### Describe your code. What techniques, data structures, or approaches did you use to increase the efficiency of your code?

In our project, we de decided to use the sequential algorithm for training process. In our code we have a class named HandwritingNN, where the constructor initializes the inputs , size of the hidden hidden layers , size of the output and the bias inputs. Most importantly we initialize the random weights given for the inputs and the number of hidden newtons. We have made our program function to any kind of change in the number of hidden neurons, Input neurons and output neurons.

In the class HandwritingNN, we have two functions: training and test. In the training function first we divide the data set into training set and validation set. The 40,000 inputs are divided into 38,000 training data and 2,000 validation data. Several epochs of training and validation is performed, and the input is reshuffled in every epoch before dividing into test and validation sets. The training is stopped as soon as the validation gets an accuracy above 95% or when it loops through the number of epochs, which can be defined in the code. the . The maximum number of epoch After working on the training data and the validation data , input data is modified such that the bias input can be placed in the place of the output in the original training data. Input is then normalized so that the model can train properly.After that model is trained and the weights are updated using the back propagation technique. The final weights are used to calculate the output and the test accuracy is calculated.

We have used numpy for the manipulation of data and calculation process, pandas for importing the data files and matplot for plotting the results in a graphs.

In our early stages we got a very low accuracy, from 8% to 20% and our training vas taking a very long time. Our training would take 2 hour to compute 50 epochs and the accuracy was still very low. Then we decided to move to batch process, but we were constantly getting an error (exponential error). We realized that we were wasting a lot of time trying to fix the error, so we decided to move back to sequential and try increasing our accuracy and efficiency. We tried using different number of hidden neurons, and finally what did the trick was changing the random weights to avery low values (Multying by 0.001). Now our program was running much faster(about 50 seconds per epoch) with accuracy of 90+ in every epoch.

### 2. What is the architecture of your MLP? Why did you choose to use this setup?

Our input data has 4000 rows by 785 columns. Since we are doing a sequential training we train and update our weights single row at a time. So our architecture is 784\*50\*10. We tried different numbers of hidden neurons, but our best accuracy and efficiency was with 50 hidden neurons. We have 784 input neurons , 50 hidden neurons and 10 output neurons.

