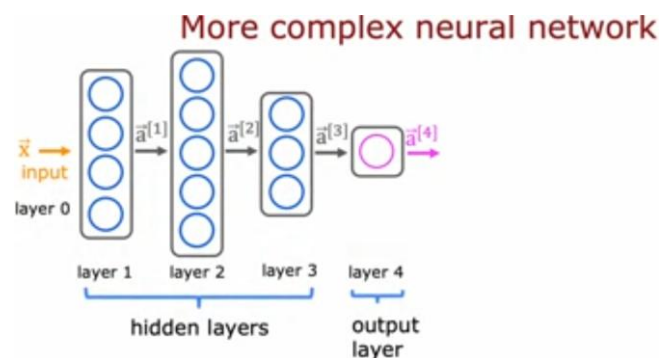


More complex neural networks

In the last video, you learned about the neural network layer and how that takes this inputs a vector of numbers and in turn, outputs another vector of numbers. In this video, let's use that layer to build a more complex neural network. Through this, I hope that the notation that we're using for neural networks will become clearer and more concrete as well.

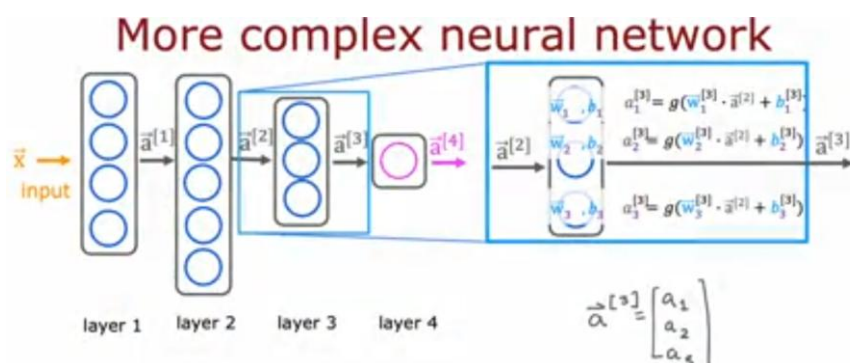
Let's take a look. This is the running example that I'm going to use throughout this video as an example of a more complex neural network. This network has four layers, not counting the input layer, which is also called Layer 0, where layers 1, 2, and 3 are hidden layers, and Layer 4 is the output layer, and Layer 0, as usual, is the input layer. By convention, when we say that a neural network has four layers, that includes all the hidden layers in the output layer, but we don't count the input layer.



This is a neural network with four layers in the conventional way of counting layers in the network. Let's zoom in to Layer 3, which is the third and final hidden layer to look at the computations of that layer. Layer 3 inputs a vector, a superscript square bracket 2 that was computed by the previous layer, and it outputs a_3 , which is another vector.

What is the computation that Layer 3 does in order to go from a_2 to a_3 ? If it has three neurons or we call it three hidden units, then it has parameters w_1, b_1, w_2, b_2 , and w_3, b_3 and it computes a_1 equals sigmoid of w_1 . product with this input to the layer plus b_1 , and it computes a_2 equals sigmoid of w_2 . product with again a_2 , the input to the layer plus b_2 and so on to get a_3 .

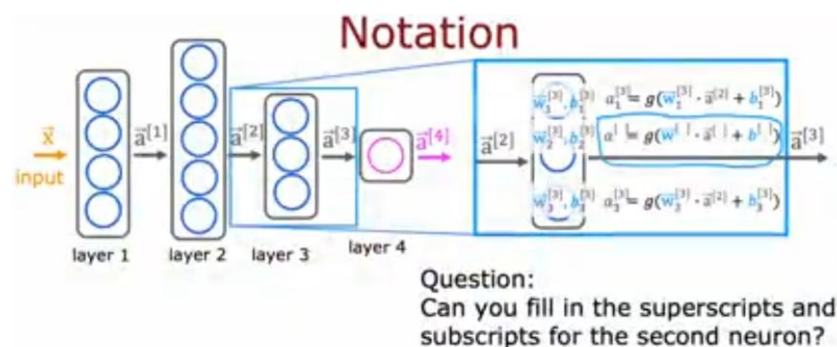
Then the output of this layer is a vector comprising a_1, a_2 , and a_3 . Again, by convention, if we want to more explicitly denote that all of these are quantities associated with Layer 3 then we add in all of these superscript, square brackets 3 here, to denote that these parameters w and b are the parameters associated with neurons in Layer 3 and that these activations are activations with Layer 3.



