Report For Patten Analysis

Username: jg495
Name: Jun Guan

Abstract

There are series of classifications exist. I only chose the way that is learned from the lecture. Firstly, I am going to get a feature vector for each training image. Secondly, I will choose the Bayes law and unsupervised Gaussians Mixture Model to build classifier for different type of images, after that I will use the confusion matrix to test and discuss my classifier. During all the process, I mainly focus on the three different parts: how to build a successful classifier, what will affect the accuracy rate of my classifier and how to improve the performance of my classifier.

1.Introduction

Firstly, I am going to talk about what is classification. A classification is an ordered set of related categories used to group data according to its similarities. Our task is to build a classifier to classify binary shapes that is a white (or black) region in a picture of black and white pixels. There are many challenges for this task, such as how to get feature vectors, how to construct a classifier, how to train the classifier and how to test it.

2.Theoretical Background

2.1 Feature Vector

A feature is an amount of values use to describe an object. For this task, I need to write code to transfer binary image into the feature vectors into my classifier. I will work out the feature vectors by using chain-code and the Fourier transform.

2.1.1 Chain-code

The chain code is “a lossless compression algorithm for monochrome images.” The basic principle of the chain code is to use eight different numbers which from 0 to 8 to encode the boundary of the image. I could use the chain code to transfer the image to a series number such as: 0,2,1,4,2,5,7,5,3…

2.1.2 Fourier Transform

The Fourier Transform relies on the fact that all the functions could be rewritten by adding sinusoidal waves together. The Fourier transforms convert data in time (or space) into frequencies. Therefore, I will use Fourier Transform to transform the chain-code (will be in angle in this task) into frequencies. After that, I can distinguish the useful information and noisy because the useful signals are located at low frequencies and noises are at high frequencies of the data. Fourier transform could help me to improve the accuracy rate of my classifier.

2.2 Bayes’ Law

The Bayes’ Law is one of the most famous laws of calculating the probability. It gives the posterior probability.
The formula for the Bayes’ Law is:

$$p(\theta|\mathbf{x}) = \frac{p(\mathbf{x}|\theta)p(\theta)}{p(\mathbf{x})}$$

In this task,
θ is the image class, such as (face, alien).
x is the feature vector of the testing image.
p(θlx) is the probability that the test image belong to class θ.
p(xlθ) is the likelihood which is calculated by Multivariate Normal Distribution.
p(θ) is the prior of the class θ(according to MAP).
p(x) is Gaussian Mixture Models(GMM).

2.2.1 Multivariate Gaussian Distribution

The Multivariate Gaussian model is used for multidimensional continuous data. In this task, the formula of the Multivariate Gaussian Model is:

$$p(x|\mu, C) = \frac{1}{(2\pi)^{K/2}|C|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T C^{-1}(x-\mu)\right),$$

C is the covariance of the matrix that is consists of training images' feature vector.
x is the testing images' feature vector.
K is the number of dimension
u is the row's mean of the matrix which is consists of training image's feature vector.
Example to show what is the u:

$$\begin{matrix}1 & 1 & 1\\ 2 & 2 & 2\\ 3 & 3 & 3\end{matrix} \quad \xrightarrow{\text{u}} \quad \begin{matrix}1\\ 2\\ 3\end{matrix}$$

C is much more complicated to calculate. The formula to calculate C is:

$$C = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu)(x_i-\mu)^T.$$

n is the number of the sample points(number of training images).
X_i is the feature vector of each training image.
u is the same as above.
T means the transpose of it.
The 1/n-1 partial in the formula is to obtain the unbiased estimate.

2.2.2 Gaussian Mixture Models(GMM)

GMM is a common classifier used for multidimensional continuous data. Each Gaussian in the mixture is called a "“component” and the GMM estimates the density of all the data. The formula for the GMM is:

$$p(\mathbf{x}) = \sum_i p(\mathbf{x}|\theta_i)p(\theta_i)$$

p(xIθi) is the multidimensional Gaussian Model for image class θi.
p(θi) is the prior of the class. p(θi) = Ni/N (the number of shapes in that class divided by the total number of shapes).

The GMM have two different methods: Supervised GMM Fitting and Unsupervised GMM Fitting.

