AMATH 582 Homework 4

Junzhao Liu

March 11, 2021

Abstract

The goal of this homework is to perform analysis of MNIST, include doing the SVD analysis of the digit images, singular value spectrum and image reconstruction. We also need to build linear classifier for both two digit and three digits to group the pictures.

1 Introduction and Overview

First Use a wavelet transform on each image to detect the edges and then find the principal components of the wavelet transforms to see how digits differ in the principal component basis. Use linear discriminant analysis to determine some threshold that separates different digits by using the train data, lastly test the algorithm use the test data.

1.1 Reshape each image into a column vector

The data we downloaded has the size of 60000*28*28, which means there are 60000 pictures, and each picture has a size of 28*28, we can use reshape function and a for loop to turn each picture to a column vector.

1.2 Edge detection

Convert the image to double precision, and start with a single-level discrete wavelet transform. cA is the approximation (low frequency content), cH is the horizontal details, cV the vertical details, and cDthe diagonal details. Plot the gaph and find the best one for tracing of the edges.

1.3 determine threshold

Linear Discriminant Analysis(LDA)is to find a suitable projection that maximizes the distance between the inter-class data while minimiz-ing the intra-class data.

2 Theoretical Background

As we learn in class in order to find the right subspace to project onto, first, we need to calculate the mean for each group for each feature.

Define the between class scatter matrix

$$\mathbf{S}_B = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T$$

within-class scatter matrix

$$\mathbf{S}_w = \sum_{j=1}^{2} \sum_{\mathbf{x}} (\mathbf{x} - \mu_j)(\mathbf{x} - \mu_j)^T$$

classify between more than two groups

$$\mathbf{S}_w = \sum_{j=1}^{N} \sum_{\mathbf{x}} (\mathbf{x} - \mu_j)(\mathbf{x} - \mu_j)^T \qquad \mathbf{S}_B = \sum_{j=1}^{N} (\mu_j - \mu)(\mu_j - \mu)^T$$

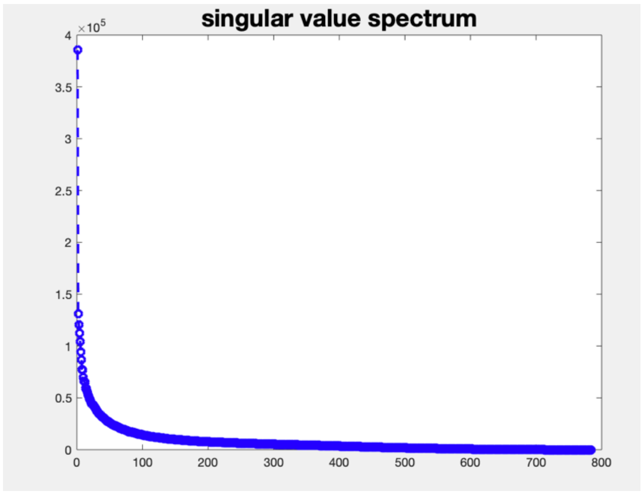
Find w to measure the variance within each group

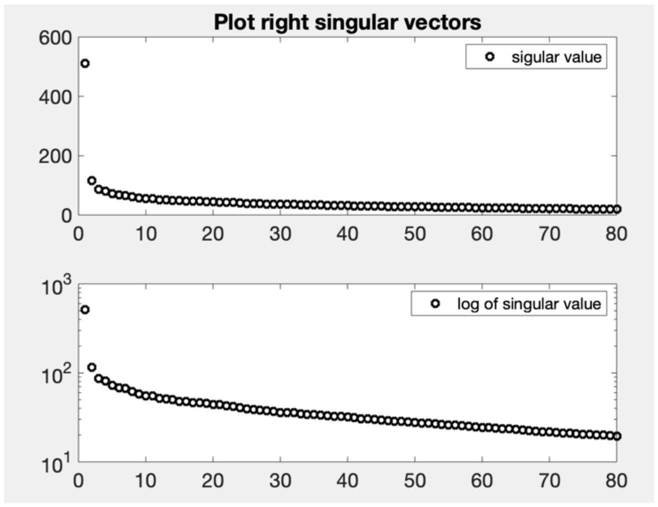
$$\mathbf{w} = \text{argmax} \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