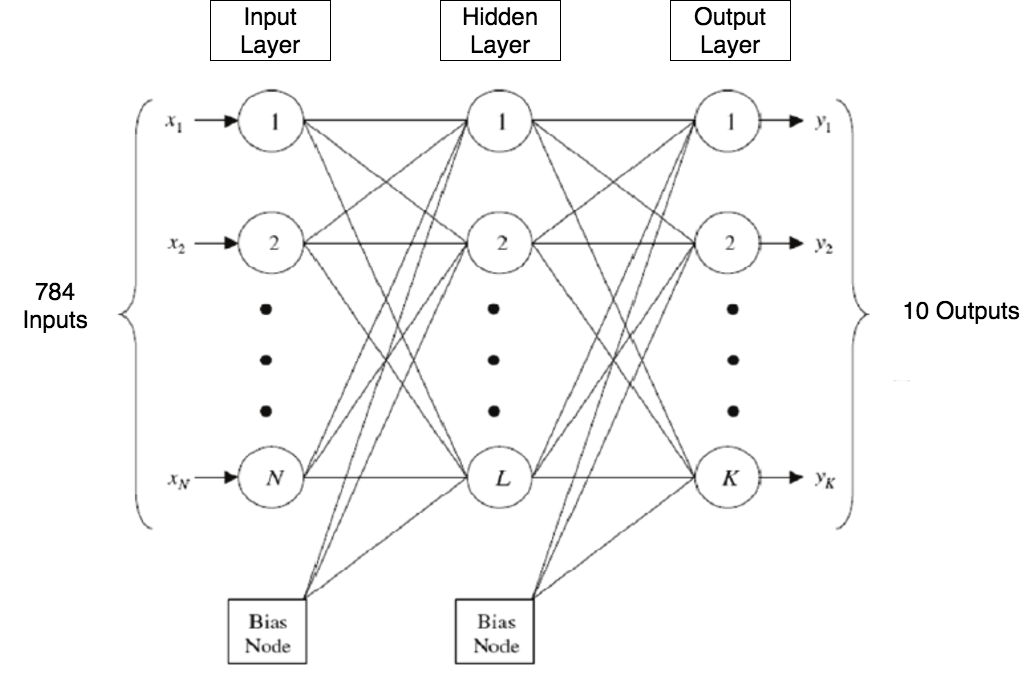
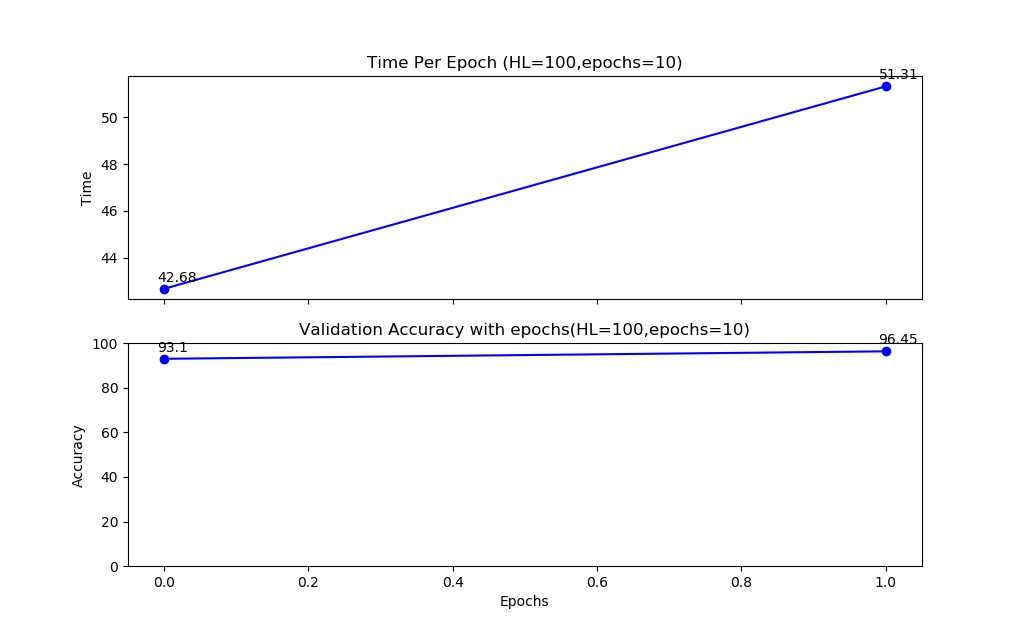
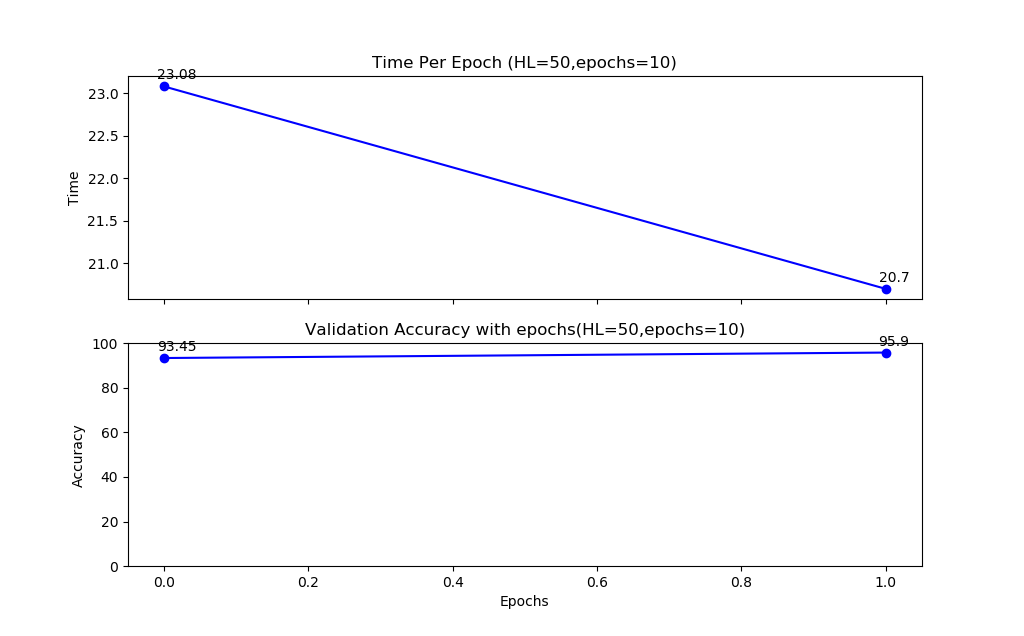