Notice that this term here is w_{13} superscript square bracket 3, meaning the parameters associated with Layer 3. product with a superscript square bracket 2, which was the output of Layer 2, which became the input to Layer 3. That's why it has a_2 here because it's a parameter associator of Layer 3. product with, and there's a a_2 there because is the output of Layer 2.

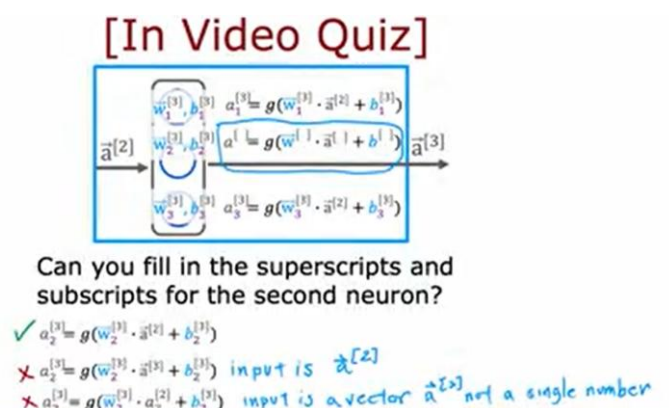
Now, let's just do a quick double check on our understanding of this. I'm going to hide the superscripts and subscripts associated with the second neuron and without rewinding this video, go ahead and rewind if you want, but prefer you not. But without rewinding this video, are you able to think through what are the missing superscripts and subscripts in this equation and fill them in yourself?

Once you take a look at the end video quiz and see if you can figure out what are the appropriate superscripts and subscripts for this equation over here. If you chose the 1st option, then you got it right!



The activation of the 2nd neuron at layer 3 is denoted by 'a' three two. To apply the activation function, g, let's use the parameters of this same neuron. So w and b will have the same subscript 2 and superscript square bracket 3. The input features will be the output vector from the previous layer, which is layer 2. So that will be the vector 'a' superscript 2. The second option is using vector 'a' 3 which is not the output vector from the previous layer. The input to this layer is 'a' two.

And the 3rd option has a two two as input, which is a single number rather than the vector. Because recall that the correct input is a vector, a two, with the little arrow on top, and not a single number.



To recap, a_3 is activation associated with Layer 3 for the second neuron hence, this a_2 is a parameter associated with the third layer. For the second neuron, this is a_2 , same as above and then plus b_3 too. Hopefully, that makes sense. Just the more general form of this equation for an arbitrary Layer 0 and for an arbitrary unit j, which is that a deactivation outputs of layer l, unit j, like a_{32} , that's going to be the sigmoid function applied to this term, which is the wave vector of layer

Notice, this is not l , this is l minus 1, like a_{l-1} above here because you're dot-producting with the output from the previous layer and then plus b_j , the parameter for this layer for that unit j . This gives you the activation of layer l unit j , where the superscript in square brackets l denotes layer l and a subscript j denotes unit j .

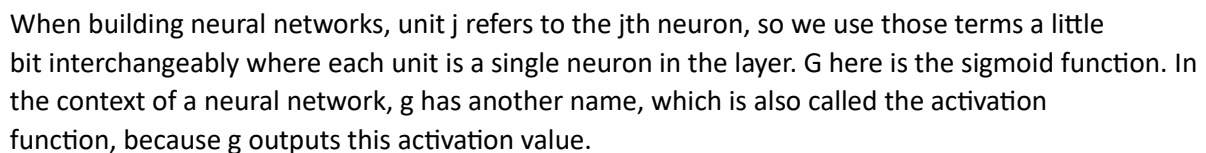


Diagram illustrating the forward pass of a neural network with 4 layers. The input vector $\vec{x}^{(0)}$ is fed into layer 1. The output of layer 1 is $\vec{a}^{(1)}$, which is fed into layer 2. The output of layer 2 is $\vec{a}^{(2)}$, which is fed into layer 3. The output of layer 3 is $\vec{a}^{(3)}$, which is fed into layer 4. The output of layer 4 is $\vec{a}^{(4)}$.

The diagram shows the calculation of the activation value $a_j^{(l)}$ for a specific unit j in layer l . The activation value is calculated as:

$$a_j^{(l)} = g(\vec{w}_j^{(l)} \cdot \vec{a}^{(l-1)} + b_j^{(l)})$$

where g is the sigmoid activation function. The diagram also shows the calculation of the output of layer $l-1$, $\vec{a}^{(l-1)}$, which is the input to layer l .

With this notation, you now know how to compute the activation values of any layer in a neural network as a function of the parameters as well as the activations of the previous layer. You now know how to compute the activations of any layer given the activations of the previous layer. Let's put this into an inference algorithm for a neural network. In other words, how to get a neural network to make predictions.