Supervised GMM:
The Supervised GMM mean that "a human some how guides the learning stage". In the supervised GMM I need to assume that all the sample points have already had a label attached to them by a human and the number of components N is determined by the number of distinct labels.

Unsupervised GMM:
The Unsupervised GMM means automatically determining the number of Gaussian components,and the parameters and priors for each one. But in Unsupervised GMM, I need to decide which points belong to which cluster. Additionally, I need to use the Expectation Maximisation(EM) algorithm to deal with the missing labels. These labels are not explicit in the measured data. The EM algorithm computes the missing labels by maximising the expectation. However there are 3 problems occurred while I was using EM with the GMM:
1: The data in any one class are not Gaussian distributed
2: The data clusters overlap.(This will affect the performance of my classifier)
3: The EM algorithm does not guarantee a global maximum

According to their different features, I decide to use the unsupervised GMM in my task.

2.2.3 The Maximum A-Posterior (MAP) Solution

This method is used to get the Maximum Likelihood. According to the MAP, I will add p(θ) to the express the prior belief. Hence in the Bayes' law, I multiple the p(θ) with the probability to get more accurate result.

2.3 Confusion Matrix

Confusion Matrix is a common way to test the classifier. The Confusion Matrix is using table to show the testing result.
For example:

	Star	Alien	Butterfly	Face
Star	10	2	0	0
Alien	2	11	0	0
Butterfly	0	0	12	2
Face	0	3	3	13

Each row shows the test result for each type of testing image. The diagonal of the table is the number of images have been correctly classified and all other values are the wrong testing results. Take the above table as an example. The accuracy rate of the classifier should be calculate as: all the right test value / all the test value = (10+11+12+13)/58 = 80%.
Hence, the accuracy rate for the sample classifier is 80%.

3.Method

3.1 Build and Train Classifier

3.1.1 - Get the feature vector of binary images

Step 1 - Loading the training binary shapes from the image folder.
Step 2 - Using the chain-code to encode the boundary of the training binary shapes.
Step 3 - transfer the chain code to the angle, by doing this, we can treat the chain code as the sinusoidal waves.
Step 4 - Fourier transform of the angle. This is because the image's chain code includes noisy that I do not want to put into my feature vector. All the noisy in the sinusoidal waves are shown as "high frequency", it will vary my sinusoid wave rapidly. After the Fourier transform, I could easily see the frequencies which are noisy and use low pass filtering to throw away these noises.
Step 5 - Build the low pass filtering and use it to keep the low frequency values of the angle which have been Fourier transformed. After passing though the filter, the main shape of the boundary will be smoother and it will give me more significant feature vectors of the shape. Doing in this way could help me to improve the accuracy rate of the classifier.
Step 6 - Absolute the value of filtered angle, by doing this, we could get rid of the effect of rotation shapes, and it could improve the performance of the classifier. And finally, I get the feature vector of the shape which contains mostly useful information for the classification with few redundancies.
The code is shown below:

```
%Step:1 -load image--------------------------
imageName = [name num2str(a,'%03d') '.gif'];
im = imread([imagedir imageName]);
im = logical(im); %Convert the original intensity values into logical 1s and 0s

%Step:2-chain-code--------------------------
c = chainCode(im);

%Step:3-chain-code to angle--------------------------
angles = c(3,:)*(2*pi/8) ;

%Step:4-Fourier Transform of angle--------------------------
anglesFFT = fft(angles);

%Step:5-Low pass filtering--------------------------
N = numOfLowestF; %number of lowest frequencies to keep -  filter
filter = zeros(size(angles));

filter(1:N) = 1;
filter(end-N+2:end) = 1;
filteredFFT = anglesFFT .* filter; % Apply the filter by scalar multipliacation

%Step:6-Absulut the value--------------------------
FV = abs(filteredFFT)/100;    % The feature vector
```

3.1.2 - Develop the classify

For each type of the shape, I build a classifier by using the feature vectors of this type of training images.

Step 0 - Get the feature vectors of this type of training images.(have done at 3.1.1)
Step 1 - Calculate the mean(u) matrix of feature vectors of the images.