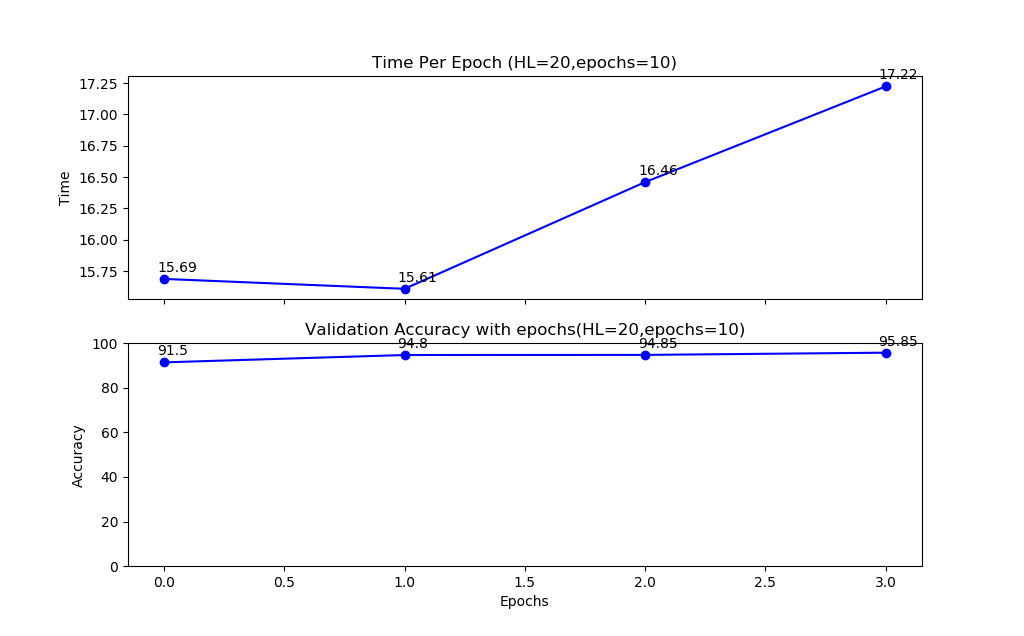
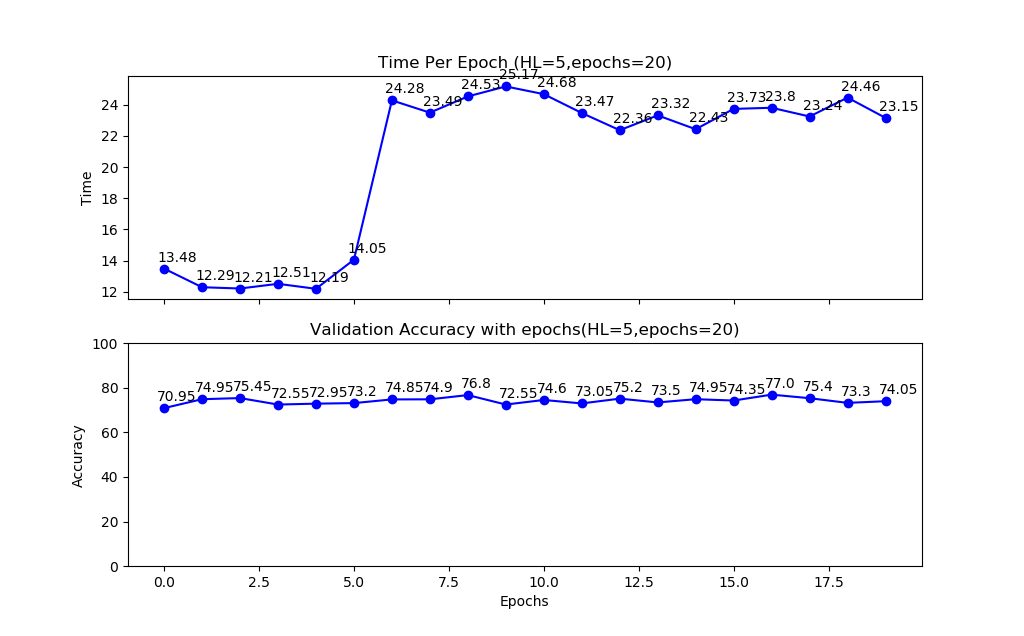