3 Algorithm Implementation and Development

1. Use `function reshape to turn each image to a column vector`
2. Store columns that represent the same digit into one matrix
3. Turn the matrix to wavelets
4. Use function trainer to find the value for USV and threshold
5. use test data set to test the output above and find the accuracy

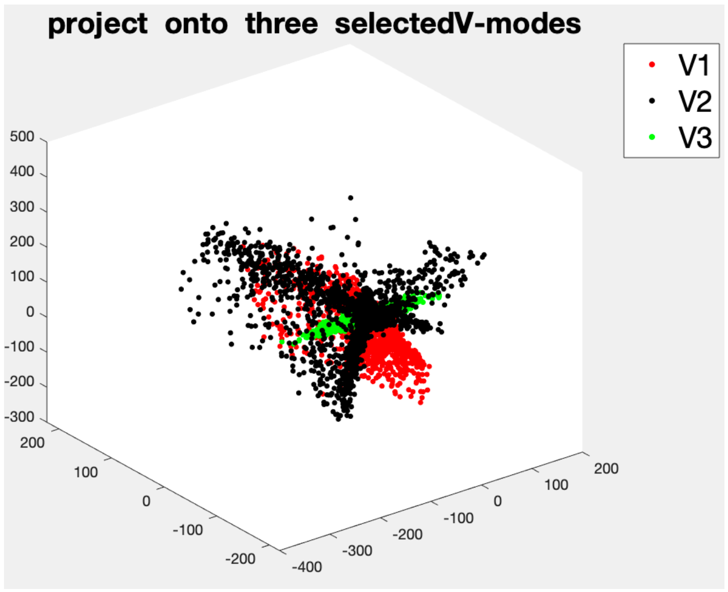
3 Computational Results





Rank three is necessary for good image reconstruction

V is a unitary matrix withv1andv2as its columns, U is unitary matrix with u1and u2 as its columns, and Σ is a diagonal matrix with σj on its diagonals.



one and six is the easiest to separate with accuracy of 0.9994
two and three is the hardest to separate with the accuracy of 0.7479

SVM can sperate between 10 digits for about 0.867 accuracy

LDA is doing better on both hardest pair and easiest pair.

Summary and Conclusions

I found LDA performs better for multiple group analysis than SVM support vector machines

Appendix A MATLAB Functions

[U,S,V] = svd(A) performs a singular value decomposition of matrix A, such that A = U*S*V'.

B = reshape(A,sz1,...,szN) reshapes A into a sz1-by-...-by-szN array where sz1,...,szN indicates the size of each dimension. You can specify a single dimension size of [] to have the dimension size automatically calculated, such that the number of elements in B matches the number of elements in A. For example, if A is a 10-by-10 matrix, then reshape(A,2,2,[]) reshapes the 100 elements of A into a 2-by-2-by-25 array.

