AMATH 482
Homework 5 Report
Fangzheng Sun
1239942

Abstract

In this project, I am going to deal with music in different genres and from different musician and explore a method to distinguish between them.

Introduction

In each case, I am supposed to load different pieces of music and distinguish them using the method of SVD and Linear Discrimination Analysis (LDA). Test one is to distinguish 3 songs in three different genres and test 2 is to distinguish 3 songs in the same genre, but from different bands. And test 3 is to distinguish between 3 genres, each with 3 different songs.

Theoretical background

1. Gabor Filter. The simplest Gabor window to implement is a Gaussian time-filter centered at some time τ with width a. The equation is:

$$g(t) = e^{-a(t-b)^2}$$

   In this time-frequency analysis, I create a "t-slide" to make sure the filter go over the whole piece of music.

2. Spectrogram. A spectrogram is a visual representation of the spectrum of frequencies in a sound or other signal as they vary with time or some other variable. In the analysis, spectrogram is produced after the filtering.

3. Singular Value Decomposition (SVD): SVD is a factorization of matrix into a number of

constitutive components all of which have a specific meaning in applications. For a matrix A ($m \times n$),

$$A = U\Sigma V^*$$

The SVD decomposition of the matrix A thus shows that the matrix first applies a unitary transformation preserving the unit sphere via $V*$. This is followed by a stretching operation that creates an ellipse with principal semiaxes given by the matrix Σ. Finally, the generated hyperellipse is rotated by the unitary transformation U.

4. Linear Discrimination Analysis (LDA): Based on machine learning, we can divide samples to "training samples" and "test samples" and computer will learn to distinguish between samples and dived them to categories.

Algorithm Implementation and Development

Test 1:

In this test, I selected the following 3 pieces of music: *Black Hole Sun* by Sound Garden, *Pathetic* by Beethoven and *Beat It* by Michael Jackson. They are mp3 files. Before using them, I checked that their bit rates were all 44100, and their lengths were all above 4:30 so that I can get enough samples. Since they have two channels, I use the first one. Besides, since my computer cannot run successfully with the whole pieces of music in this assignment, I must decimated them by 100 so that the sampling rates were 1/100 of the original, but the results were hardly affected by that. Every 5 seconds music was one sample, and I extracted 50 samples in each song, totally 250 seconds. Because most of songs had some blank time at the beginning, I cut the first 10 seconds so that my samples were 5-second parts from 11s to 260s. For each sample, I applied gabor filter and fft the filtered signals, storing the data to the spectrogram matrix, and added up all the to a sum matrix. In SVD process, I put the three sum matrices to a new one with 150 columns which represent total 150 samples, 50 for each song respectively. Then I applied economy SVD to the matrix and got U, S, V matrices. Finally I extracted date from 2-4 rows in V and divide them to 3 genres which represented the 3 songs and executed machine learning method (randomly selecting 50 samples, 30 for training and 20 for testing), plotting the bar graph.

Test 2:

This test has the same algorithm and similar codes to the first test. The three songs are in the same genre but from three bands. *Black Hole Sun* by Sound Garden, *Man In The Box* by Alice In Chains and *Nothingman* by Pearl Jam. All else are the same as the 1st part.

Test 3:

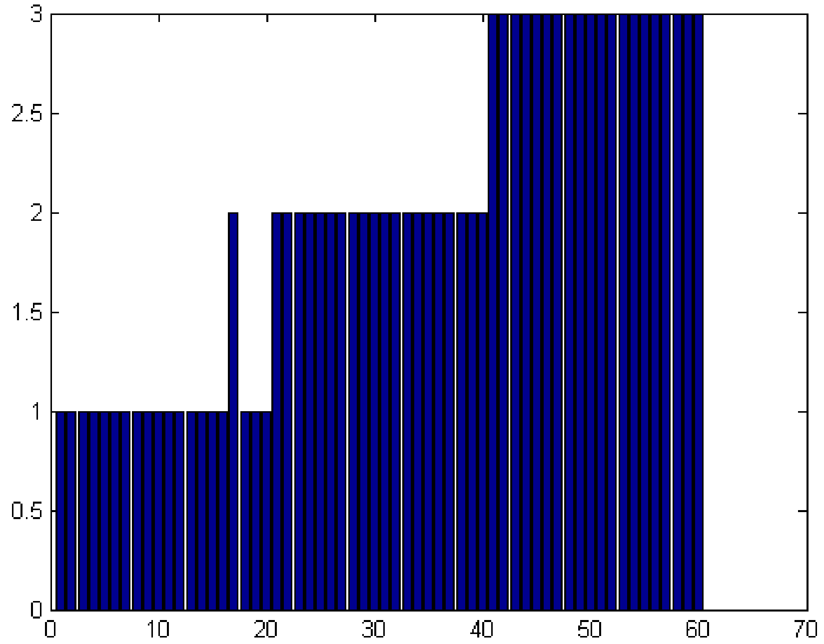
In this part I am going to distinguish between different genres instead of individual songs. For each genres among jazz, classical and rock music, I selected 3 pieces of music, as the following,

Genre	Piece 1	Piece 2	Piece 3
Jazz	Billie Holiday - Lover Man	Georgia on my Mind- Ray Charles	Count Basie - April In Paris
Classical	Beethoven - Pathetic	Chopin - Nocturne	Beethoven - symphony 9 Movement2
Rock	AC-DC - Thunderstruck	AC-DC - Whole Lotta Rosie	AC-DC - For Those About To Rock

In each of the 9 songs, I chose 20 5-second samples, from 11s to 110s, putting them together in a 180-column matrix and doing SVD. For LDA, I mixed every 60 columns randomly to 40 trainings and 20 tests which represents one genre. Finally I plotted the bar graph.

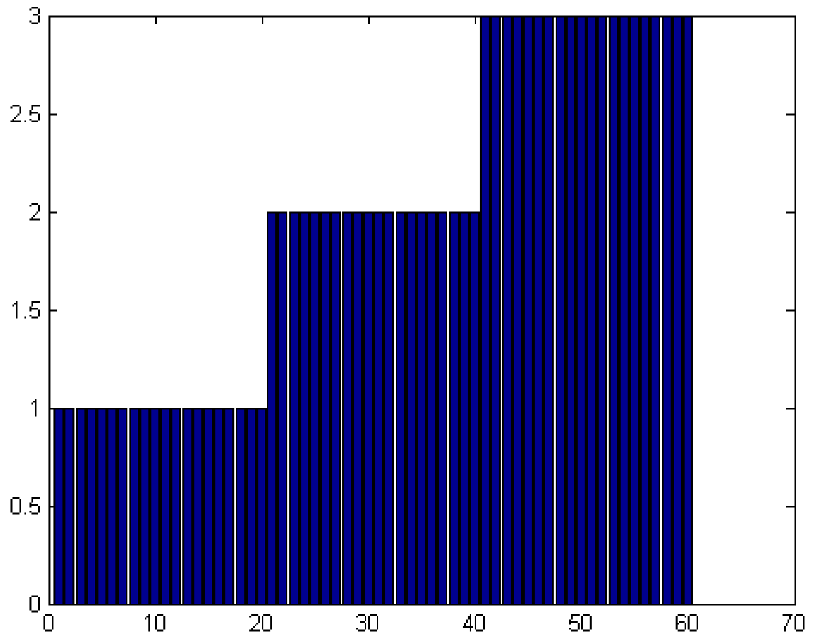
Computational Results

For test 1, the final graph was pretty good, after several trails, I got this:

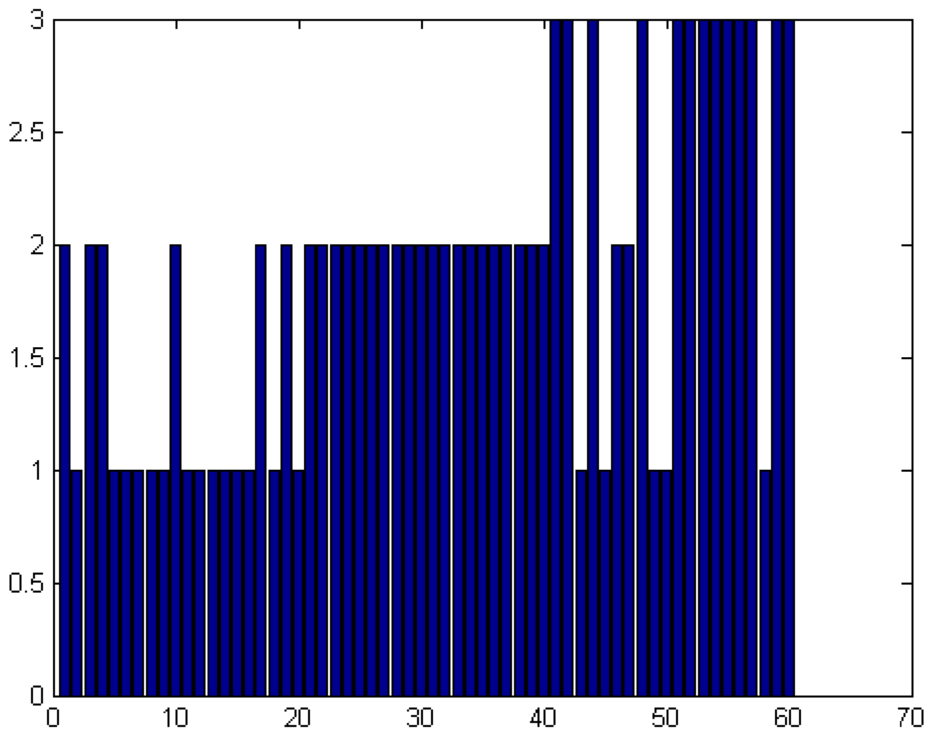


In the 60 tests, only one failed. The error rate is 1/60.

For test 2, the results were also perfect, and I can get 0 error rate.



For test 3, the results are not as perfect as those from the first 2 tests. In the picture, the error rate is 13/60. From the chart, it is clear that computer had difficulty distinguishing between jazz and rock, while successfully distinguishing the classical music. This might due to the similarity between that two genres of songs or because the pieces of song I select were not representative to their genres so that machine learning did not go well.



Summary and conclusions

Based on the analysis above, although the accuracy in the test 3 was not very high, we can conclude that LDA is a good method to distinguish among different music, training with some of their samples and testing the remaining.