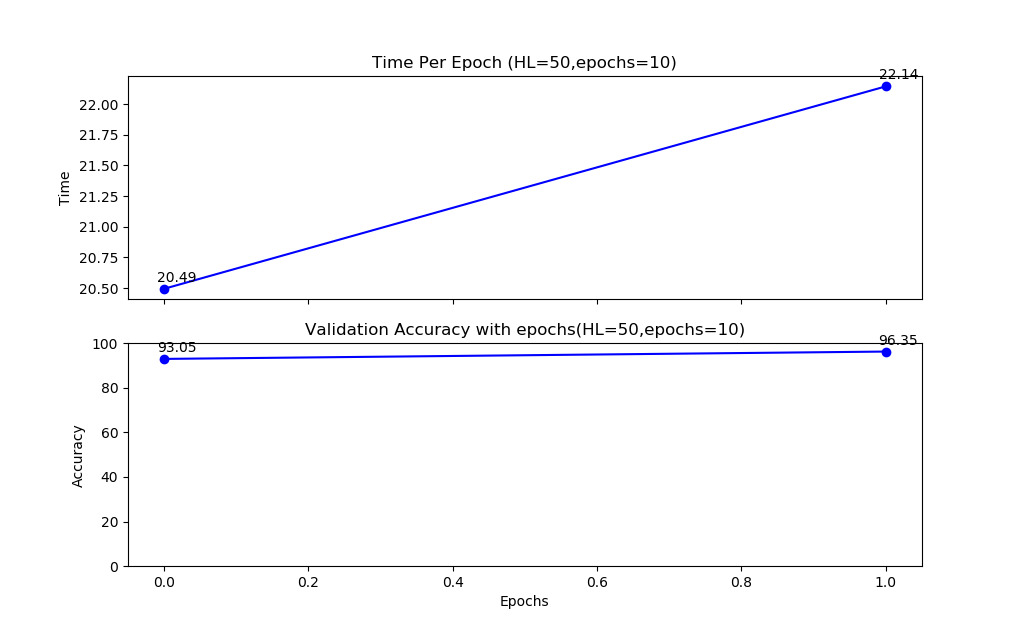
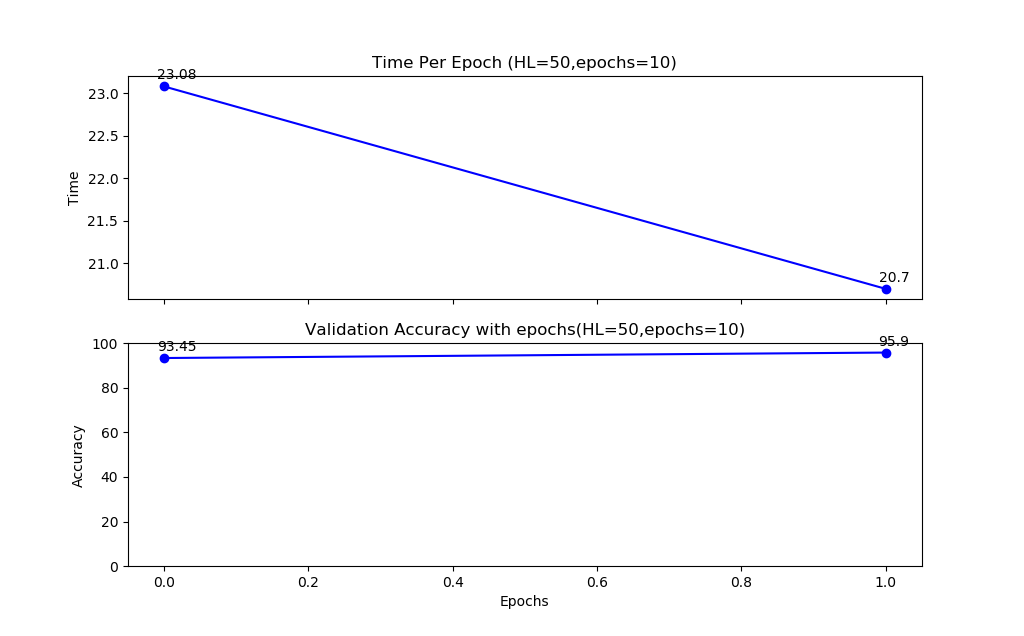
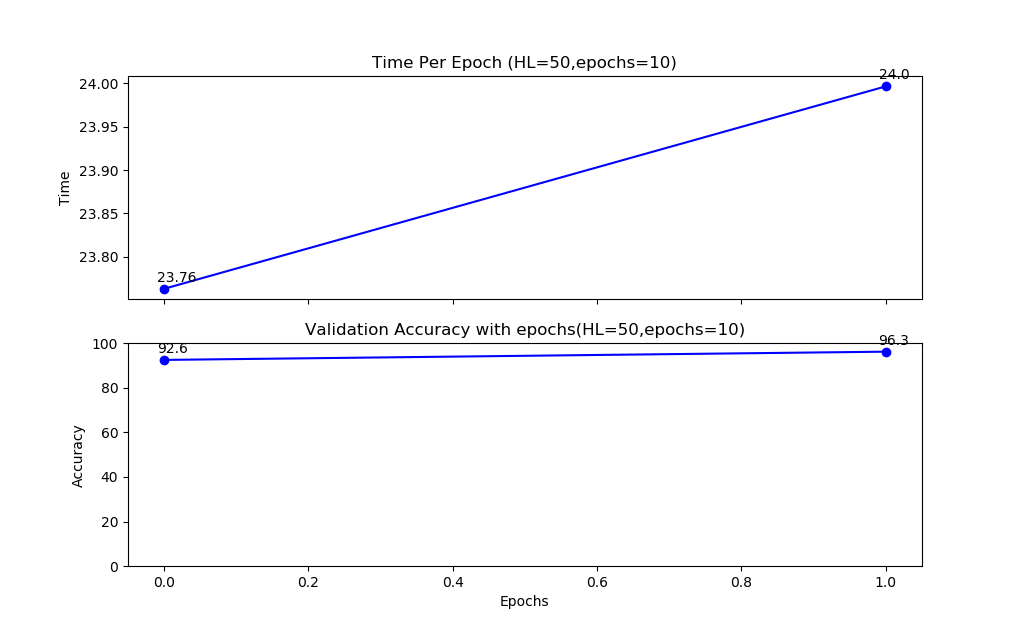
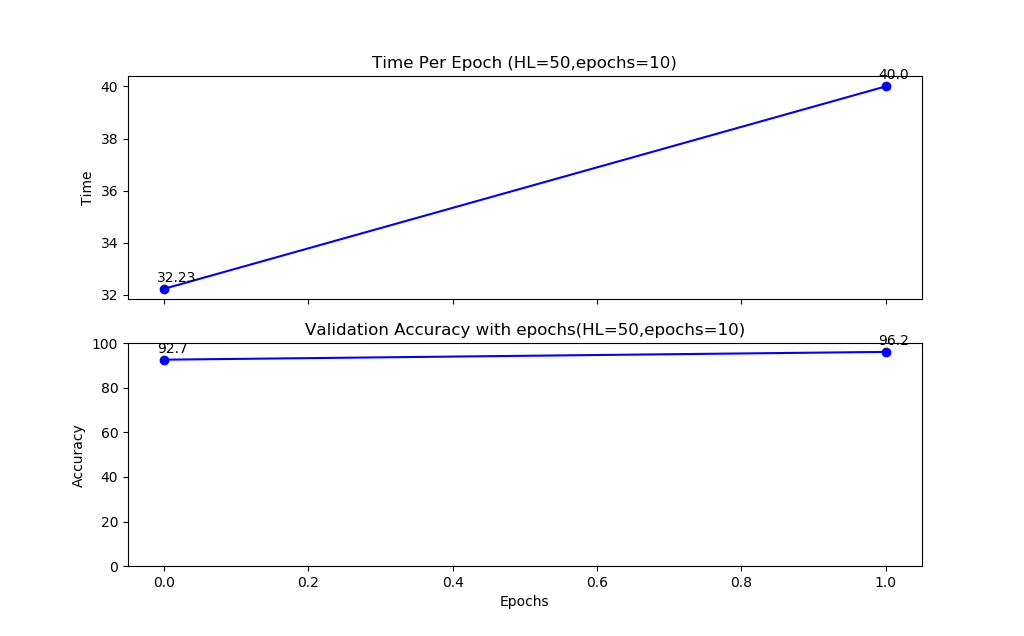
Figure: Architecture of MLP

