CS391L Machine Learning Assignment 1

Srinath Tankasala,
st34546

Abstract

In this assignment, a classification method to identify handwritten digits is developed. OCR systems have been around for more than 20 yrs and are extensively used in the banking industry to read cheques. They rely on computers being able to recognize handwritten digits on an image. A KNN (K-Nearest Neighbour) approach along with PCA (Principal Component Analysis) is used to create a digit classifier. The accuracy and influential parameters for the classifier are studied to arrive at the optimal classifier.

I. Introduction

The problem of classifying digits is well explored in literature. With the rise of digital cameras and imagery, it is possible to apply statistical methods on image data. For this assignment, the standard MNIST image dataset (D) is used. This consists of grayscale training images and test images for tuning our classifier. The MNIST database has images of size 28x28 pixels. For building the classifier, all pixels in the image are treated as features of the dataset. Thus each image in the dataset has 784 features.

There are 10 different labels possible for every image (0-9). Thus our classifier model is a function that maps the image features to a label, i.e. $f : R^{784} \rightarrow R$. For classification of an image into one of these labels, we use the k nearest neighbour method. However, since the KNN method relies on finding the distance between the new image and all the images in the dataset, it becomes slower with increased training data.

A large training data set makes the KNN method more accurate but results in a trade off with computational time. This is where a simplification using PCA is helpful. This is discussed in the next 2 sections.

II. Classifier: K Nearest Neighbour

In the KNN method, the function value at any given point (X) is approximated to the function value of the point at closest distance in the training dataset. The most intuitive one is the 1st nearest neighbour where the function value is approximated to the value of the point at closest distance in the dataset.

The distance measurement typically used is the Euclidean norm, L2 norm. However, L1 or L3 norms can also be taken as they represent a measure of distance. L1 norm is also commonly referred to as the metropolitan distance. In this assignment the L1 norm, i.e. metropolitan distance was used as it is computationally faster than L2 norm.

$$f(X) = f(X_0)\ ,\ where\ \min_{X_i \epsilon D} ||X - X_i||_1 = ||X - X_0||_1, \text{D is the training data set} \tag{1}$$

The performance of the classifier is determined by how well it is able to calculate the decision boundary. The decision boundary separates data labelled in different classes. Points that fall on the decision boundary have ambiguous labels. Since we do not know if the shape of the decision boundary is linear for this problem, we go with kNN instead of logistic regression. A 1st nearest neighbour does not give a smooth decision boundary and handles data outliers poorly.

To get a robust decision boundary and reject outliers, a weighted kNN approach can be used where the k nearest neighbours are weighted to get the actual function value. There are many weighted kNN algorithms available in literature, the simplest of which is a majority vote kNN. For such cases K is taken as odd so that a winner exists. In cases where there is a voting tie between 2 or more labels, K value is increased until the tie is broken.

$$f(X) = mode(f(X_k)) \tag{2}$$

where X_k is set of k nearest neighbours

III. Principal Component Analysis

Each image in the training data set consists of 784 features. Not all features contain equal amount of information about the image however. For instance the border pixels around the image are always black and do not contribute any information that can help identify the digit in the image.

Hence, instead of using all 784 features to determine the digit, PCA is used to obtain the most important feature components. This reduces the number of features that we have to work with making the problem easier to solve. This also helps avoid overfitting the training data set.

Principle Component analysis is a mathematical technique used to extract the linearly independent components in the data and obtain an orthogonal basis. The identified components of this orthogonal basis are then ranked based on eigenvalues. Since not all components are equal in weightage, we can obtain a good approximation of the image by choosing even a small number of the principal components.

Given our dataset A of size 60000x784, we need to find its principal components. Let one of the principal components be an image v, 784x1. We wish to find v such that the sum of projections of all points on that data set onto v is maximized.

$$\begin{aligned} &maximize\ projection\ A * v \\ &\rightarrow \max_{v}\ (Av)^T * (Av) \\ &\implies \max_{v}\ v^T(A^TA)v \end{aligned}$$

The above has become equivalent to finding the induced 2 norm of the matrix A^TA and can be obtained by solving for its eigenvalues and eigenvectors. The eigenvectors are bound to be orthogonal as A^TA is symmetric positive semidefinite.

Hence, the steps to finding the principal components of our data set A(60000*784) are:

1) Remove mean image from the dataset , i.e $Y = A - column\ mean(A)$
2) Cost/Covariance matrix (C), = $\frac{1}{60000}Y^TY$
3) Solve for V, $CV = V\Lambda$, where Λ is a diagonal matrix of the eigenvalues of C
4) select number of columns (principal components) to keep (n)
5) For new test image I (784x1), obtain its projection, $p = (I - column\ mean(A))^T[v_1v_2...v_n]$
6) reconstruct image from projection, $I_{proj} = [v_1v_2v_3...v_n]p^T + column\ mean(A)$

The mean image mentioned above is the image obtained by taking the mean of all images in the data set. Images can be approximated with one's choice of principal components. One can even perceive this visually as below:

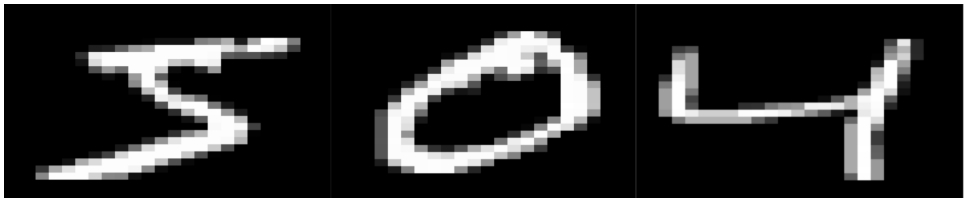

Fig. 1: First 3 images of the MNIST training set with all 28x28 pixels

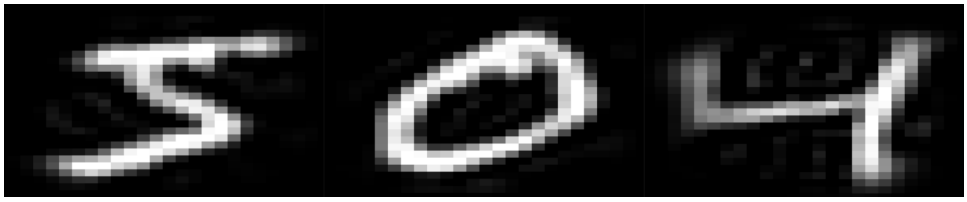

Fig. 2: Projections of the first 3 images of the MNIST training set with 100 principal components

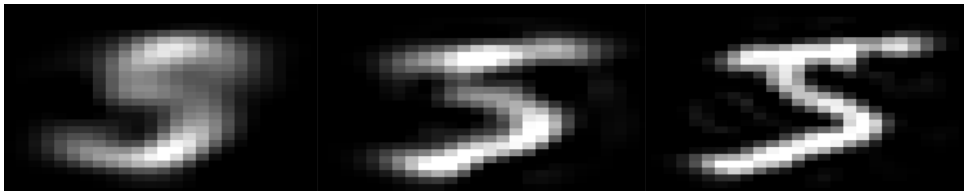

Fig. 3: Projections of the first image using 10, 30 and 100 principal components respectively

It can be seen from Fig 3 that a good approximation can be achieve using only 30 components. The principal components/ eigenvectors by themselves are images. They represent the image units needed to reconstruct a digit. To visualize them, the first 3 eigenvectors are shown below. Note the brightness of the normalized eigenvectors have been scaled by 3000 to make them visible.

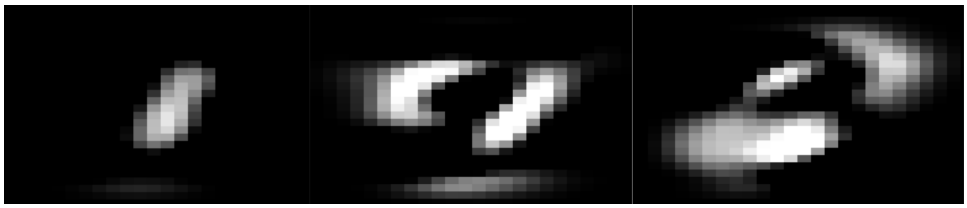

Fig. 4: First 3 principal components of the entire(60000) MNIST training data set

Hence the final methodology for digit recoginition is:

1) Decide on the size of the training set A
2) Find the eigenvectors/PCs of the Covariance matrix of (A-colsum(A))
3) Decide the number of eigenvectors to use for approximating your images
4) Project the training data set A onto the selected Eigenspace. Let it be called A_{proj}
5) Given a test image to identify, 'I'. Project 'I' onto the selected Eigenspace, say I_{proj}
6) Find the k nearest neighbours to I_{proj} in A_{proj}. Let us call it $k(I_{proj})$
7) The label of I is equal to the mode of labels of $k(I_{proj})$

IV. Results and Discussion

Given the 7 step process in the previous section, the identification accuracy of our classifier depends on 3 different parameters:

1) Training set size
2) Number of principal components
3) Number of nearest neighbours (k) chosen

A. *Effect of the size of training data set*

It would be intuitive that the having more data results in better accuracy of the classifier model. However the returns from increasing data set diminishes past a certain point. This means that we may not get any significant improvement in accuracy by collecting more data. This is important as collecting data takes time and money.

When selecting a subset of the training data, it is important that our subset sample captures the distribution of the digits, i.e it must not induce bias. This means that we must ensure that the subset has equal likelihood of picking any digit. If our sample is say more than 10% for one digit and less than 10% for another digit, there is training bias. This is not accounted for here as it is beyond the scope of this assignment. For simplicity the first N images of the training set were chosen. The graph below shows the labeling accuracy of the classifier for 10 and 30 principal components and using k=1.

