Vignesh Ravindranath
vgr325

Classifying the NMIST digits using their Eigenvalues (PCA)

1 Overview

The purpose of this assignment is to build an efficient algorithm(s) to classify NMIST digits. The NMIST dataset contains $28x28$ pixel images of hand written numerical digits, and when unraveled, contain 784 pixels (features) per image in total; the training dataset in total contains 60,000 images. This can quickly become computationally expensive as the special dimension is extremely large. One standard way to proceed would be to representing the given data with a reduced number of features. In other words, the same image can be represented with less data (features), while retaining the majority of its information. Principal component analysis, or PCA, is one such method to do so. It works by projecting the data onto the first k eigenvectors that have the largest eigenvalues. Once the data has been projected onto a smaller space, it can be classified using a k-Nearest Neighbor (kNN) algorithm.

2 Method

To reduce the dimensionality of the dataset, we first need to find the eigenvectors and eigenvalues of the sample covariance matrix Σ. The sample covariance matrix is represented as $\Sigma = \mathrm{AA^T}$, where A is a $x \times k$ matrix, x is the number of pixels, and k is is the number of samples taken from the original dataset. Note that we want k to be less than x. However, if we look at the equation,

$$AA^T\boldsymbol{v} = \mu\boldsymbol{v}$$

we see that the covariance matrix is a $x \times x$ matrix $((x \times k) \cdot (k \times x))$. Rather, we can attempt to represent Σ as a smaller matrix that still captures the majority of the variation by considering,

$$AA^T A\boldsymbol{v} = \mu A\boldsymbol{v}$$

where $A\boldsymbol{v}$ is the eigenvector of Σ. This is a smaller space because $A^T A$ is only a $k \times k$.

3 Experiments

First, we need to ensure that the PCA algorithm (*hw1FindEigendigits*) works as intended. As mentioned previously, the PCA algorithm finds the mean and eigenvectors of the reduced covariance matrix. The eigenvector matrix can be multiplied by the sample data A to find the eigendigits. The first five original MNIST images and the projected (500) eigendigits can be seen in **Figure 1**. Next, we can attempt to reconstruct the images using the same mean and eigendigits matrix. In essence, we are reducing the original image (losing information) and using the mean and eigendigits matrix to reproduce the original image. As seen in **Figure 2**, with 100 eigenvectors, the reconstructed images are similar to the original images with minimal loss of information.

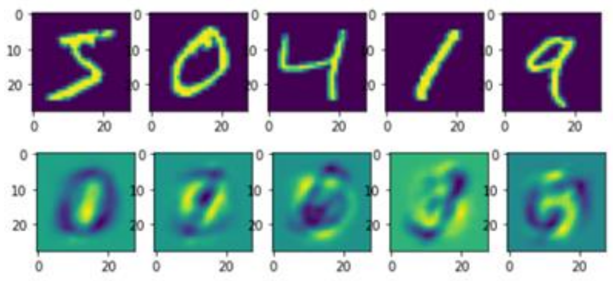

Figure 1: Original Images vs Eigendigits

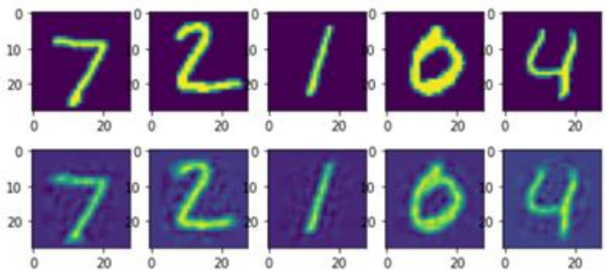

Figure 2: Original Images vs Reconstructed Images

Next, we can attempt to optimize our algorithm and discover inherent characteristics of the data. The MNIST dataset provides 60,000 training examples and 10,000 testing examples; however, for the purpose of time, we can train the kNN on a smaller training dataset. **Figure 3** illustrates the accuracy as the size of the training dataset increases. From the plot, it seems as though the accuracy plateaus at an accuracy of about 0.92 as the training size reaches 5000 training points. So for further testing, I will be using close to 5000 training points (I actually use 10% of the 60,000 training examples as that is a round number).

However, what we really want to know is the minimal number of features needed to represent the original data without much loss of variability. To test this, I calculated the accuracies (kNN, k=3) with variable eigenvector dimensions. **Figure 4** indicates that the accuracy is largest when the eigenvector dimension is 50 (accuracy = 0.93). In other words, the MNIST images can be represented by the top 50 features without much loss of information.