The following figures shows different accuracy with time for different number of hidden neurons:



### 3. What is the effectiveness of your classifier? (accuracy, error rate, etc)

The training phase is only completed if the accuracy is above 95% or if the given number of epochs is completed. To do this we take 2000 data rows from the training data set before each training epoch for validation. After the the system is trained with the 38000 data, validation is done and validation accuracy is calculated. So the validation accuracy is always above 95% and test accuracy is always above 90%.

Following chart shows the accuracy and runtime of our model. 

### 4.How long does your model take to train?

Each epoch in our final model is taking 20 to 30 seconds for training and validation. Our best model is taking 2 to 3 epochs ho get to a 95% accuracy in validation. So our model is taking less than a minute for the training process. The figures in answer number 3 shows the time taken for our training.

### 5. Did you choose batch, sequential, or mini-batch training? Why?

We choose the sequential training because it worked fine for us. Using sequential training we got the good validation accuracy and the good test accuracy. So we focused more on the sequential rather than the batch training. We tried batch training too but we got errors in the sigmoid function saying exponential overflow. Using the sequential process, we got the validation accuracy of 95.35 with 50 hidden neurons in just two epochs. We got this accuracy in less than a minute of runtime, so we decided to stick to sequential processing. Also, our data contains a lot of zero values, for which sequential processing would be a better option.