Putting it Together

First, pick a network architecture; choose the layout of your neural network, including how many h
how many layers in total you want to have.

- Number of input units = dimension of features $x^{(i)}$
- Number of output units = number of classes
- Number of hidden units per layer = usually more the better (must balance with cost of comp
more hidden units)
- Defaults: 1 hidden layer. If you have more than 1 hidden layer, then it is recommended that
units in every hidden layer.

Training a Neural Network

1. Randomly initialize the weights
2. Implement forward propagation to get $h_\Theta(x^{(i)})$ for any $x^{(i)}$
3. Implement the cost function
4. Implement backpropagation to compute partial derivatives
5. Use gradient checking to confirm that your backpropagation works. Then disable gradient ch
6. Use gradient descent or a built-in optimization function to minimize the cost function with th

When we perform forward and back propagation, we loop on every training example:

```
3      (Get activations a(l) and delta terms d(l) for l = 2,...,L
```

The following image gives us an intuition of what is happening as we are implementing our neura