Coding Project 4: Teaching a Computer to Recognize Written Numbers

Justin Hexem

3/10/2023

Abstract

In this project, we used a machine learning algorithm to be able to examine an image of a hand-drawn number and discern what number the drawing is referring to. To do this, we start with a training set of $30,000$ images of hand-drawn numbers that we will then use to "teach" our MATLAB script how to recognize each digit. This is done through the use of edge detection, principal component analysis, and linear discriminant analysis to separate the data for each image and obtain our threshold value. This threshold value is used as the reference that will help us sort our $5,000$ test images into their respective digits. Once we test these new images, we obtain a success rate above 95%.

1 Introduction

To be able to reliably discern which of the digits $0-9$ are present in the image we are looking at, we must take advantage of machine learning. We can think of machine learning as instructing a computer to design its own algorithm to solve a problem. This can be done through the use of "training data" that is fed into the program to help the computer detect patterns that might not be apparent to our human senses. Once the computer program is trained, we can feed it new data that is has not seen before to evaluate how good the computer program is at solving the problem it was designed for.

In this report, we will use machine learning to train a MATLAB program to recognize numbers from hand-drawn images of numbers. This will be

achieved through the use of several mathematical operations such as the wavelet transform, the singular value decomposition, a generalized eigenvalue problem, and projecting onto an appropriate vector that separates our data. These operations will allow our program to take in as an input $30,000$ images of hand-written numbers as training data, and an additional $5,000$ images of hand-written numbers as data to test the success rate of our program. We will examine which numbers look similar to our program and which numbers the computer can easily distinguish to try to determine the defining features that our program looks for for each number.

2 Theoretical Background

In this section, we will discuss the process of linear discriminant analysis and how it can be used to solve the particular problem we are working on.

2.1 Linear Discriminant Analysis

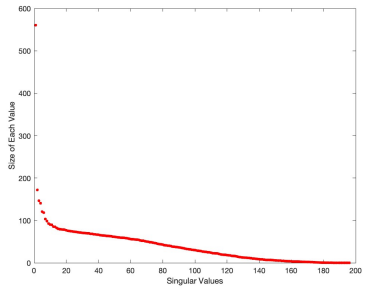
Linear discriminant analysis is the process of taking data that is in one of two categories and maximizing the distance between the inter-class data while minimizing the distance between the intra-class data. What this means is that we must find the vector $\vec{w}$ such that the difference of the means of each category is maximized, while the variance of each category is minimized. This can be accomplished by first computing the covariance matrices for the between-class data and for the within-class data. We may denote the the between-class covariance matrix as S_B and the within-class covariance matrix as S_W. More formally, these are given by

$$S_B = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T \tag{1}$$

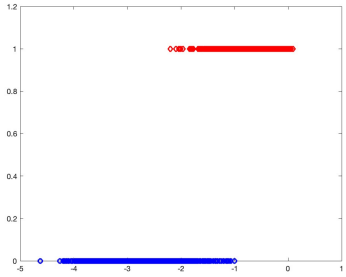
$$S_W = \sum_{j=1}^{2} \sum_{x} (x - \mu_j)(x - \mu_j)^T. \tag{2}$$

This means we may state the problem of finding our $\vec{w}$ vector as the following maximization problem

$$w = \underset{w}{\operatorname{argmax}} \frac{w^T S_B w}{w^T S_W w}. \tag{3}$$



(a) Singular Values of the Data



(b) Projections onto LDA Vector (blue=0 and red=1)

Figure 1: Singular Values and Projections for the Zeros and Ones

This maximization problem can then be solved by instead finding the solution to the generalized eigenvalue problem

$$S_B w = \lambda S_W w \tag{4}$$

where we are looking for the eigenvector $\vec{w}$ that corresponds to the maximum eigenvalue λ. This vector has the properties that we desire, so by projecting our data onto this vector we are able to have the best chance at differentiating the two categories of data from each other.