Step 2 - Calculate the covariance (C) matrix of the feature vectors of the images.

```
%step:0 get the feature vectors of the image------------
training_FV = getFeatureVector(name,1,numOfImage,numOfLowestF);
%Step:1 get the mean of the feature vectors-------------
mu = mean(training_FV,2);
%step:2 Calculate the Covariance of the feature vectors--------------
U = bsxfun(@minus,training_FV,mu);
Cb = zeros(numOfLowestF,numOfLowestF);
for n = 1:numOfImage
  u = U(:,n);
  Cbi = u * u';
  Cb = Cbi+Cb;
end
C = Cb/(numOfImage - 1);
```

Step 3 - Use the mean and covariance to create a multivariate Gaussian distribution(P(xlθ)) for this type of images.
Step 4 - Calculate the unsupervised GMM by using the result from the Step 3(multivariate Gaussian distribution) and the prior of each type of images.
Step 5 - In order to get more accurate result, I applied MAP at this stage. I need to maximise the posterior probability. Hence,I calculated the prior(p(θ)) of this type of images and multiplied it with the likelihood.
Step 6 - To build a classifier by Byes' Law using all the above information(the probability of the data may be quickly approach to zero, so I logarithm of the likelihood to prevent the underflow error.):

$$p(\theta|\mathbf{x}) = \frac{p(\mathbf{x}|\theta)p(\theta)}{p(\mathbf{x})}$$

```
putIn = bsxfun(@minus,testFV,mu_class);
nFact = (2*pi)^(numOfFilter/2)*det(C_class)^(1/2);
p = log(exp(-0.5*(putIn)'*invC_class*(putIn))/nFact)+log(weight);
```

The p(x) is constant while we classify each testing image. So, If I do not count the p(x) into the Bayes' Law will not affect my classifier result. (Because I do not want the exactly probability value, I juts want to compare the probability which is calculated by the Byes' Law)

3.2 Test Classifier

Step 1 - Load testing images.
Step 2 - Get the feature vectors of the testing images.
Step 3 - To substitute the feature vectors into the Bayes' Law of each type of image.
Step 4 - After Step 3,I will get 4 probabilities and the shape is classified as the shape class θ, according to the maximum p(xlθ)p(θ).
Step 5 - Build a Confusion Matrix table for the testing result. The Confusion Matrix table should look like this:

	Star	Alien	Butterfly	Face
Star				
Alien				
Butterfly				
Face				

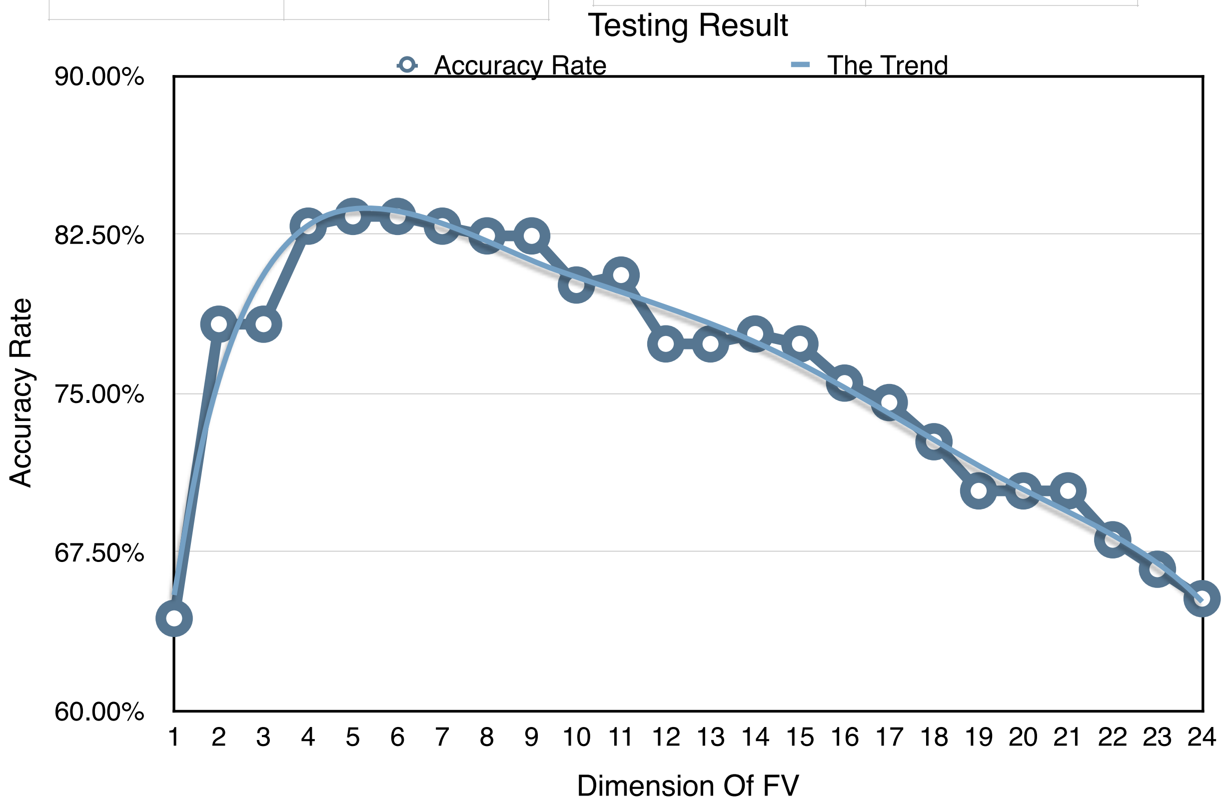
Step 6 - Analysis the Confusion Matrix table,calculate the accuracy rate of my classifier and analysis my classifier.
Step 7 - In order to improve the accuracy rate of my classifier.I need to find the best number of the dimension of feature vectors.Hence,I will change the number of the dimensions of feature vector and go back to Step 1 to repeat it.