Appendix A: MATLAB functions

linspace – Generate linearly spaced vector

fft – Fast Fourier Transform

fftshift – Shift Zero frequency component to center of spectrum

audioread – load music file

decimate – decrease sampling rate

randperm – random permutation

classify – discriminant analysis

bar – bar graph

Appendix B: MATLAB codes

Test 1:

```
clear all; close all; clc;

[black, Fs]=audioread('Soundgarden - Black Hole Sun.mp3');
[path, Fs]=audioread('pathetic Beethoven.mp3');
[beat, Fs]=audioread('Michael Jackson - Beat It.mp3');

%decimate the songs because the original version is too large and
%my computer can't run the whole code. And I select the first channel
%of the audio
black=decimate(black(1:Fs*260,1),100);
path=decimate(path(1:Fs*260,1),100);
beat=decimate(beat(1:Fs*260,1),100);

% spectrogram
n=5*Fs/100;
t2=linspace(0,5,n+1); t=t2(1:n);
k=(2*pi/5)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);
tslide=0:0.5:5;

black_total=[]; %build empty spectrogram matrix
```

```
path_total=[];
beat_total=[];
for j=3:52 %5-second/sample, 50 samples for each song, start from 11s
    black_spec=[];%spectrogram for each sample
    path_spec=[];
    beat_spec=[];
    black_sample=black(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    path_sample=path(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    beat_sample=beat(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    for j2=1:length(tslide)
        g=exp(-10*(t-tslide(j2)).^2); %gabor filter
        black_vg=fft(g.*black_sample);
        black_spec=[black_spec; abs(fftshift(black_vg))];
        path_vg=fft(g.*path_sample);
        path_spec=[path_spec; abs(fftshift(path_vg))];
        beat_vg=fft(g.*beat_sample);
        beat_spec=[beat_spec; abs(fftshift(beat_vg))];
    end
    %put all spactrograms into one matrix for each song
    col_size=size(black_spec,1)*size(black_spec,2);
    black_spec=reshape(black_spec,[col_size,1]);
    black_total=[black_total black_spec];
    col_size=size(path_spec,1)*size(path_spec,2);
    path_spec=reshape(path_spec,[col_size,1]);
    path_total=[path_total path_spec];
    col_size=size(beat_spec,1)*size(beat_spec,2);
    beat_spec=reshape(beat_spec,[col_size,1]);
    beat_total=[beat_total beat_spec];
end

% SVD
X=[black_total path_total beat_total];
[U,S,V] = svd(X,'econ');
Y=U'*X;

%% LDA
q1=randperm(50);
q2=randperm(50);
q3=randperm(50);
xblack=V(1:50,2:4);
xpath=V(51:100,2:4);
xbeat=V(101:150,2:4);
xtrain=[xblack(q1(1:30),:); xpath(q2(1:30),:); xbeat(q3(1:30),:)];
xtest=[xblack(q1(31:end),:); xpath(q2(31:end),:);
```

```
xbeat(q3(31:end),:)];
ctrain=[ones(30,1);2*ones(30,1);3*ones(30,1)];
pre=classify(xtest,xtrain,ctrain);
bar(pre);
```

Test 2:

```
clear all; close all; clc;

[black, Fs]=audioread('Soundgarden - Black Hole Sun.mp3');
[alice, Fs]=audioread('Alice In Chains - Man In The Box.mp3');
[pearl, Fs]=audioread('Pearl Jam - Nothingman.mp3');

%decimate the songs because the original version is too large and
%my computer can't run the whole code. And I select the first channel
%of the audio
black=decimate(black(1:Fs*260,1),100);
alice=decimate(alice(1:Fs*260,1),100);
pearl=decimate(pearl(1:Fs*260,1),100);

% spectrogram
n=5*Fs/100;
t2=linspace(0,5,n+1); t=t2(1:n);
k=(2*pi/5)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);
tslide=0:0.5:5;

black_total=[]; %build empty spectrogram matrix
alice_total=[];
pearl_total=[];
for j=3:52 %5-second/sample, 50 samples for each song, start from 11s
    black_spec=[];%spectrogram for each sample
    alice_spec=[];
    pearl_spec=[];
    black_sample=black(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    alice_sample=alice(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    pearl_sample=pearl(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    for j2=1:length(tslide)
        g=exp(-10*(t-tslide(j2)).^2); %gabor filter
        black_vg=fft(g.*black_sample);
        black_spec=[black_spec; abs(fftshift(black_vg))];
        alice_vg=fft(g.*alice_sample);
        alice_spec=[alice_spec; abs(fftshift(alice_vg))];
        pearl_vg=fft(g.*pearl_sample);
        pearl_spec=[pearl_spec; abs(fftshift(pearl_vg))];
```

```
    end
    %put all spactrograms into one matrix for each song
    col_size=size(black_spec,1)*size(black_spec,2);
    black_spec=reshape(black_spec,[col_size,1]);
    black_total=[black_total black_spec];
    col_size=size(alice_spec,1)*size(alice_spec,2);
    alice_spec=reshape(alice_spec,[col_size,1]);
    alice_total=[alice_total alice_spec];
    col_size=size(pearl_spec,1)*size(pearl_spec,2);
    pearl_spec=reshape(pearl_spec,[col_size,1]);
    pearl_total=[pearl_total pearl_spec];
end

% SVD
X=[black_total alice_total pearl_total];
[U,S,V] = svd(X,'econ');
Y=U'*X;

%% LDA
q1=randperm(50);
q2=randperm(50);
q3=randperm(50);
xblack=V(1:50,2:4);
xalice=V(51:100,2:4);
xpearl=V(101:150,2:4);
xtrain=[xblack(q1(1:30),:); xalice(q2(1:30),:); xpearl(q3(1:30),:)];
xtest=[xblack(q1(31:end),:); xalice(q2(31:end),:);
xpearl(q3(31:end),:)];
ctrain=[ones(30,1);2*ones(30,1);3*ones(30,1)];
pre=classify(xtest,xtrain,ctrain);
bar(pre);
```

