**Model Representation II**

To re-iterate, the following is an example of a neural network:



In this section we'll do a vectorized implementation of the above functions. We're going to define a new variable *zk*(*j*) that encompasses the parameters inside our g function. In our previous example if we replaced by the variable *z* for all the parameters we would get:

|  |
| --- |
| *a*1(2) = *g*(*z*1(2))  *a*2(2) = *g*(*z*2(2))  *a*3(2) = *g*(*z*3(2)) |

In other words, for layer *j* = 2 and node *k*, the variable *z* will be:



The vector representation of *x* and *z*(*j*) is:



Setting *x* = *a*(1), we can rewrite the equation as:

|  |
| --- |
| *z*(*j*) = Θ(*j* − 1)*a*(j − 1) |

We are multiplying our matrix Θ(*j* − 1) with dimensions *sj*×(*n* + 1) (where *sj* is the number of our activation nodes) by our vector *a*(j − 1) with height (*n* + 1). This gives us our vector *z*(*j*) with height *sj*. Now we can get a vector of our activation nodes for layer j as follows:

*a*(*j*) = *g*(*z*(*j*))

Where our function g can be applied element-wise to our vector *z*(*j*).

We can then add a bias unit (equal to 1) to layer j after we have computed *a*(*j*). This will be element *a0*(*j*) and will be equal to 1. To compute our final hypothesis, let's first compute another *z* vector:

*z*(*j+1*) = Θ(*j*) *a*(*j*)

We get this final z vector by multiplying the next theta matrix after Θ(*j-1*) with the values of all the activation nodes we just got. This last theta matrix Θ(*j*) will have only **one row** which is multiplied by one column *a*(*j*)) so that our result is a single number. We then get our final result with:

hΘ(*x*) = *a*(*j+1*) = *g*(*z*(*j*+1))

Notice that in this **last step**, between layer j and layer j+1, we are doing **exactly the same thing** as we did in logistic regression. Adding all these intermediate layers in neural networks allows us to more elegantly produce interesting and more complex non-linear hypotheses.