Appendix B MATLAB code

```
clear all; clc
 [images, labels] = mnist_parse('train-images.idx3-ubyte', 'train-
labels.idx1-ubyte');
% ImageVector = reshape(permute(images,[3 1 2]),[],1);
 imagetrans = permute(images,[3 1 2]);
ImageVector = zeros(784,60000);
for i = 1:60000
current=imagetrans(i,:,:);
ImageVector(1:784,i) = reshape(current,784,1);
end
X=ImageVector(:,2);
Y=ImageVector(:,3);
Z=ImageVector(:,5);
[U,S,V] = svd(ImageVector,'econ');
sig = diag(S);
plot(sig,'b--o','Linewidth',2);
title('singular value spectrum','Fontsize',20)
V3 = [V(1,:)',V(2,:)',V(3,:)'];
Vimage3 = V3'*ImageVector;
V4 = [V(4,:)',V(5,:)',V(6,:)'];
Vimage4 = V4'*ImageVector;
V5 = [V(7,:)',V(8,:)',V(9,:)'];
Vimage5 = V5'*ImageVector;
 %plot3(Vimage[1],Y,Z,'k.','Markersize',10)
 plot3(Vimage3(1,:),Vimage3(2,:),Vimage3(3,:)','r.','MarkerSize',10)
 hold on
 plot3(Vimage4(1,:),Vimage4(2,:),Vimage4(3,:)','k.','MarkerSize',10)
 plot3(Vimage5(1,:),Vimage5(2,:),Vimage5(3,:)','g.','MarkerSize',10)
 %plot3(Xrank1(:,1),Xrank1(:,2),Xrank1(:,3),'r.','MarkerSize',10)
% axis vis3d
% hold on
% Xrank1 = S(1,1)*U(:,1)*V(:,1)';
% plot3(Xrank1(:,1),Xrank1(:,2),Xrank1(:,3),'r.','MarkerSize',10)
% Xrank2 = U(:,1)*S(1,1)*V(:,1)' + U(:,2)*S(2,2)*V(:,2)';
% plot3(Xrank2(:,1),Xrank2(:,2),Xrank2(:,3),'g.','Markersize',10)
title('project  onto  three  selectedV-modes','Fontsize',20)
 legend('V1','V2','V3','Fontsize',20)

pick two digit
digit5 = reshape(imagetrans(1,:,:),28,28);
```

```
imshow(digit5)
digit0 = reshape(imagetrans(2,:,:),28,28);
imshow(digit0)
digit4 = reshape(imagetrans(3,:,:),28,28);
imshow(digit4)
n=0;
digit5_wave = digit_wavelet(digit5_matrix);
digit0_wave = digit_wavelet(digit0_matrix);
digit4_wave = digit_wavelet(digit4_matrix);
feature=50;
%[U,S,V,threshold,w,sort5,sort0] =
dc_trainer(digit5_wave,digit0_wave,feature);


%viw first 6 graph five and zero overlap
[U,S,V] = svd([digit4_wave digit0_wave],'econ');
figure(1)
for k = 1:6
    subplot(2,3,k)
    ut1 = reshape(U(:,k),14,14);
    ut2 = rescale(ut1);
    imshow(ut2)
end

Plot right singular vectors
figure(2)
subplot(2,1,1)
plot(diag(S),'ko','Linewidth',2)
title('Plot right singular vectors','Fontsize',20)
set(gca,'Fontsize',16,'Xlim',[0 80])
legend('sigular value','Fontsize',13)
subplot(2,1,2)
semilogy(diag(S),'ko','Linewidth',2)
set(gca,'Fontsize',16,'Xlim',[0 80])
legend('log of singular value','Fontsize',13)

figure(3)
for k = 1:3
    subplot(3,2,2*k-1)
    plot(1:40,V(1:40,k),'ko-')
    subplot(3,2,2*k)
    plot(1:40,V(11807:11846,k),'ko-')
end
    subplot(3,2,1),
    set(gca,'Ylim',[-1 1],'Fontsize',12), title('digit5')
    subplot(3,2,2), set(gca,'Ylim',[-1 1],'Fontsize',12), title('digit0')
    subplot(3,2,3), set(gca,'Ylim',[-1 1],'Fontsize',12)
    subplot(3,2,4), set(gca,'Ylim',[-1 1],'Fontsize',12)
    subplot(3,2,5), set(gca,'Ylim',[-1 1],'Fontsize',12)
