1b – Design Decisions

* design decisions
* design implementations
* deviations

There was a general design that we collectively decided would be the best approach to attacking the recreation of the Convolutional Neural Network. This was a bottom up design. Working from the bottom we created the features of the CNN first. These features being the fundamental layers an image would be filtered through after being broken down through normalization and convolution. Once these features were established we then looked upwards towards a more directed approach of connecting them together to form the desired filtered network. Because we were given a table of the path of the network this step was easy to implement. The output of one layer feature became the input of another and so on. At this point the network was wrapped into a higher layer structure that was dynamic in nature. Acting like a black box function given an array of images and set of parameters a matrix output of probabilities could be produced.

Design implementations were also taken closely into consideration. Our Matlab coding reflects procedural programming more commonly found in the C language. This was done in creating supporting files, functional programming, and nested for loop structures. A few of the supporting files consisted of the convolve, the ReLU, and the maxpool function. Function calls in the neuralNet script called to these functions passing the necessary parameters along with it. These functions are much like functional programming in c++ where the purpose is to have a module or block of program code that handles a particular task. These modules come in handy for being reusable. By implementing this into our neural net we are able to greatly reduce the number of lines in our code while at the same time making it more robust. We have assurance in the function that when we call that section of code to handle a specific job it will stand by itself. Digging deeper into these function modules there was also some design implementation at a lower level. In the project description under the basic operations all of the computational building blocks were defined using summations. For example in fully connected the output is determined by summations in all dimensions of the image including the rows columns and channels. We extracted this summation notation and created nested for loop structures to handle looping through all iterations of the dimensions and computing the summations in a totaling type of way. In fact the for loop structure was used throughout the project even in the neuralNet code. It was decided that the for loop be used here for the vast amount of images that have to be run through the CNN to produce the probability vectors.

As a group few deviations from the project description were taken. The only noticeable deviations were towards the end in the grading section in which the questions were done out of order. For instance we partook on the exploratory mode of section 1e before testing some of the required performance evaluations.