3 Results

Once we import the image data, we will reshape the data into one large matrix where each column of this matrix will be one of the images that has been unstacked into a column vector. We then take the wavelet transform of this matrix and then apply the singular value decomposition in order to find an appropriate low-rank approximation for our data. Figure 1a shows the plot of all of the singular values of our data. With the method we are using to distinguish the numbers, we can only compare two categories of data. This means we will isolate only the images with handwritten zeros and ones from our training data set to compare. We observe that the plot of the singular values has extremely heavy tails with a few extremely large values. To find the best low-rank estimate of our data, we will take the rank-15 approximation by inspection. Once we do this, we will then calculate the

linear discriminant analysis matrices that keep track of the between-class variance and the within-class variance in order to find our projection vector $\vec{w}$. After completing LDA, we will take the training data set and project the zeros and the ones onto this vector. Figure 1b shows the plot of the zeros (in blue) and the ones (in red) projected onto this vector $\vec{w}$. We see that this vector does a decent job at separating our data, and from this we may find our threshold value. After determining our threshold value, we can then do this entire process again to our test data to be able to tell whether our test image is a zero or a one. After this process is complete, we find that this algorithm is able to differentiate a handwritten zero and one with a 99.72% success rate.

When I gave my algorithm pictures of one of each digit from one to nine, the algorithm was able to distinguish my numbers reasonably well when I only compared it to one other number. However, once I compared it with the "one vs all" approach to every number it started to incorrectly identify a couple of the numbers. The accuracy that I was able to achieve with the single number vs single number approach was much higher than the accuracy with the one number vs all of the rest of the numbers approach.

4 Conclusion

In this report we were able to train a MATLAB computer program to recognize and label a handwritten number using machine learning. This was accomplished through the use of the wavelet transform, singular value decomposition, and linear discriminant analysis. We applied these processes to a set of training data to obtain a threshold value that was then used to sort out unlabeled test data. This program, when comparing just the numbers zero and one, was able to achieve a 99.72% success rate.

I was able to get my success rate for all 10 numbers to an average rate of over 95% with the "one vs all" approach. This success rate was calculated by comparing each number 0-9 to all of the other nine numbers, taking the success rate for each number, and then taking the average of all of the success rates. This process was done for different numbers of singular value approximations, and the plot in Figure 2 shows the singular values on the x-axis and the success rate on the y-axis. We can see that as we get better and better approximations that our success rate increases to a point before reaching a plateau at about 95.5%. I am not sure why my average success

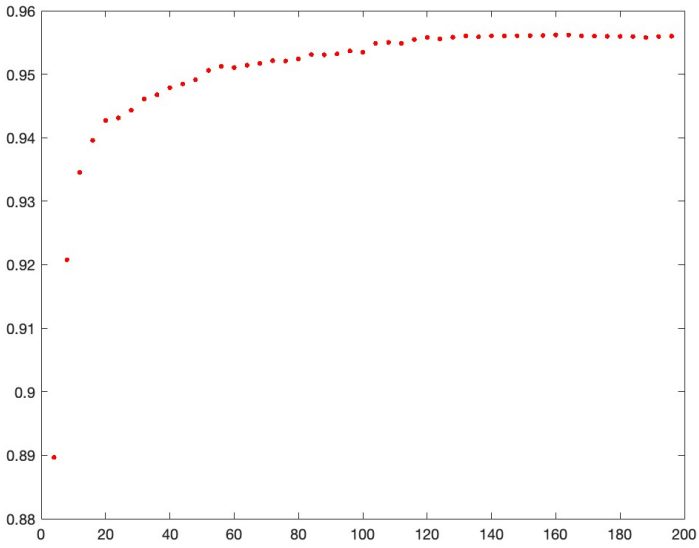


Figure 2: Plot of Singular Values vs Success Rate for "One vs All"

rate seems to be this high and I did not make many drastic changes to my code.