Machine Learning: Assignment 1: Eigen Digits

Vivek Kumar Khetan
UT-EID - vkk287

---

1. Introduction:

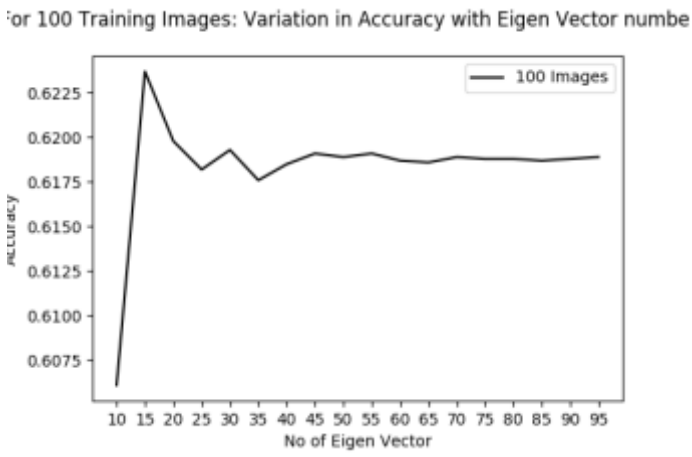
The main goal of this assignment is to learn about eigenvector decomposition of covariance and its analysis. In this assignment, we need to train the K-Nearest-Neighbor algorithm in a way that it uses minimum computation power and gives the maximum prediction accuracy.

This includes converting Matlab training data file to get trainImages, testImages, trainLabels and testLabels. The assignment has many interesting parts. The first interesting part is to get the similarity matrix for training dataset. Similarity matrix can be defined as an average value of each pixel position for each Image. It is due to the ambience condition and can be seen as effect of ambient brightness or lighting conditions. The second interesting part is to project the dataset into the eigen space and reduce its dimensionality. As using all the train Images will be little slower than using few of the Images, I used few of the training Images and then calculated the covariance matrix of training Image dataset. After calculating the eigenvectors from training set of digit data, the next important step was to project the training data point into the eigen space to reduce its dimensionality and to again decrease the use of our computational resources. Then, I subtracted the similarity matrix from the test Image dataset and projected the test Data points to the Eigen space. I used Scikiit Learn's implementation of K nearest neighbor to train and test my model. I experimented using various value for train Image data point and various number of eigen vector to project my data points. I was able to get the accuracy up to 86% using a maximum of 700 data points to train the model and then tested on 10000 data points. I used python3 kernel in Ipython notebook and a number of different python libraries to do various computations.

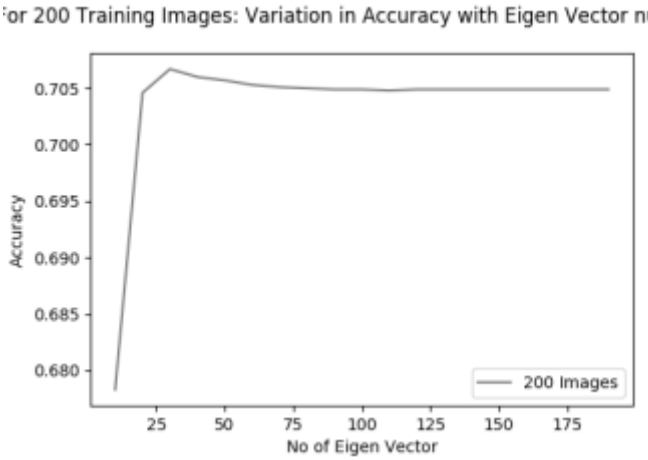
2. Experiment:

Most interesting part was to do various experiments with the data, reconstruct the test digits in the eigen space and to visualize that how the projected test point varies from the original data point, although both are same. I also tried and calculated accuracy with varying no of eigen vectors as well as training Images. Each eigen vector project some of the Image variation on the eigen space, it was interesting to see how sometime even increasing the number of eigen vector was not resulting on increase of accuracy, sometime it even decreased the accuracy.

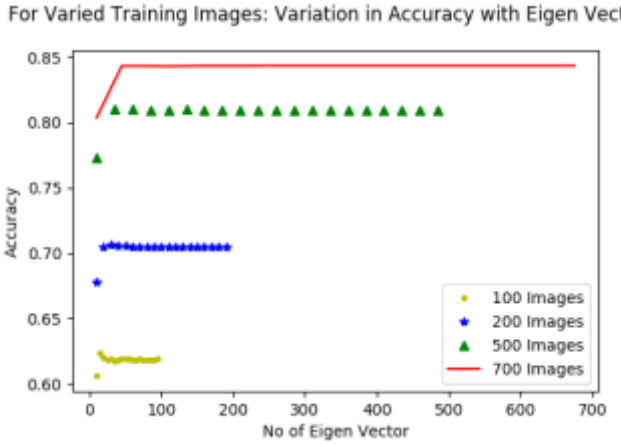
3. Discussion:



In this plot with 100 training data point and 10000 test data points. It is interesting to see how the knn model is achieving a maximum accuracy when around 15 number of eigen vectors were used. Although the decrease in accuracy after increasing the number of eigen vector was in second decimal points but it is clear that in this case around 15 eigen vectors were able to capture the maximum variation in the data points.



Almost similar results were evident when 200 Image data points were used for training although we will need around 25 eigen vectors to capture the maximum variation in our training data points.



