# Previous work

Earlier on Friday, I went to the lab to get the basics done; the data data import is all done, make sure to do the **Initialization** step outlined in the next section when you open the project, as that part isn’t persistent.

This lab reuses the accelerometer work we’ve done in Lab 2.

I have set up the accelerometer and the roll/pitch calculations, you might want to quickly put in a copy of the low pass filter such that we only use smoothed values. I heard the TA say that he trained the model specifically with smoothed values, so that might give the best results.

# Initialization

**Whenever you work on the project, make sure to run the *load\_from\_file.vi* instrument before doing anything**. You should select the weights CSV then the output CSV, in the right order, when running it. It will populate the shared variable *weights* that the model uses to predict anything.

# Next step

The general flow of this is as follows:

1. Gather data from accelerometer
2. Implement the summation such that 8 neuron cells are properly computing their values; you should be able to do that by bundling/clustering and then applying math to everything as a group.
3. Use the sigmoid function on the resulting values to produce 3 output values
4. Use the built-in max function to select the highest value of the sigmoid’s results.

This part is outlined pretty well in the documentation, you’ll grab the weights from the shared variable (use *Search* then select it, I haven’t been able to find it otherwise), and combine them with the data from the accelerometer and the roll/pitch This is the part that is spelled out as:

The result of that is the value of the node after the first layer; you repeat this to again with the 2nd set of weights to produce the output layer, which is 3 data points. It’s my understanding that you calculate the summation, then the sigmoid (below), and the resulting value is what is passed along.

Finally, the value with the highest end value is the one that decides which LED is lit. In principle, predictions should be right most of the time, from what I saw.

For best results, you might also want to *normalize* the values so they all fit in a range [-1,1], you can do that through arithmetic by dividing by the maximum and multiplying by the range of possible values.