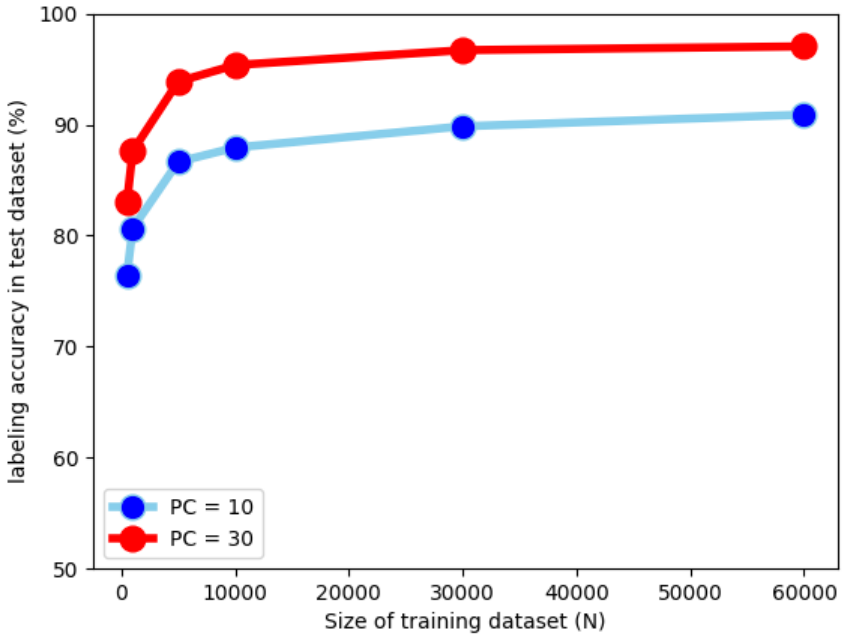

Fig. 5: Test accuracy vs size of training data set, PC is Principal components

As can be seen above, the algorithm improves its accuracy as the amount of training data increases. However this increase in accuracy slows down and from 30000 to 60000 we see a mere 2% improvement in performance. As expected the model that uses more PCs gives higher accuracy for the same training data size. The model that uses only 10 principal components underfits the training data, so it does not perform as well on test images compared to the model with 30 PCs

B. *Effect of number of principal components (PCs)*

Choosing higher number of PCs increases the accuracy of our model only upto a point. Choosing excess number of PCs will cause the model to overfit the training data and result in lower performance on the test data. This means that as the number of training data increases, the testing accuracy decreases. Hence it is best to choose an optimal number of PCs to not overfit the training data.

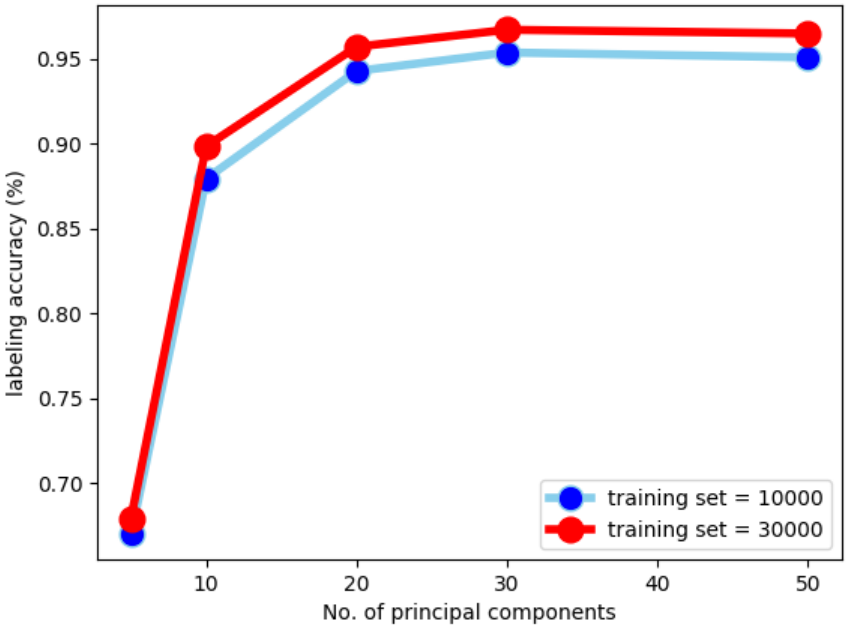

Fig. 6: Test accuracy vs no. of principal components

As can be seen above, for the N = 10000 case, the training accuracy slightly falls from 95.7% to 95% as PCs goes from 30 to 50. This is even more visible from the case of N = 30000 where our model has overfit the training data resulting in poorer accuracy.

The effect of choosing multiple k nearest neighbour has not been studied in this work. However, it is very useful in cases where the decision boundary is highly non-linear and the training data is noisy.

V. Conclusion

In conclusion, a kNN classifier enhanced with PCA was implemented to classify the digits in a handwritten image. The choice of training data size and number of Principal Components was explored.

If we notice that increasing the number of components is leading to a decrease in the overall testing accuracy, then the model is overfitting the training data and we must reduce the number of PCs being used. A large training data set makes the KNN method more accurate but results in a trade off with computational time, hence an optimum recognition time must be aimed for. The highest classifier accuracy of 96.7% was observed at PCs = 30 and training data size = 30000 and k=1.