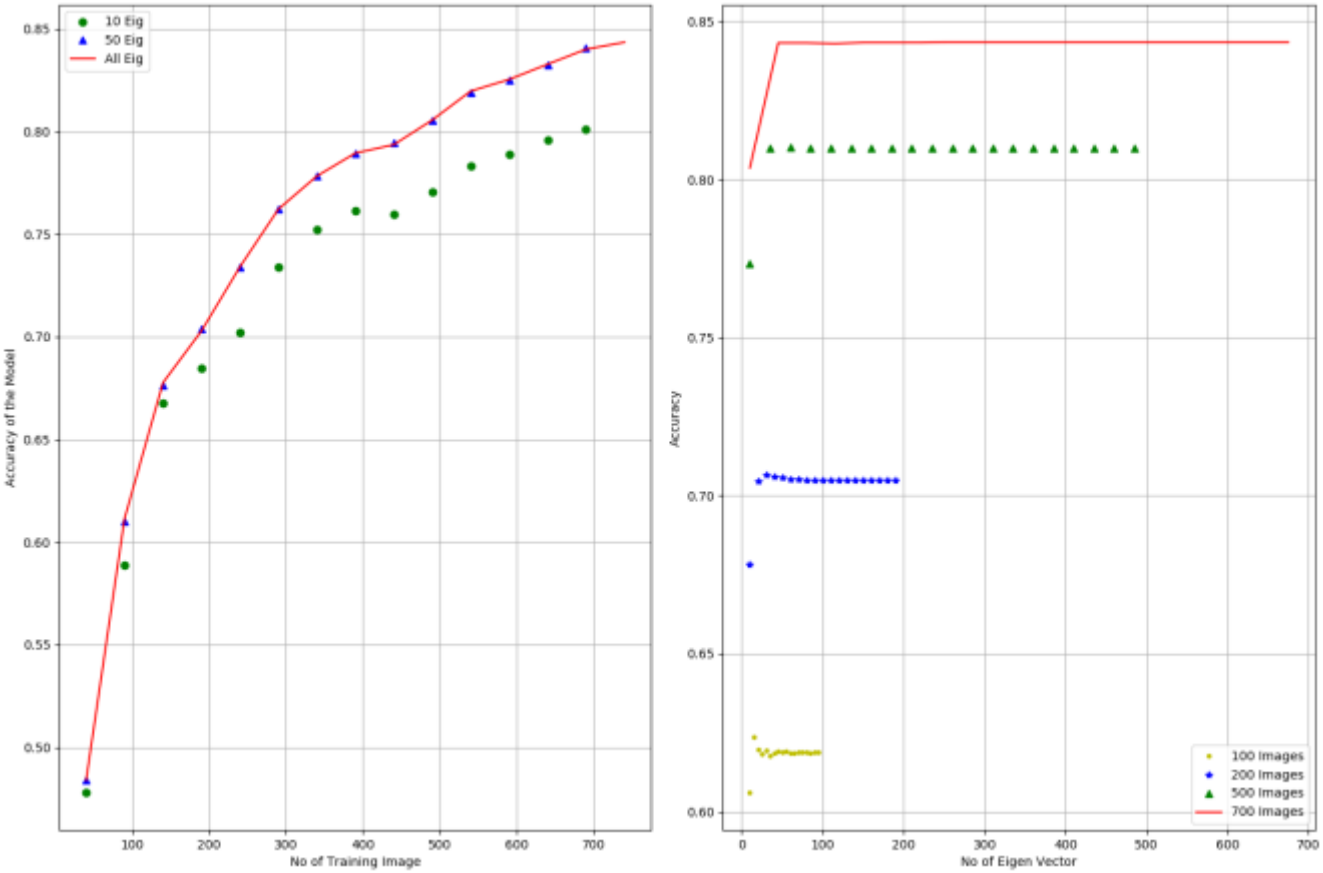
I also performed this experiment by using various number of training Image data point and varying the no of eigen vector in each of the experiment. The image on the left shows how accuracy vary with various training data point and by changing Number of eigen vectors for capturing the data points variance.

4. Conclusion and Future Work:

It is very interesting to see that only increasing the number of training data points are not able to improve the accuracy drastically. One good way to train our model can be to use all the training data points we have and project them into eigen space using all the eigen vectors. It will definitely capture the maximum variance and the model trained this way will have superior accuracy but at the a very high computational cost. Another way of doing the same thing can be to use the optimal number of eigen vector so that it captures the enough variance of data point and gives us a good accuracy measure.

The image below is a summary of all the experiments I performed on the data points. The Plot on the left shows variation in accuracy when three different number of eigen vectors were used for projection and for each of the case training image size was continuously changed. It is evident that using 700 data points for training and all the eigen vector for projection gave around 86% of accuracy but we can also achieve an accuracy of around 80% using only 650 training Images and just 10 eigen vectors.

The plot on the right was created by using fix number of training Image data points in 4 different cases and varying the number of eigen vector for projection in each of the case.



We can achieve a test accuracy up to 80% on 10000 test Image data points by using just 450 training Image data points and around 50 eigen vectors.

For the future work, I would be try to make an animation that shows the variation in accuracy as we change number of training data points as well as no of eigen vectors. I would also like to explore the ways to find the exact eigen vectors and a subset of training data points to capture the maximum variance and obtain the best accuracy at