

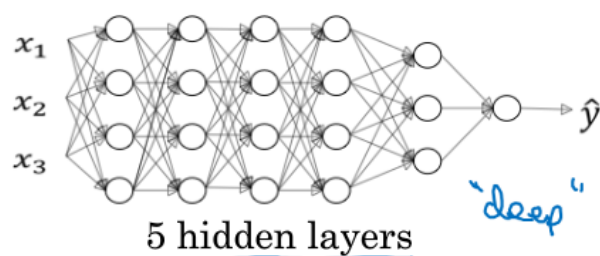
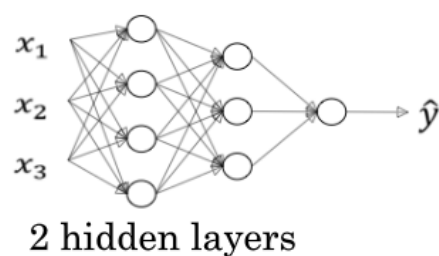
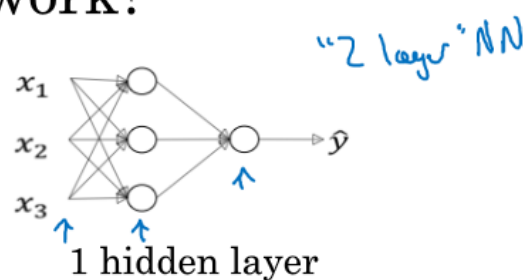
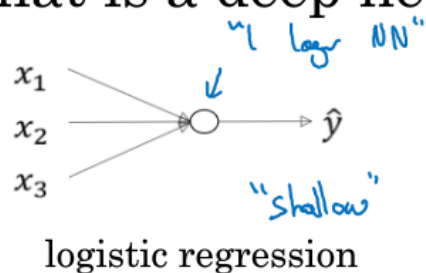
# 神经网络和深度学习 第四周 深层神经网络

## 4.1 深层神经网络

什么是深层神经网络？

深层神经网络可以理解为隐藏层比较多的神经网络。

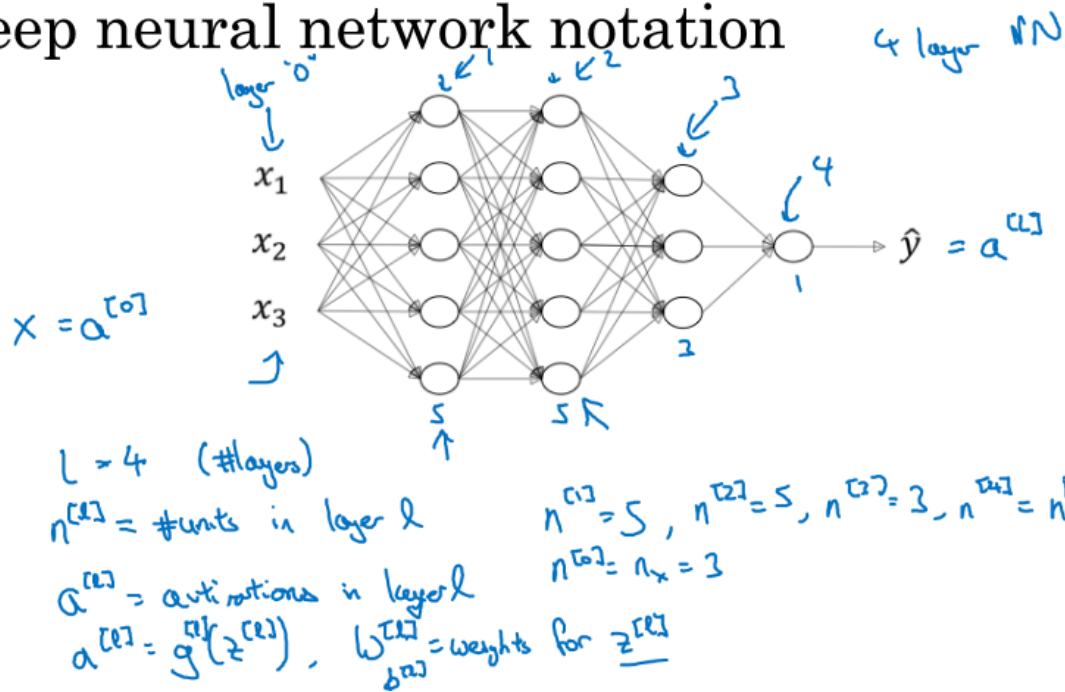
### What is a deep neural network?



