Why deep Neural Networks?

- Neural Networks having multiple hidden layers are better as we can think of each hidden layer detecting a singular feature such as say edges that would progress to facial features such as eyes, nose that would then later be able to detect entire faces.
- Thus,earlier layers of the neural network can be thought of as detecting simple functions, like edges. And then composing them together in the later layers of a neural network so that it can learn more and more complex functions.
- Deep networks seem to work well because of types of functions you can compute with different AND gates, OR gates, NOT gates, basically logic gates, where number of hidden units is relatively small. So the number of nodes or the number of circuit components or the number of gates in this network is not that large. However, if you're forced to compute this function with just one hidden layer, the computation ends up being exponentially large in the number of bits.

Building Blocks of Deep Neural Networks

Forward Propagation

- For a hidden layer l in our neural network, we can have as seen before $W^{[l]}$, $b^{[l]}$, $a^{[l]}$. Thus
  $\mathbf{Z^{[l]} = W^{[l]}.a^{[l-1]} + b^{[l]}}$, where $a^{[l-1]}$ is the input from the previous layer and $\mathbf{a^{[l]} = g^{[l]}(Z^{[l]})}$ is the output for the next layer.
- For the first iteration $a^{[0]}$ = X i.e. our input matrix.
- Here, we note that it is useful to cache the value of $Z^{[l]}$ for back propagation.

Backward Propagation

- Here, for a layer l going backward from layer l+1, we input $da^{[l]}$ and output $da^{[l-1]}$, $dW^{[l]}$, and $db^{[l]}$.

dW =

- It is convenient to similarly cache $dW^{[l]}$ and $db^{[l]}$ along with $Z^{[l]}$.
- Thus, using these derivative terms that are outputted, we can update $\mathbf{W^{[l]} = W^{[l]} - \alpha dW^{[l]}}$ and $\mathbf{b^{[l]} = b^{[l]} - \alpha db^{[l]}}$.

Once all layers are finished in forward and backward propagation, that completes one iteration of gradient descent.

Hyperparameters and Parameters

- Parameters are all variables that we have already discussed so far such as W,b,etc. Hyperparameters are values that determine the final value of the parameters that we end up using, for example
  - learning rate alpha

  - iterations
  - number of hidden units/layers
  - choice of activation function
- Basically hyper parameters are changed a lot based on our experiments with the code till we find the best value for it.