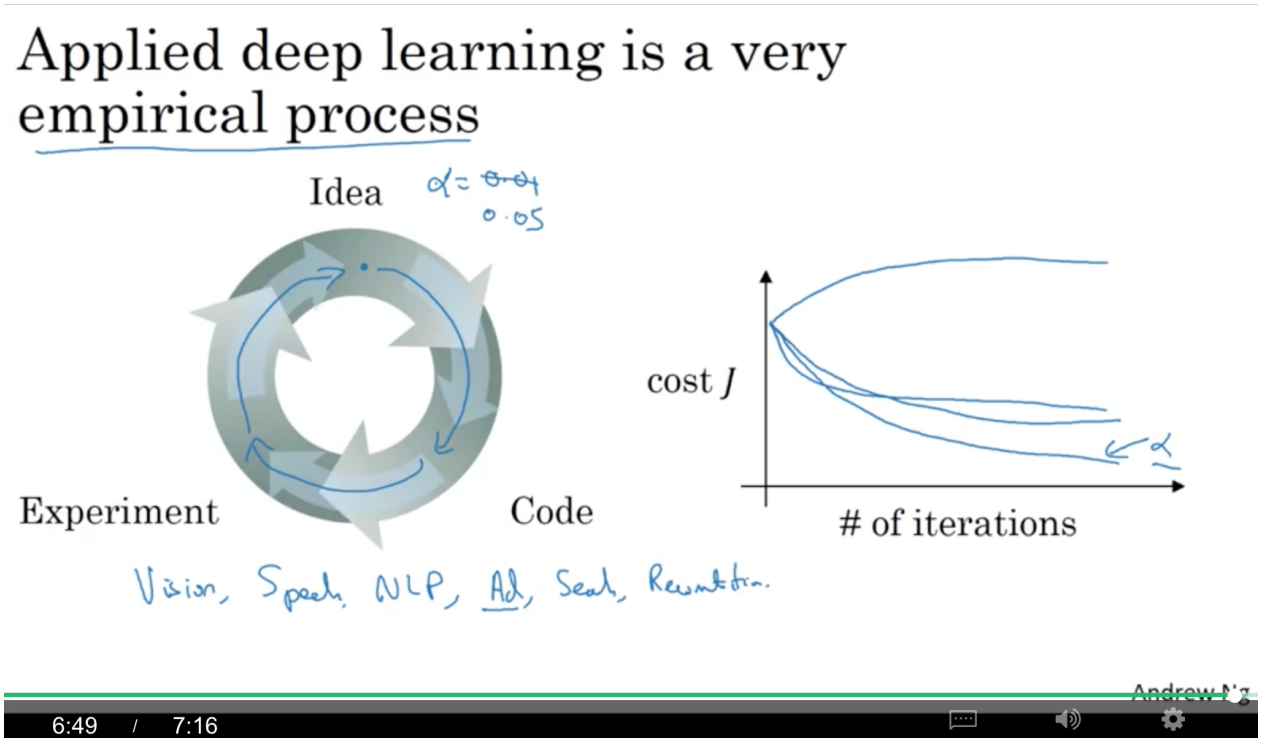
对于最后一层L，如果是sigmoid并采用logloss，那么：



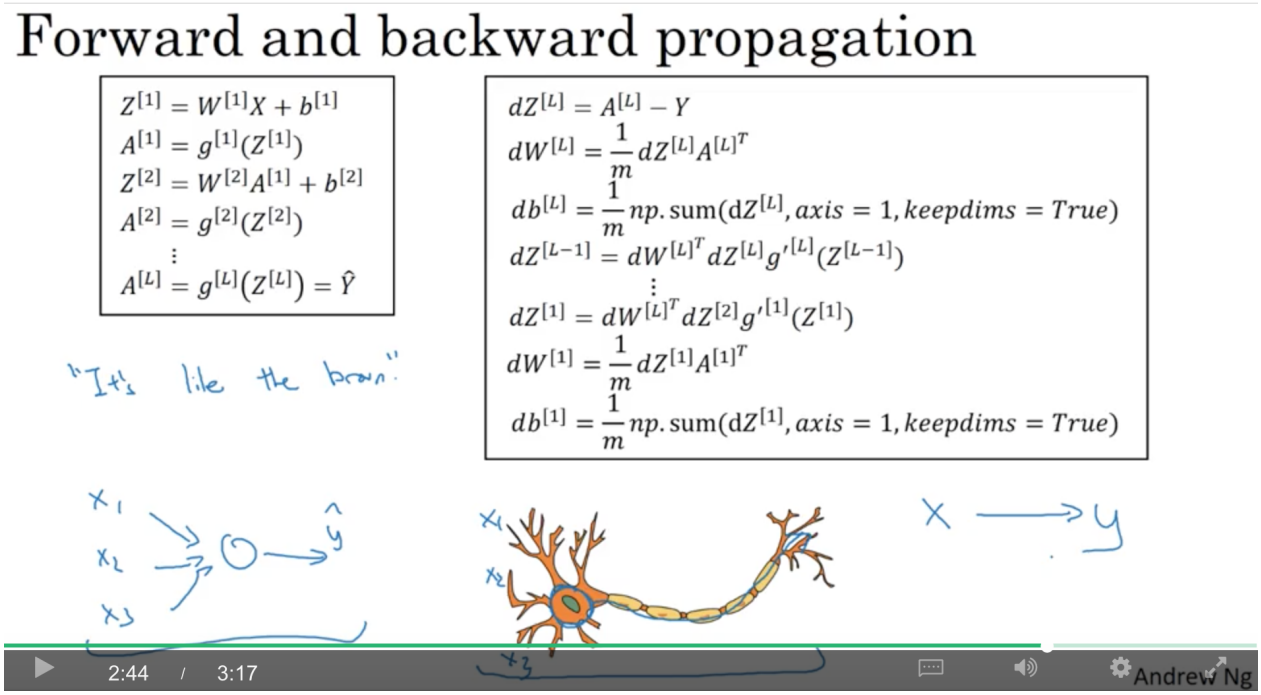
parameters vs hyperparameters

图中下方是：momentum，mini-batch size，regularization





what does this have to do with the brain?



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	$(n^{[L]}, n^{[L-1]})$	$(n^{[L]}, 1)$	$Z^{[L]} = W^{[L]}A^{[L-1]} + b^{[L]}$	$(n^{[L]}, 209)$