Andrew

深层神经网络的每个元素的标记

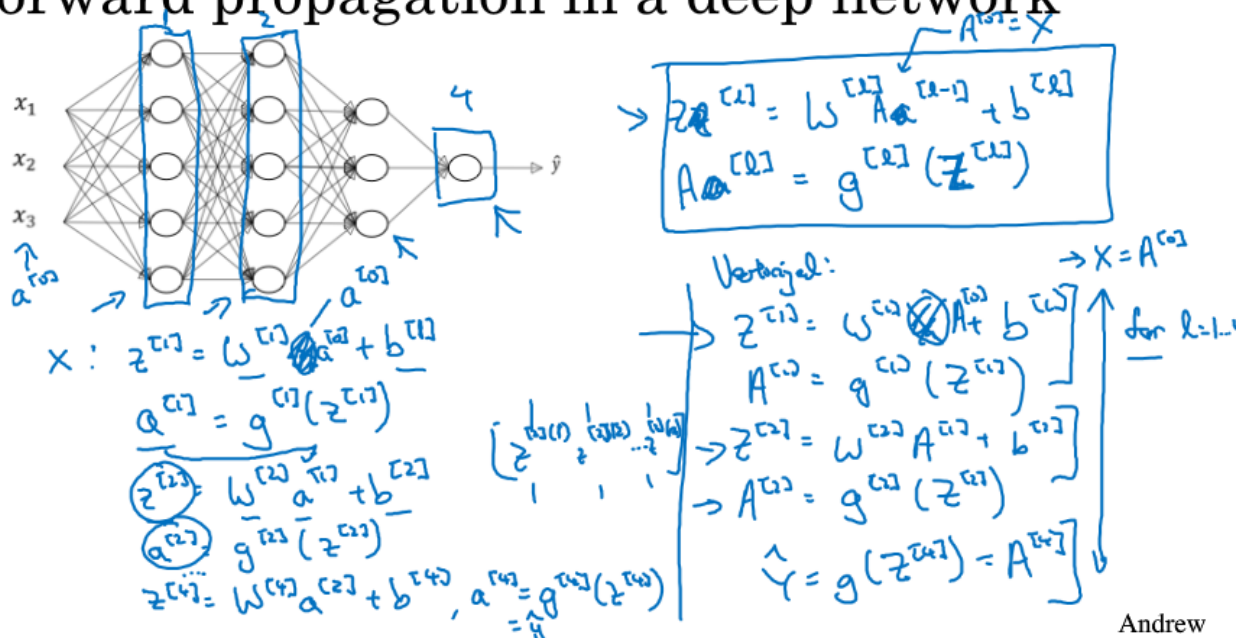
# Deep neural network notation



Andrew

## 4.2 深层网络中的前向传播算法

### Forward propagation in a deep network

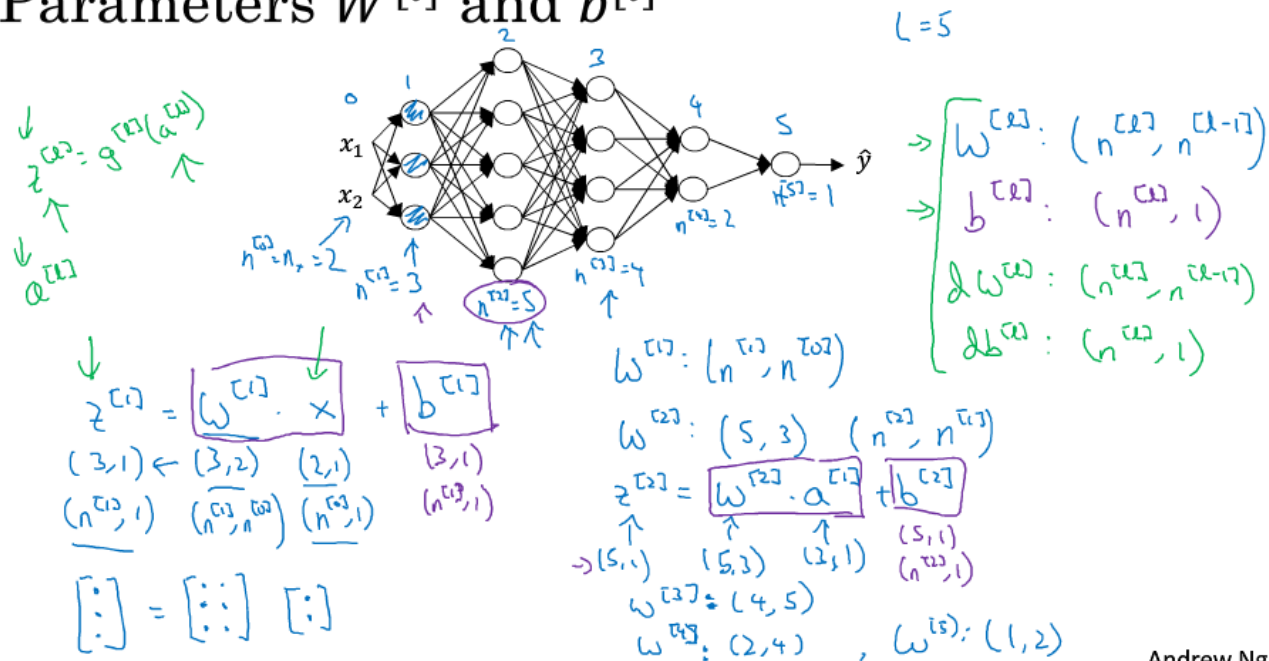


Andrew

根据图中神经网络，直观的理解为从左往右传播。

## 4.3 核对矩阵的维度

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