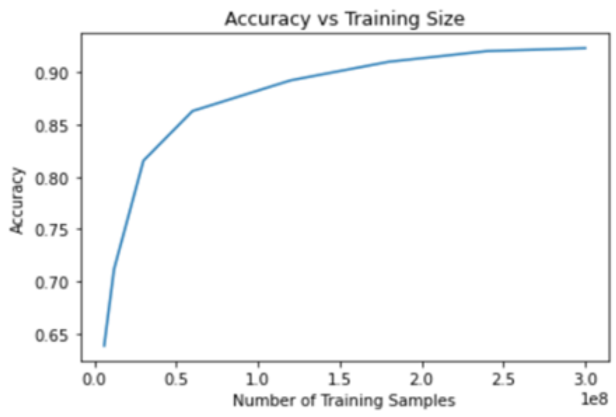

Figure 3: Accuracy vs Training Size (k=3)

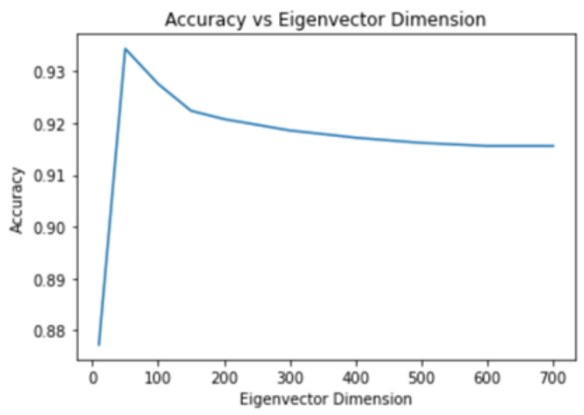

Figure 4: Accuracy vs Eigenvector Dimension (k=3)

Lastly, we can optimize the number of nearest neighbors the kNN uses to classify new images. **Figure 5** illustrates the accuracy vs nearest neighbors (k) and indicates that the optimal k value is 3.

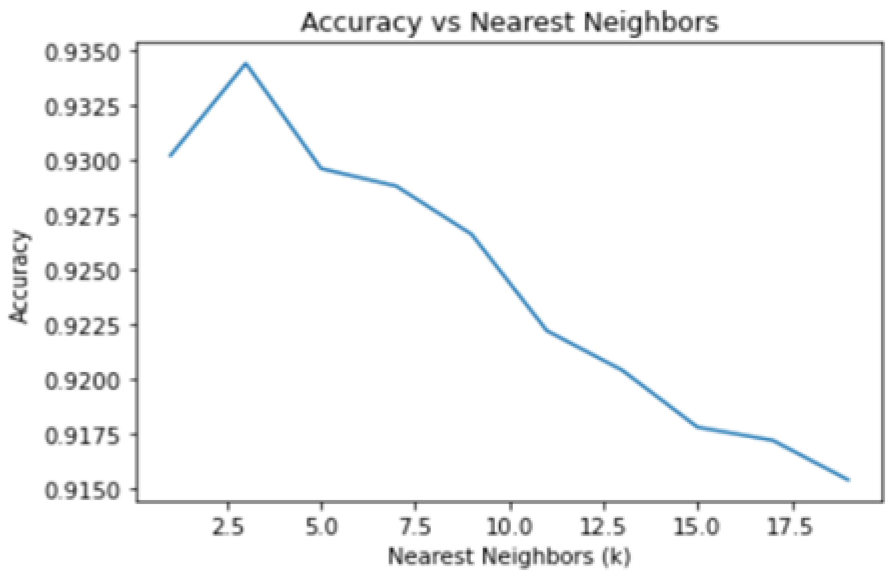

Figure 5: Accuracy vs Nearest Neighbors (k)

4 Results and Discussion

The overall accuracy of the combined PCA and kNN algorithms was found to be 0.9344, using the top 50 eigenvectors from 700 eigenvectors and a k value of 3 (kNN). The accuracies were tested on the first 5000 testing data points. The PCA algorithm (*hw1FindEigendigits*) illustrated that we can reconstruct images with variable eigenvector dimensions. As the number of dimension increases (towards 700), the images retain the majority of their information; however, the algorithm requires more time to execute. Similarly, as the number of dimension decreases (towards 0) the images lose more and more information. The PCA algorithm proved to be a robust algorithm to reduce the feature space of NMIST images while preserving the majority of the images' information.