4.Result And Discussion

4.1 - Result

The number of training image is equal to the number of testing image.

Dimension Of FV	Accuracy Rate(%)
1	64.35
2	78.24
3	78.24
4	82.87
5	83.33
6	83.33
7	82.87
8	82.41
9	82.41
10	80.09
11	80.56
12	77.31
13	77.31
14	77.78
15	77.31
16	75.46
17	74.54
18	72.69
19	70.37
20	70.37
21	70.37
22	68.06
23	66.67
24	65.28



The testing result is shown above:

I only used the range which from 1 to 24 of dimensions of feature vectors. The number of dimension of the feature vector cannot greater than 25, this is because once the number of dimension is greater than the number of sample points, the matrix will be singular. In my training images, the smallest number of sample points is 25.

The confusion matrix table for dimension of feature vector is 5:

	Star	Alien	Butterfly	Face
Star	20	4	1	0
Alien	0	40	2	0
Butterfly	0	1	49	0
Face	0	27	1	71

The confusion matrix table for dimension of feature vector is 6:

	Star	Alien	Butterfly	Face
Star	20	5	0	0
Alien	0	39	3	0
Butterfly	0	2	48	0
Face	0	26	0	73

4.2 - Discuss

4.2.1 - From the result table above, I can know that the accuracy rate of my classifier will reach to maximum(81.02%) when the dimension of the feature vector is 5 or 6.

4.2.2 - From the trend in the graphic table, we can see the accuracy rate is increasing as the dimension of feature vector from 2 to 5.The accuracy rate is decreasing as the dimension of feature vector from 6 to 24. Hence, I know that too many or too less dimension of feature vector may exert negative effect performance on my classifier. I think the reasons are:
1 - If the dimension of feature vector is too small, the classifier cannot get enough information from the feature vectors. My classifier will cannot be capable to classify the images based on the limited feature vectors.
2 - If the dimension of feature vector is too many, the feature vector will include a lot of noisy which would exert negative effect on my classifier.
So, 5 or 6 dimension of feature vector is the best for my classifier. 5 or 6 dimension of feature in my task could help classifier to catch enough useful information of shapes.

4.2.3 - Before the testing, I thought my classifier will not perform well when classify the rarer shapes because I was using MAP and the rarer shape’s prior will be very small. Nevertheless, the result shows my consideration can be ignored because the feature vectors of the rare images are quite different from the other images.

4.2.4 - From the confusion matrix table above and also other confusion matrix of different dimension of feature vector, I found that a lot of face images are classified as alien. I think the reason is as following:
Each of the face training image in the face class is spread quite broadly in feature space, however the class centres is quite close.To be more specific, the feature vector of the different face images

are significantly different, however there are no enough training image to help the classifier to find out the difference. Hence the classifier will find it very hard to tell face class from alien class. Currently,I know 2 standard way to measure the similarity of two different classes: “KL-divergence” and “Bhattcharyya distance”.