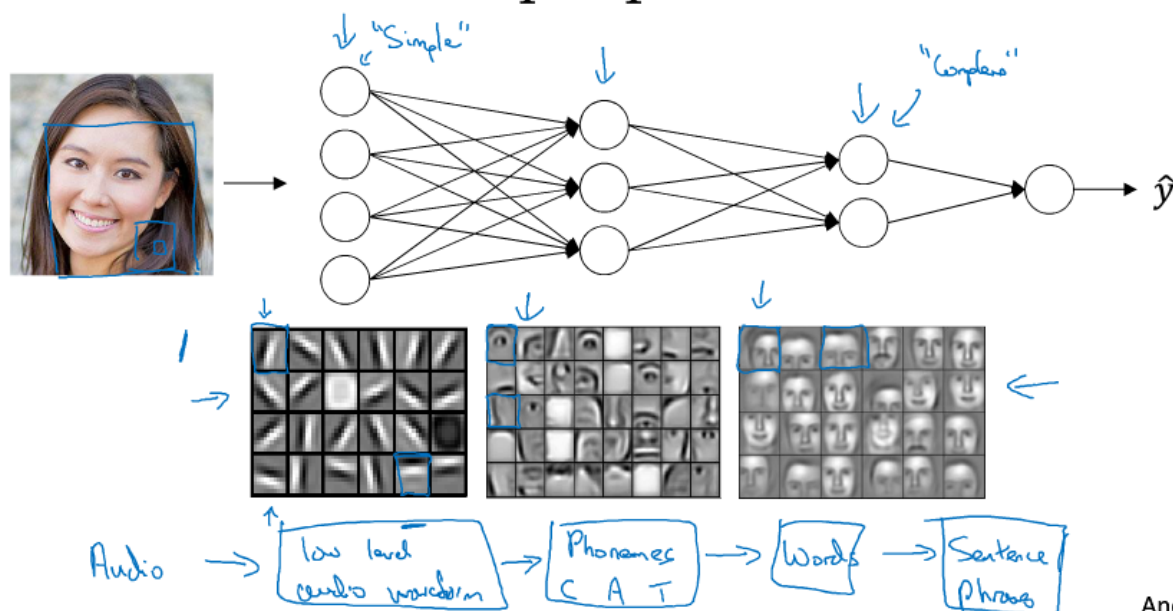
Andrew Ng

对于第 $l$ 层神经网络，单个样本其各个参数的矩阵维度为：

- $W^{[l]}: (n^{[l]}, n^{[l-1]})$
- $b^{[l]}: (n^{[l]}, 1)$
- $dW^{[l]}: (n^{[l]}, n^{[l-1]})$
- $db^{[l]}: (n^{[l]}, 1)$
- $Z^{[l]}: (n^{[l]}, 1)$
- $A^{[l]} = Z^{[l]}: (n^{[l]}, 1)$