Test 3:

```
clear all; close all; clc;

%jazz
[jazz1, Fs]=audioread('Billie Holiday - Lover Man.mp3');
[jazz2, Fs2]=audioread('Georgia on my Mind- Ray Charles.mp3');
[jazz3, Fs3]=audioread('Count Basie - April In Paris.mp3');
jazz1=decimate(jazz1(1:Fs*110,1),100);
jazz2=decimate(jazz2(1:Fs*110,1),100);
jazz3=decimate(jazz3(1:Fs*110,1),100);
```

```
%classical
[classical1, Fs4]=audioread('pathetic Beethoven.mp3');
[classical2, Fs5]=audioread('Chopin - Nocturne.mp3');
[classical3, Fs6]=audioread('Beethoven symphony 9 Movement2.mp3');
classical1=decimate(classical1(1:Fs*110,1),100);
classical2=decimate(classical2(1:Fs*110,1),100);
classical3=decimate(classical3(1:Fs*110,1),100);

%rock
[rock1, Fs7]=audioread('AC-DC - Thunderstruck.mp3');
[rock2, Fs8]=audioread('AC-DC - Whole Lotta Rosie.mp3');
[rock3, Fs9]=audioread('AC-DC - For Those About To Rock.mp3');
rock1=decimate(rock1(1:Fs*110,1),100);
rock2=decimate(rock2(1:Fs*110,1),100);
rock3=decimate(rock3(1:Fs*110,1),100);
%%
% spectrogram
n=5*Fs/100;
t2=linspace(0,5,n+1); t=t2(1:n);
k=(2*pi/5)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);
tslide=0:0.5:5;

jazz1_total=[]; %build empty spectrogram matrix
jazz2_total=[];
jazz3_total=[];
classical1_total=[];
classical2_total=[];
classical3_total=[];
rock1_total=[];
rock2_total=[];
rock3_total=[];
for j=3:22 %5-second/sample, 20 samples for each song, start from 11s
    %for jazz
    jazz1_spec=[];%spectrogram for each sample
    jazz2_spec=[];
    jazz3_spec=[];
    jazz1_sample=jazz1(5*(j-1)*Fs/100+1:+5*j*Fs/100)';
    jazz2_sample=jazz2(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    jazz3_sample=jazz3(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    for j2=1:length(tslide)
        g=exp(-10*(t-tslide(j2)).^2); %gabor filter
        jazz1_vg=fft(g.*jazz1_sample);
        jazz1_spec=[jazz1_spec; abs(fftshift(jazz1_vg))];
        jazz2_vg=fft(g.*jazz2_sample);
```

```
        jazz2_spec=[jazz2_spec; abs(fftshift(jazz2_vg))];
        jazz3_vg=fft(g.*jazz3_sample);
        jazz3_spec=[jazz3_spec; abs(fftshift(jazz3_vg))];
    end
    %put all spactrograms into one matrix for each song
    col_size=size(jazz1_spec,1)*size(jazz1_spec,2);
    jazz1_spec=reshape(jazz1_spec,[col_size,1]);
    jazz1_total=[jazz1_total jazz1_spec];
    col_size=size(jazz2_spec,1)*size(jazz2_spec,2);
    jazz2_spec=reshape(jazz2_spec,[col_size,1]);
    jazz2_total=[jazz2_total jazz2_spec];
    col_size=size(jazz3_spec,1)*size(jazz3_spec,2);
    jazz3_spec=reshape(jazz3_spec,[col_size,1]);
    jazz3_total=[jazz3_total jazz3_spec];
    %for classical
    classical1_spec=[];
    classical2_spec=[];
    classical3_spec=[];
