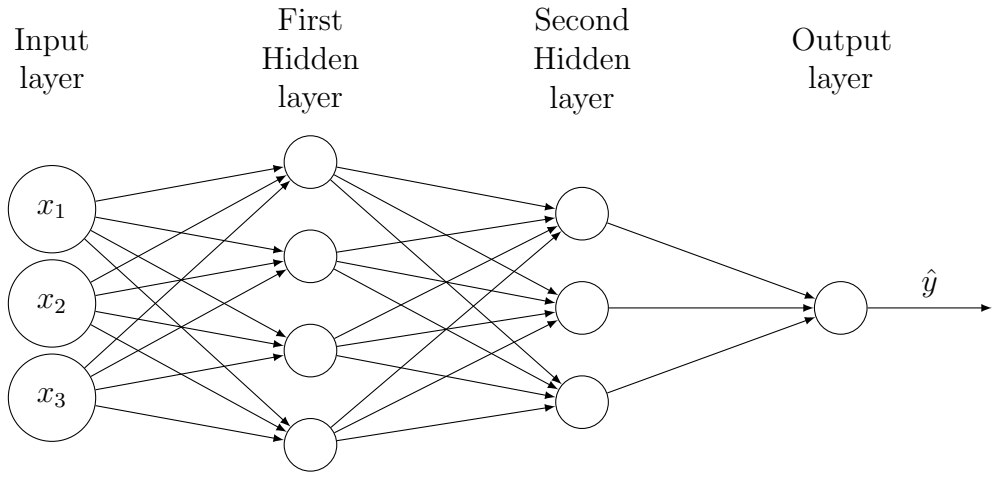
Deep L-layer neural network

In the former sessions we studied very shallow neural networks as the logistic regression (one layer neural network) and a 2 layer Neural Network. In this session we will study the more general case with L layers neural networks. A three layer Neural Network is shown below.



Notation:

L	The number of layers in the neural network
$n^{[l]}$	The number of units in the layer l
$W^{[l]}$	The weights matrix in the layer l
$b^{[l]}$	The bias vector in the layer l
$a^{[l]}$	Activation vector in the layer l
$a^{[0]} := x$	
$a^{[l]} := \hat{y}$	

Forward Propagation in a Deep Network

For a the input $X = \left[x^{(1)} \mid \cdots \mid x^{(m)}\right]$ the forward propagation is given by the following equations:

$$Z^{[i]} = W^{[i]} A^{[i-1]} + b^{[i]} \qquad \forall i \in \{1, \ldots, l\}$$
$$A^{[i]} = g^{[i]}(Z^{[i]}) \qquad \forall i \in \{1, \ldots, l\}$$

It's extremely important (and sometimes really difficult) to get the right matrix dimensions

$$W^{[i]} \in \mathbb{R}^{n^{[i]} \times n^{[i-1]}} \qquad A^{[i]} \in \mathbb{R}^{n^{[i-1]} \times m}$$
$$b^{[i]} \in \mathbb{R}^{n^{[i]} \times m} \qquad Z^{[i]} \in \mathbb{R}^{n^{[i]} \times m}$$

Deep representations

Why is it important for the neural networks to be *deep* and have a lot of hidden layers?

Intuitively, you can think of the earlier layers of the neural network as detecting simple functions. And then composing them together in the later layers of a neural network so that it can learn more and more complex functions.

This simple to complex hierarchical representation, or compositional representation, applies in several types of data loke images (for problems offace recognition) and audio (for problems of speech recognition)

The advantage of multiple layers is that they can learn features at various levels of abstraction. For example, if you train a deep convolutional neural network to classify images, you will find that the first layer will train itself to recognize very basic things like edges, the next layer will train itself to recognize collections of edges such as shapes, the next layer will train itself to recognize collections of shapes like eyes or noses, and the next layer will learn even higher-order features like faces. Multiple layers are much better at generalizing because they learn all the intermediate features between the raw data and the high-level classification.

Forward and Backward Propagation

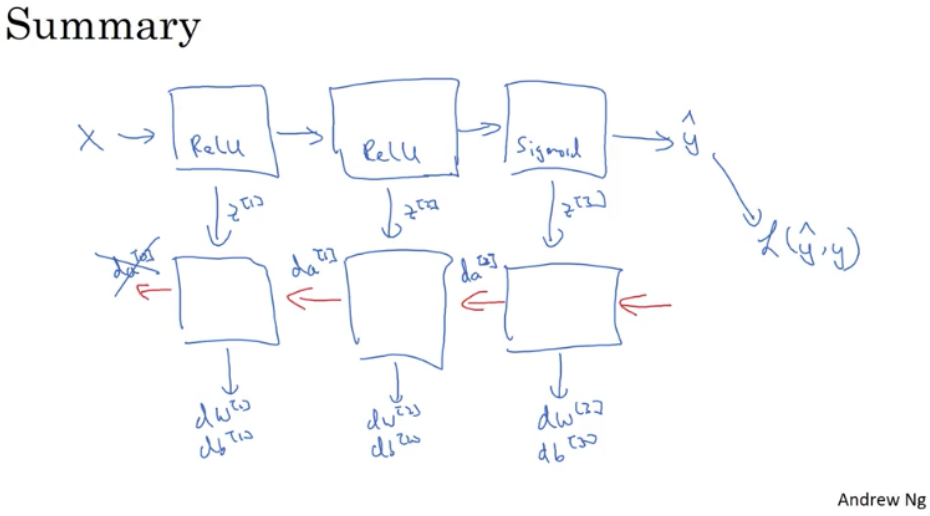
When doing backward propagation for the layer l we have

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

Where the gradients are given by:

$$
\begin{aligned}
dZ^{[l]} &= dA^{[l]} * g^{[l]'}(Z^{[l]}) \\
dW^{[l]} &= \frac{1}{m} dZ^{[l]} A^{[l-1]\top} \\
db^{[l]} &= \frac{1}{n} sum(dZ^{[l]}) \\
dA^{[l-1]} &= W^{[l]\top} dZ^{[l]}
\end{aligned}
$$

A summary of the forward and backward propagation process for a three layer neural network is ilustrated in the following image:



Parameters vs hyperparameters

Parameters:

$W^{[i]}\ \forall i \in \{1, ..., L\}$	The weights matrix in each layer
$b^{[i]}\ \forall i \in \{1, ..., L\}$	The bias vector in each layer

Hyperparameters:

α	The learning rate
n_{iter}	The number of iterations
L	The number of hidden layers
$n^{[i]}\ \forall i \in \{1, ..., L\}$	The number of hidden units in each layer
$n^{[i]}\ \forall i \in \{1, ..., L\}$	The activation function for each layer

And several other hyperparameters like momentum, minibatch size and regularization.