## 4.4 为什么使用深层表示

# Intuition about deep representation



Andrew Ng

对于人脸识别，神经网络的第一层从原始图片中提取人脸的轮廓和边缘，每个神经元学习到不同边缘的信息；网络的第二层将第一层学得的边缘信息组合起来，形成人脸的一些局部的特征，例如眼睛、嘴巴等；后面的几层逐步将上一层的特征组合起来，形成人脸的模样。随着神经网络层数的增加，特征也从原来的边缘逐步扩展为人脸的整体，由整体到局部，由简单到复杂。层数越多，那么模型学习的效果也就越精确。

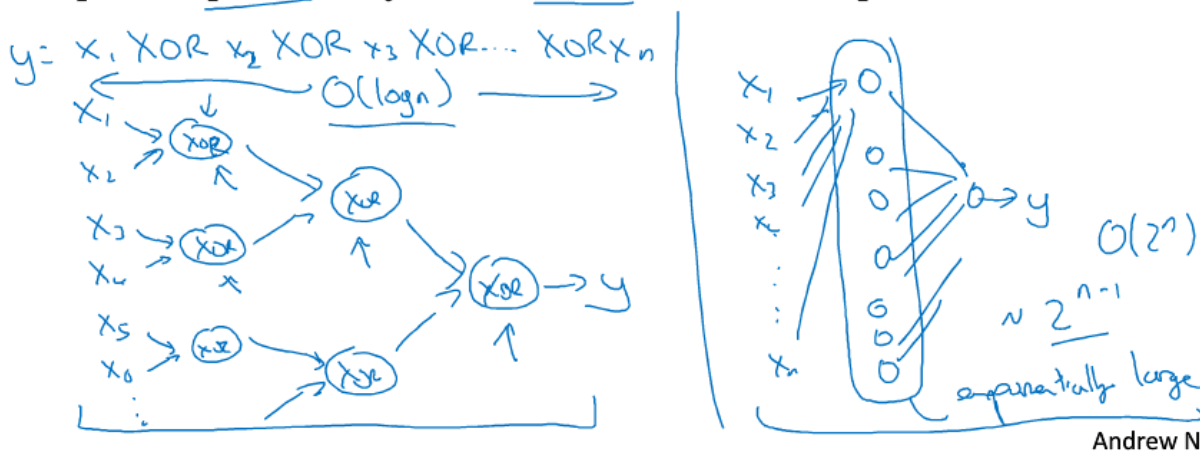
对于语音识别，第一层神经网络可以学习到语言发音的一些音调，后面更深层次的网络可以检测到基本的音素，再到单词信息，逐渐加深可以学到短语、句子。

所以从上面的两个例子可以看出随着神经网络的深度加深，模型能学习到更加复杂的问题，功能也更加强大。

电路逻辑计算：

## 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.



Andrew Ng

假定计算异或逻辑输出：

$$y = x_1 \oplus x_2 \oplus x_3 \oplus \dots \oplus x_n$$

对于该运算，若果使用深度神经网络，每层将前一层的相邻的两单元进行异或，最后到一个输出，此时整个网络的层数为一个树形的形状，网络的深度为 $O(\log_2(n))$ ，共使用的神经元的个数为：

$$1 + 2 + \dots + 2^{\log_2(n)-1} = 1 \cdot \frac{1 - 2^{\log_2(n)}}{1 - 2} = 2^{\log_2(n)} - 1 = n - 1$$

即输入个数为 $n$ ，输出个数为 $n-1$ 。

但是如果不适用深层网络，仅仅使用单隐层的网络（如右图所示），需要的神经元个数为 $2n-1$ 个。同样的问题，但是深层网络要比浅层网络所需要的神经元个数要少得多。

