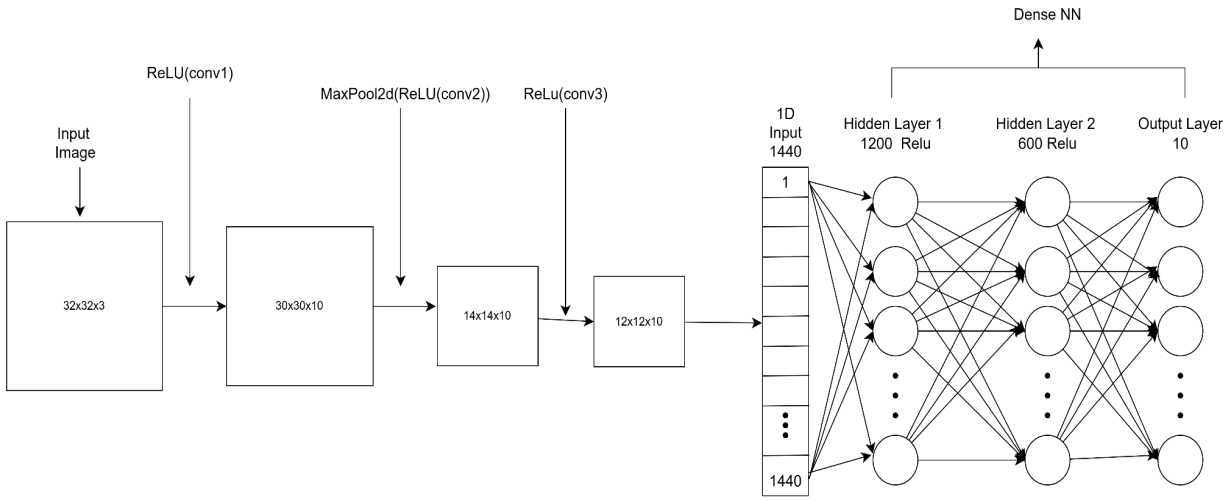
Assignment 2 Report

CNN Model

The CNN that is implemented in Assignment 2 is a convolutional neural network with 3 convolutional layers, 2 hidden layers and an output layer. The neural network can be seen in the diagram below.



Using the model we can derive the number of parameters

The number of weights for the fully connected nn is

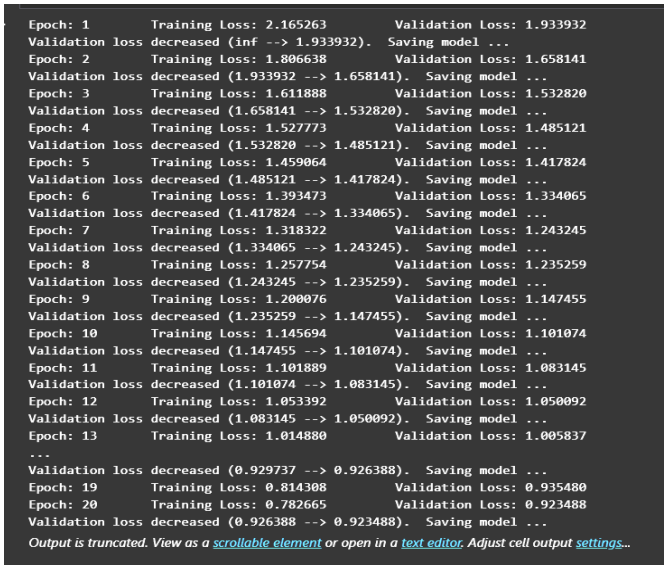
(1440+1) * 1200 + (1200+1) * 600 + (600+1) * 10 = 2,455,810
Conv1 = 3*10*9 + 10 = 280
Conv2 = 10*10*9 +10 = 910
Conv3 = 10* 10*9 + 10 = 910

The total number of parameters for this neural network is 2457910.

CNN Analysis

This structure was created by following the directions of the assignment as well as trial and error. Using this neural network the best accuracy I could achieve was around 67% percent with 20 epochs.

Looking at the validation loss at each epoch we can see a steady decline. While a better model would have an even steeper decline in validation loss this model shows that it does not overfit as we can see the same decline between training loss and validation loss. The average time to train the model was 6 minutes using an rtx 2070.



```
Epoch: 1        Training Loss: 2.165263         Validation Loss: 1.933932
Validation loss decreased (inf --> 1.933932).  Saving model ...
Epoch: 2        Training Loss: 1.806638         Validation Loss: 1.658141
Validation loss decreased (1.933932 --> 1.658141).  Saving model ...
Epoch: 3        Training Loss: 1.611888         Validation Loss: 1.532820
Validation loss decreased (1.658141 --> 1.532820).  Saving model ...
Epoch: 4        Training Loss: 1.527773         Validation Loss: 1.485121
Validation loss decreased (1.532820 --> 1.485121).  Saving model ...
Epoch: 5        Training Loss: 1.459064         Validation Loss: 1.417824
Validation loss decreased (1.485121 --> 1.417824).  Saving model ...
Epoch: 6        Training Loss: 1.393473         Validation Loss: 1.334065
Validation loss decreased (1.417824 --> 1.334065).  Saving model ...
Epoch: 7        Training Loss: 1.318322         Validation Loss: 1.243245
Validation loss decreased (1.334065 --> 1.243245).  Saving model ...
Epoch: 8        Training Loss: 1.257754         Validation Loss: 1.235259
Validation loss decreased (1.243245 --> 1.235259).  Saving model ...
Epoch: 9        Training Loss: 1.200076         Validation Loss: 1.147455
Validation loss decreased (1.235259 --> 1.147455).  Saving model ...
Epoch: 10       Training Loss: 1.145694         Validation Loss: 1.101074
Validation loss decreased (1.147455 --> 1.101074).  Saving model ...
Epoch: 11       Training Loss: 1.101889         Validation Loss: 1.083145
Validation loss decreased (1.101074 --> 1.083145).  Saving model ...
Epoch: 12       Training Loss: 1.053392         Validation Loss: 1.050092
Validation loss decreased (1.083145 --> 1.050092).  Saving model ...
Epoch: 13       Training Loss: 1.014880         Validation Loss: 1.005837
...
Validation loss decreased (0.929737 --> 0.926388).  Saving model ...
Epoch: 19       Training Loss: 0.814308         Validation Loss: 0.935480
Epoch: 20       Training Loss: 0.782665         Validation Loss: 0.923488
Validation loss decreased (0.926388 --> 0.923488).  Saving model ...
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Future Optimizations

While the assignment had us creating 2 hidden layers I would like to continue experimenting with just one hidden layer. The current model has too many parameters for the current complexity of the input. This results in a model that overfits easily in many changes I implemented such as adding epochs, increasing the number of the second hidden layer output, and increasing the number of outputs in the convolutional layers.

Another consideration is the size of the input of the fully connected neural network. This is controlled by the output of the convolutional layers. Since we started with a 32x32 image I felt that reducing that to a 6x6 image would lose too much data. However this is something that can be looked further into as that would allow a smaller input into the first hidden layer.