    subplot(3,2,6), set(gca,'Ylim',[-1 1],'Fontsize',12)

identified by the threshhold
figure(5)
subplot(1,2,1)
histogram(sort5,30);
hold on,
plot([threshold threshold], [0 10],'r')
```

```
set(gca,'Xlim',[-3 4],'Ylim',[0 10],'Fontsize',14)
title('digit5')
subplot(1,2,2)
histogram(sort0,30);
hold on, plot([threshold threshold], [0 10],'r')
set(gca,'Xlim',[-3 4],'Ylim',[0 10],'Fontsize',14)
title('digit0')

%test matrix
[test_images, test_labels] = mnist_parse('t10k-images.idx3-ubyte', 't10k-
labels.idx1-ubyte');
%take out 5 and zero matrix
testVector = zeros(784,10000);
 for i = 1:10000
 current=imagetrans(i,:,:);
 testVector(1:784,i) = reshape(current,784,1);
 end

n=0;
test0_matrix = zeros(784,1001);
for i = 1:10000
    if labels(i) == 9
        n = n+1;
        test0_matrix(:,n) = testVector(:,i);
    end
end

n0=0;n1=0;n2=0;n3=0;n4=0;n5=0;n6=0;n7=0;n8=0;n9=0;
test0_matrix = zeros(784,1001);
test1_matrix = zeros(784,1127);
test2_matrix = zeros(784,991);
test3_matrix = zeros(784,1023);
test4_matrix = zeros(784,980);
test5_matrix = zeros(784,863);
test6_matrix = zeros(784,1014);
test7_matrix = zeros(784,1070);
test8_matrix = zeros(784,944);
test9_matrix = zeros(784,978);
for i = 1:10000
    if labels(i) == 0
         n0=n0+1;
        test0_matrix(:,n0) = testVector(:,i);
    elseif labels(i) == 1
          n1=n1+1;
        test1_matrix(:,n1) = testVector(:,i);
    elseif labels(i) == 2
          n2=n2+1;
        test2_matrix(:,n2) = testVector(:,i);
    elseif labels(i) == 3
          n3=n3+1;
        test3_matrix(:,n3) = testVector(:,i);
    elseif labels(i) == 4
          n4=n4+1;
        test4_matrix(:,n4) = testVector(:,i);
    elseif labels(i) == 5
            n5=n5+1;
        test5_matrix(:,n5) = testVector(:,i);
    elseif labels(i) == 6
```

```
            n6=n6+1;
        test6_matrix(:,n6) = testVector(:,i);
    elseif labels(i) == 7
            n7=n7+1;
        test7_matrix(:,n7) = testVector(:,i);
    elseif labels(i) == 8
           n8=n8+1;
        test8_matrix(:,n8) = testVector(:,i);
    elseif labels(i) == 9
           n9=n9+1;
        test9_matrix(:,n9) = testVector(:,i);
    end
end

function [sucRate] = correction(test1_matrix,test2_matrix)
digit1_wave = dc_wavelet(test1_matrix);
digit2_wave = dc_wavelet(test2_matrix);
feature=50;
[U,S,V,threshold,w,sort5,sort0] =
dc_trainer(digit1_wave,digit2_wave,feature);
pval = [];
n_1=0;
n_2=0;
test1_2_matrix =[test1_matrix,test2_matrix];
TestNum = size(test1_2_matrix,2);
test1_2_matrix_wave = dc_wavelet(test1_2_matrix);
IMat = U'*test1_2_matrix_wave;
pval = w'*IMat;
total = size(test1_2_matrix_wave,2);
ResVec = (pval>threshold);
n1 = round(size(test1_matrix,2)/2);
n2 = size(ResVec,2)-n1;
hiddenlabels0 = zeros(1,n1);
hiddenlabels1 = ones(1,n2);
hiddenlabels = [hiddenlabels0,hiddenlabels1];
err = abs(ResVec - hiddenlabels);
errNum = sum(err);
sucRate = 1 - errNum/TestNum;
```