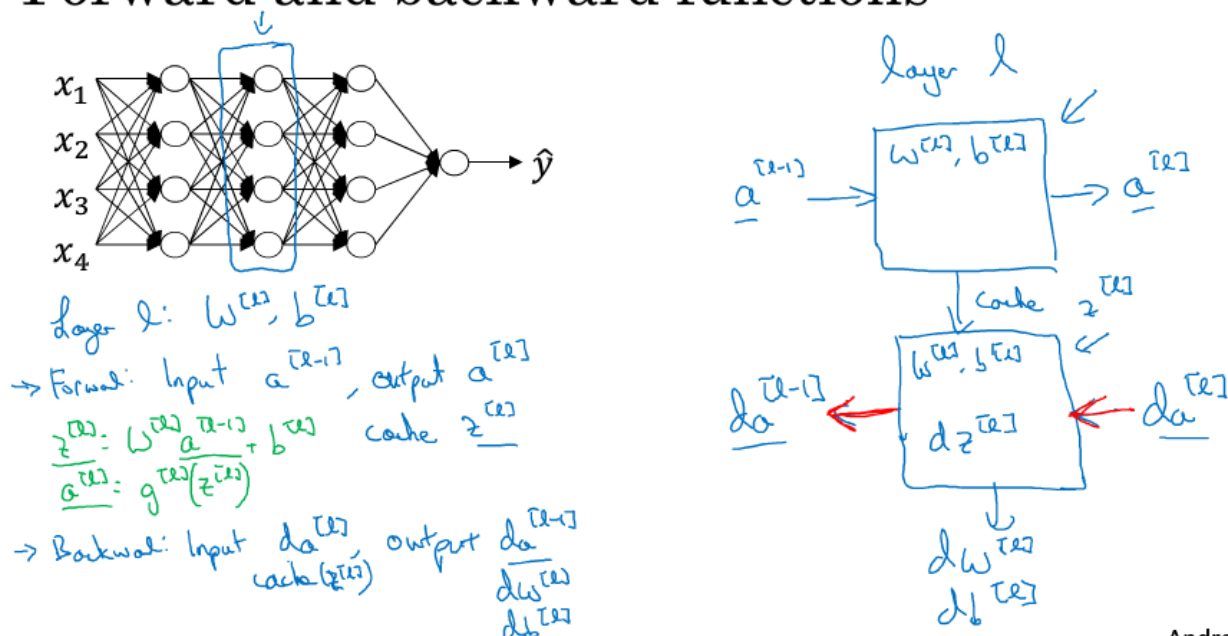
## 4.5 搭建深层神经网络块

首先给定DNN的一些参数：

- $L$ ：DNN的总层数；
- $n^{[l]}$ ：表示第 $l$ 层的包含的单元个数；
- $a^{[l]}$ ：表示第 $l$ 层激活函数的输出；
- $W^{[l]}$ ：表示第 $l$ 层的权重；
- 输入 $x$ 记为 $a^{[0]}$ ，输出 $\hat{y}$ 记为 $a^{[L]}$ 。

