Our proposed solution is to improve the accuracy of image recognition by building upon current deep learning models, intertwining various mathematical concepts and the python programming language to achieve a desired output[8][6]. Our model we will receive a series of pictures as input that will contain either a dog or cat, the result will be the system outputting the correct labels for each subsequent image that is being inputted to the model[8].

To achieve this, we will use various mathematical formulas on the input using TensorFlow as the framework. Once our model receives an image as input, it will be passed as a parameter to a composite function that will output the desired result. Inside this composite function is where the implementation of deep learning neural networking can be seen. The input will travel from node to node with it’s attached weights within the hidden layers -which is defined as chain of processing- passing it’s results to a activation or step function which then returns the outputs derived from the function.

After outputting the result, we calculate the difference between model predictions and the actual label, this process is the calculation for the loss for our function. Loss is equal to the summation of the inputs and their parameters minus the given label raised to the power of two. After the calculation of loss, we ratio the derivative of our loss value with the derivative of the parameters expecting the result to be zero, if the result is not equal to zero, we tweak our parameters and reissue the input back into the composite function until we can get as close to zero as possible. By tweaking our parameters and reissuing the result back into our function, our model should increase the correctness of a given image and associated label, overall, improving image recognition, allowing the model to “learn” and assimilate from the collected data.

The goal of this research is to analyze and process a large amount of data from observations and to see what we can determine from it [5]. It is important to understand that we will need to extract parameters from training data and use labels from testing data to be able to give on a new approach. For example, we did a fish assignment on different species of fish to use the height, width, and three lengths of training the model and we would take the rest of the parameters to test the model. (Explain more in details what the fish assignment was about. How big was the dataset? how many attributes? what was the target attribute (the one to be predicted), what were the independent attributes? what learning method have you used) To get the result of an image, we will use neural networks.

When we create a model, we will need to maintain a good level for it to perform great(what does this mean in terms of actual factors?). The training data set will be the input images that we will be implementing to train our model of getting use of seeing a dog or cat. The test data set will test, which is unlabeled test data, to see how the model is performing after being trained. Our images would go through layers of an input layer, hidden layer, and output layer [2]. Being able to use Neural Networks for better image recognition, we can pick out an original image so the specific object within the photo can receives parameters to be transform in each layer followed by being multiplied by the weights and biases, and then the output can contain the inputs that was transform from the hidden layer to the output layer to be able to determine what is being classify.

Our group will be creating a model to set it with our images. For our model, we will create it using Python. Then, we can put our images into two files for dogs and cats. In this research, we would want our model to be successful of knowing what image is being recognize from the actual image that is being carried out by layers of presenting the output of a dog or cat (Again are you implementing a NN or using a already package one?).