**UNIVERSITY OF VICTORIA**

**Department of Electrical and Computer Engineering**

**ECE 403/503 Optimization for Machine Learning**

**LABORATORY REPORT**

Experiment No: 1

Title: Handwritten Digits Recognition Using PCA

Date of Experiment: 28 May 2019

Report Submitted on: 4 June 2019

To: Jinlong Zhan - [jinlongzhan@uvic.ca](mailto:jinlongzhan@uvic.ca)

Laboratory Group No.: B01

Name(s):

Philippe Larocque - V00903595

Alexander Fung - V00904094

## Objectives

The objective of the lab is to identify handwritten digits from a large data set using the PCA method, extract the relevant information, and train a machine to recognize the handwritten data with. The performance and efficiency of the algorithm will then be tested and evaluated.[1]

## Introduction

In order to identify handwritten digits, we use a large training data set ‘x1600.mat’ which contains 1600 handwritten digits for each of 0 through 9, for a total of 16,000 handwritten digits. After applying the PCA method to the training data set, the machine should have the ability to identify handwritten digits to a high degree of accuracy.

MATLAB software was used to train the machine by taking a 28x28 pixel image and converting that into a column of data 784 digits long. Each pixel is characterize, and its features extracted. By using a testing data set ‘Te28.mat’, the algorithm was applied to the digit to categorize it into one of ten categories. The performance of the algorithm is then checked by comparing against the solution data set ‘Lte28.mat, and the performance time is also recorded.

## Results

4.3 - Evaluate the performance

The accuracy is: 95.94%

The error rate is: 4.06%, with 406 misclassifications

4.4 - Evaluate the efficiency

The time required for steps 3-6 is: 2.50 seconds

The PCA consumption time per 1000 is 0.250000

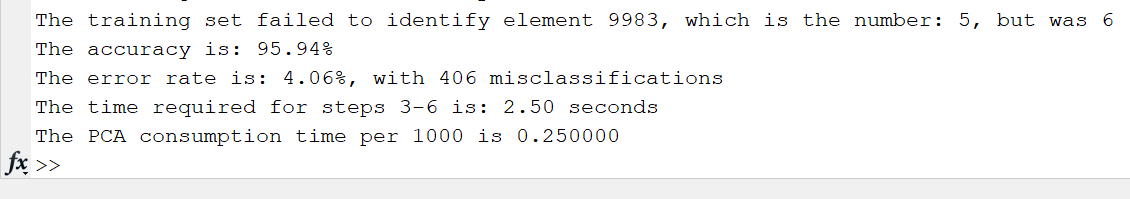


Figure 1 - Matlab console output of code from appendix A

## Discussion

While an accuracy of 95.94% is well above human accuracy, there are still some errors with the PCA method. As we can see in Appendix B, many digits are misclassified. For example, 3’s and 8’s are commonly misclassified. In particular, we also notice that digits written at an angle are also often misclassified.

## Conclusion

The objective of this experiment was to train a machine with a training set of handwritten digits, and apply that to a testing set. The performance and accuracy was found. By applying the PCA method to a training set of 10,000 handwritten digits, we were able to accurately identify 95.94% of the elements. The total number of misclassifications was 406.

The computer was able to process these in 2.50 seconds and the PCA consumption time per 1000 was 0.250000. For future testing, it is hypothesised that training the program with a larger sample set can help improve accuracy.

## References

[1] Lu, Wu-Sheng. (May 2019). Experiment 1 - Laboratory Manual ECE 403/504 Optimization for Machine Learning. [Online]. Accessed May 2019.

<https://ece.uvic.ca/~wslu/403/403pass/Trans/LabManual-ECE403-503-2019.pdf>

## Appendix A - MATLAB code

%\_\_\_\_\_\_\_\_Initialization\_\_\_\_\_\_\_\_%

clc

clear all

close all

%\_\_\_\_\_\_\_\_\_\_Experiment1\_\_\_\_\_\_\_\_\_\_%

load X1600.mat

load Te28.mat

load Lte28.mat

%\_\_\_\_\_\_\_\_\_\_Initialization\_\_\_\_\_\_\_\_\_\_%

q = 29; %Number of principal axes (As instructed in Lab Manual)

size = 1600; %number of samples for each set of digits

test\_size = 10000; %number of test samples

correct = 0; %Counter for matched img

error = 0;

%Separate all the testing data for each invidual number

%1600 columns are tests data for each number.

for i = 1:10 %itterate for each number 0 to 9

%Get data set for first number

X = X1600(:,((i-1)\*size+1):size\*i);

%Compute the mean of the first element of all 1600 set of the number.

%Transpose again to get 784x1 = a single 28x28 img of all 1600 numbers

%averaged together.

mu(:,:,i) = mean(X')';

%Data matrix.

A = X-mu(:,:,i);

%Compute Covariance matrix C = (1/m)A\*A\_transpose

C = (1/size)\*A\*(A');

[Uq(:,:,i), Sq] = eigs(C,q);

end

%Start Time

startTime = cputime;

%Comparing to testing set

for j=1:10

%Using equation in CH1 pg13, step3 f=Uq\_transpose\*(x-u), x=testSet

%Calculating Principal Component

f=Uq(:,:,j)' \* (Te28-mu(:,:,j)); %E1.4 in Lab Manual

x\_hat=Uq(:,:,j)\*f + mu(:,:,j); %E1.5 in Lab Manual

for k = 1:test\_size

e(k,j)=norm(Te28(:,k)-x\_hat(:,k));

end

end

%find minimum dist

[e\_min,index] = min(e');

result = (index')-1;

endTime = cputime; %End Time

b=1;

for a = 1:test\_size

%Compare result to Lte28.mat

if result(a) == Lte28(a)

correct = correct + 1;

else

error = error + 1;

fprintf('The training set failed to identify element %d, which is the number: %d, but was %d\n',a, Lte28(a),result(a));

end

end

totalTime = (endTime - startTime);

CPUTime = totalTime\*1000/test\_size;

correctPercent = (correct/test\_size)\*100;

errRate = (error/test\_size)\*100;

fprintf('The accuracy is: %.2f%%\n', correctPercent);

fprintf('The error rate is: %.2f%%, with %d misclassifications\n', errRate, error)

fprintf('The time required for steps 3-6 is: %.2f seconds\n',totalTime);

fprintf('The PCA consumption time per 1000 is %f\n', CPUTime)

## Appendix B

All 406 characters incorrectly identified as well as the algorithm’s guess (above). A zoomed-in sample is shown below.