前向传播和反向传播函数

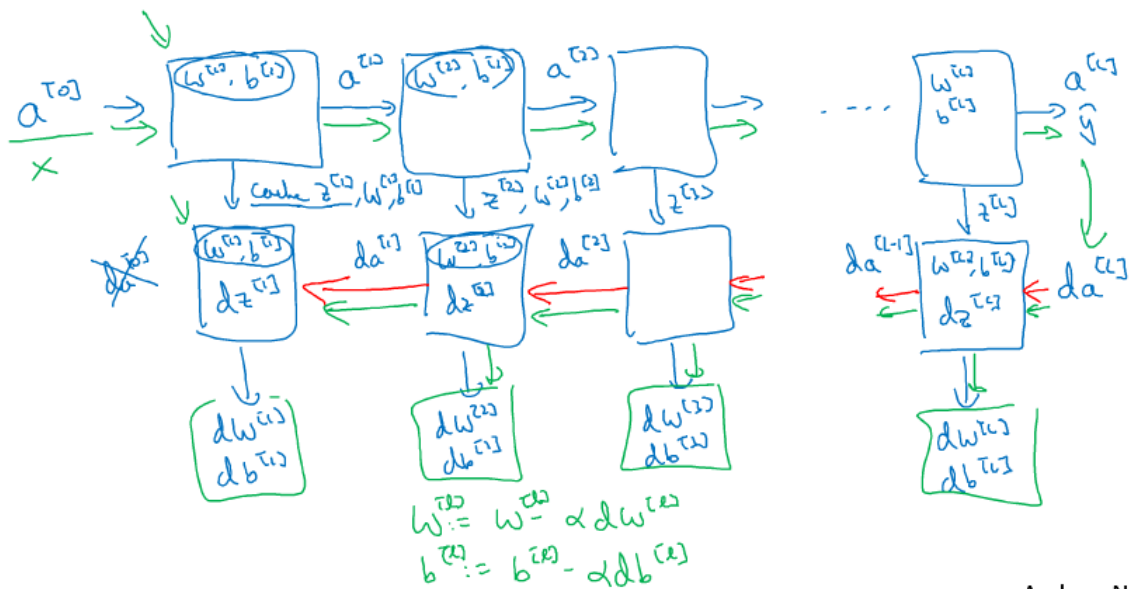
### Forward and backward functions



Andrew Ng

前向传播和反向传播函数实现的细节

# Forward and backward functions



Andrew Ng

第l层的前向传播

## Forward propagation for layer $l$

→ Input  $a^{[l-1]}$

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

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

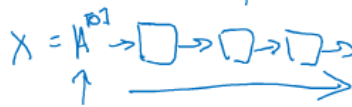
$$a^{[l]} = g(z^{[l]})$$

Vectorized:

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

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

$a^{[l-1]}$   
 $A^{[l-1]}$



Andrew Ng

根据图，知道输入，输出。

前向传播的公式如如钟左下所示，向量化程序如图中的右下所示。

第l层的反向传播

# Backward propagation for layer $l$

→ Input  $da^{[l]}$

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

$$\begin{aligned} dz^{[l]} &= da^{[l]} * g^{[l]'}(z^{[l]}) \\ dW^{[l]} &= dz^{[l]} \cdot a^{[l-1]} \\ db^{[l]} &= dz^{[l]} \\ da^{[l-1]} &= W^{[l]T} \cdot dz^{[l]} \\ dz^{[l]} &= W^{[l+1]T} dz^{[l+1]} * g^{[l+1]'}(z^{[l+1]}) \end{aligned}$$

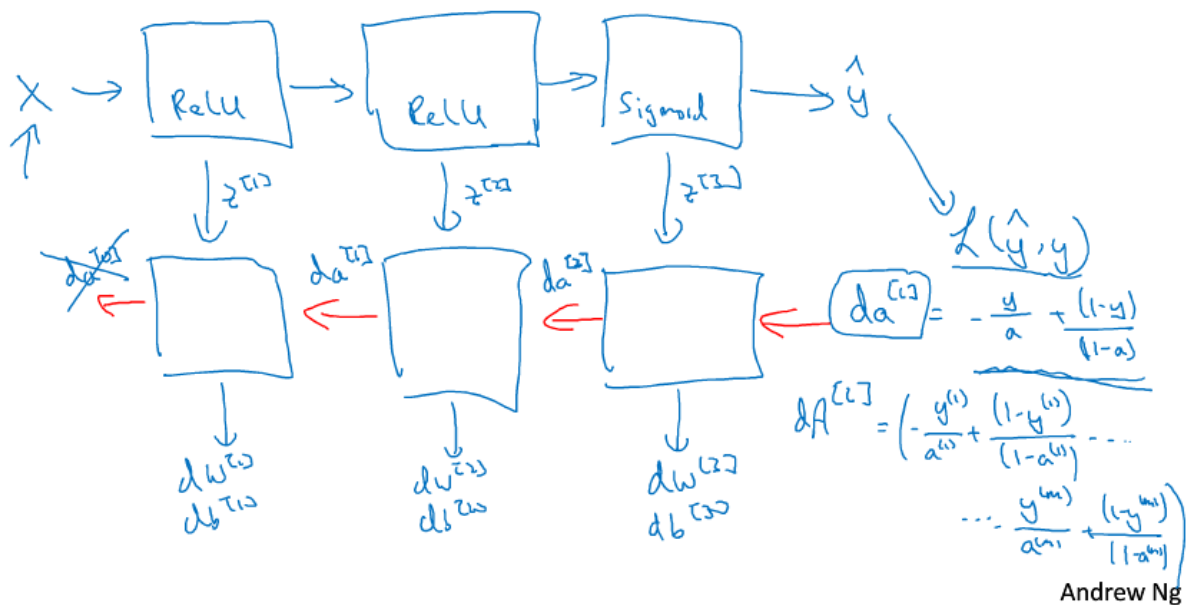
$$\begin{aligned} dz^{[l]} &= dA^{[l]} * g^{[l]'}(z^{[l]}) \\ dW^{[l]} &= \frac{1}{n} dz^{[l]} \cdot A^{[l-1]T} \\ db^{[l]} &= \frac{1}{n} np.sum(dz^{[l]}, axis=1, keepdims=True) \\ dA^{[l-1]} &= W^{[l]T} \cdot dz^{[l]} \end{aligned}$$

