On the project, I implemented the [conv-relu-pool]xN -> [affine]xM -> [softmax or SVM] architecture, wrote scripts for model evaluation, and experimented with different weights initialization.

I used the Keras Sequential model API to build my network. I created a function that added each layer sequentially. I called this function to create a KerasClassifier which can be used to cross-validate the model layer. I initially started with one layer of each--a convolutional layer, a fully-connected layer, and a softmax layer--which achieved great results without overfitting: I achieved a training accuracy of 0.9898 and a test accuracy of 0.9847.

To evaluate our models in a different way, I wrote the script to cross-validate our models using the "model_selection" module from sklearn. The cross-validation method I used was a stratified cross-validation with five folds. We decided to evaluate our models by just reserving 20% of our training data as validation data as cross-validating was computationally expensive, and splitting the data 60/20/20 obtained similar results while being more efficient.

Finally, I tried to find the best weights and bias initialization for my model. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification by He et al recommends using a certain initialization for ReLU units. However, using this initialization in the Keras API actually decreased test accuracy by around 0.0014, a negligible decrease but a decrease nonetheless. Because of this, I used the default weight initialization for keras which is glorot uniform initialization.