    classical1_sample=classical1(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    classical2_sample=classical2(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    classical3_sample=classical3(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    for j2=1:length(tslide)
        g=exp(-10*(t-tslide(j2)).^2);
        classical1_vg=fft(g.*classical1_sample);
        classical1_spec=[classical1_spec;
abs(fftshift(classical1_vg))];
        classical2_vg=fft(g.*classical2_sample);
        classical2_spec=[classical2_spec;
abs(fftshift(classical2_vg))];
        classical3_vg=fft(g.*classical3_sample);
        classical3_spec=[classical3_spec;
abs(fftshift(classical3_vg))];
    end
    col_size=size(classical1_spec,1)*size(classical1_spec,2);
    classical1_spec=reshape(classical1_spec,[col_size,1]);
    classical1_total=[classical1_total classical1_spec];
    col_size=size(classical2_spec,1)*size(classical2_spec,2);
    classical2_spec=reshape(classical2_spec,[col_size,1]);
    classical2_total=[classical2_total classical2_spec];
    col_size=size(classical3_spec,1)*size(classical3_spec,2);
    classical3_spec=reshape(classical3_spec,[col_size,1]);
    classical3_total=[classical3_total classical3_spec];
    %for rock
    rock1_spec=[];
```

```
    rock2_spec=[];
    rock3_spec=[];
    rock1_sample=rock1(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    rock2_sample=rock2(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    rock3_sample=rock3(5*(j-1)*Fs/100+1:5*j*Fs/100)';
    for j2=1:length(tslide)
        g=exp(-10*(t-tslide(j2)).^2);
        rock1_vg=fft(g.*rock1_sample);
        rock1_spec=[rock1_spec; abs(fftshift(rock1_vg))];
        rock2_vg=fft(g.*rock2_sample);
        rock2_spec=[rock2_spec; abs(fftshift(rock2_vg))];
        rock3_vg=fft(g.*rock3_sample);
        rock3_spec=[rock3_spec; abs(fftshift(rock3_vg))];
    end
    col_size=size(rock1_spec,1)*size(rock1_spec,2);
    rock1_spec=reshape(rock1_spec,[col_size,1]);
    rock1_total=[rock1_total rock1_spec];
    col_size=size(rock2_spec,1)*size(rock2_spec,2);
    rock2_spec=reshape(rock2_spec,[col_size,1]);
    rock2_total=[rock2_total rock2_spec];
    col_size=size(rock3_spec,1)*size(rock3_spec,2);
    rock3_spec=reshape(rock3_spec,[col_size,1]);
    rock3_total=[rock3_total rock3_spec];
end
jazz_total=[jazz1_total jazz2_total jazz3_total];
classical_total=[classical1_total classical2_total classical3_total];
rock_total=[rock1_total rock2_total rock3_total];

% SVD
X=[jazz_total classical_total rock_total];
[U,S,V] = svd(X,'econ');
Y=U'*X;

%% LDA
q1=randperm(60);
q2=randperm(60);
q3=randperm(60);
xjazz=V(1:60,2:4);
xclassical=V(61:120,2:4);
xrock=V(121:180,2:4);
xtrain=[xjazz(q1(1:40),:); xclassical(q2(1:40),:);
xrock(q3(1:40),:)];
xtest=[xjazz(q1(41:end),:); xclassical(q2(41:end),:);
xrock(q3(41:end),:)];
```

```
ctrain=[ones(40,1);2*ones(40,1);3*ones(40,1)];
pre=classify(xtest,xtrain,ctrain);
bar(pre);
```