根据图，知道输入，输出。

反向传播的公式如如钟左下所示，向量化程序如图中的右下所示。

使用不同激活函数执行的概要如下图

## Summary



## 4.7 参数和超参数

什么是超参数？



# What are hyperparameters?

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

Hyperparameters:  $\alpha$   
#iterations  
#hidden layers  $L$   
#hidden units  $n^{[1]}, n^{[2]}, \dots$   
choice of activation function

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

因为

**参数**：即是我们过程中想要模型学习到的信息， $W[l], b[l], W[l], b[l]$ 。

**超参数**：超参数即为控制参数的输出值的一些网络信息，超参数在某种程度上，决定了最终得到的 $w$ 和 $b$ 。

实际上，深度学习有很多其他的超参数。

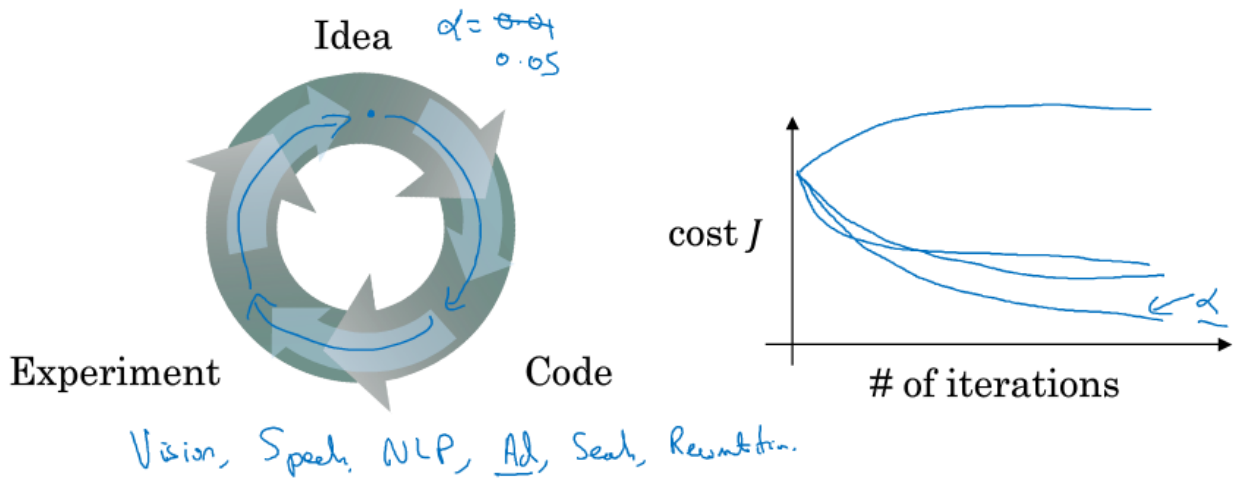
- 学习速率： $\alpha$
- 迭代次数： $N$
- 隐藏层的层数： $L$
- 每一层的神经元个数： $n^{[1]}, n^{[2]}, \dots$
- 激活函数 $g(z)$ 的选择

比如，momentum，mini batch, 几种不同的正则化参数。

在开发一个应用时，事先不知道超参数什么是合适的。就要多次尝试。



# Applied deep learning is a very empirical process



现在深度学习用于解决很多问题。从计算机视觉到语音识别，到自然语言处理，到很多结构化的数据应用。比如网络广告。或是网页搜索，或产品推荐等等。

有时候，这种设置超参数的直觉，可以推广。有时，又不能推广。所以我经常建议人们，特别是刚开始应用于新问题的人们，尝试各种超参数的取值。尽管每天你在调试最优参数，但是过一年，你的电脑的设备会变化。也有可能由于CPU、GPU和网络的原因。慢慢你会得到设置超参数的直觉。