The detail of “KL-divergence” is shown below:

$$D_{\mathrm{KL}}(P\|Q) = \int_{-\infty}^{\infty} \ln\left(\frac{p(x)}{q(x)}\right) p(x)\,\mathrm{d}x,$$

This formula is used for distribution P and Q of a continuous random variable.
P and Q are two probability distributions.
p and q are the densities of P and D.
The DkL(PIIQ) is a measure of the information lost when Q is used to approximate P.

The detail of the “Bhattcharyya distance” is shown below:

$$D_B(p,q) = -\ln\left(BC(p,q)\right) \quad BC(p,q) = \int \sqrt{p(x)q(x)}\,dx$$

p and q are two different classes.
$D_{B(p,q)}$ is the Bhattcharyya distance between p and q classes.
BC is the Bhattcharyya coefficient.

4.Conclusion

After this project I learned how to build a classifier and testing the classifier. I am confident on getting the chain-code of a image and transfer it to the feature vector. I successfully built a classifier by using the feature vectors of training images. The classifier has a good performance when I used it to classify the testing images. Generally, my project is successful and achieve the project’s aim.

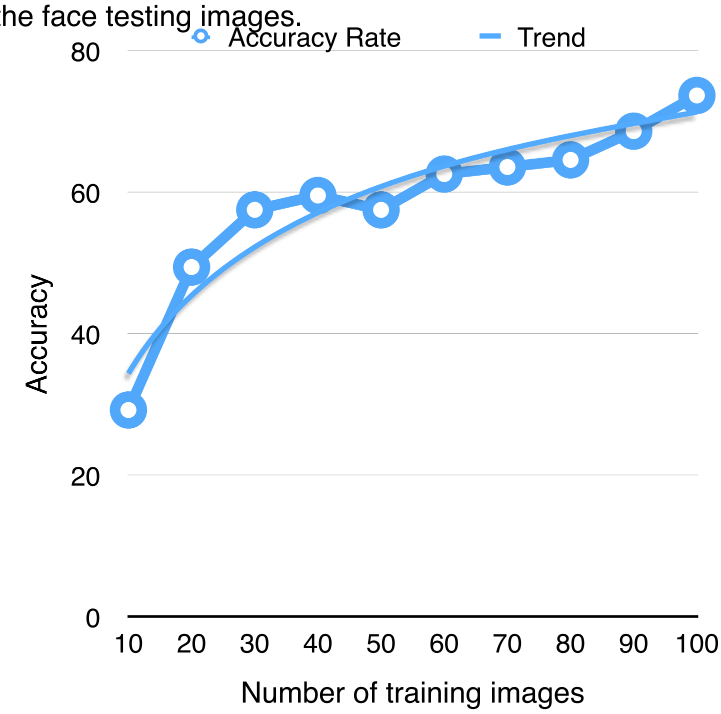
5.Further Work

During the project, I found the number of training images also could significant affect the performance of my classifier. Hence, I am going to find the relation between the accuracy rate and the number of training images.

I will only testing the face images and keep the number of the testing images as constant(100 face images as testing images).
I will use 6 dimensions of feature vector during all my further testing.

I will keep the number of other type's training images as constant.
Therefore, in my further testing, the only changing variable is the number of training images of face and the accuracy rate of classifying the face testing images.

Number of training images	Accuracy rate (%)
10	29.29
20	49.49
30	57.58
40	59.60
50	57.57
60	62.63
70	63.63
80	64.63
90	68.69
100	73.74



Result and Discuss:

From the result above, we can know that the accuracy rate of the classifier is increasing as the number of the training images increased. The trend is shown as logarithm function which mean at the beginning of the function, a slightly increase on the number of training images could lead to a dramatic increment on the accuracy rate of classifier.

Hence, If I use more training images to training the classifier, my classifier will have a better performance on classifying images. However, the training images which are given by teacher is limit, so I cannot to increase the number of training images. But generally speaking, I believe the 83.33% accuracy rate is good enough for my classifier to classify testing images.

6.Biblography

Lecture Notes
Tutorial group's material
http://www.cso.ie/en/surveysandmethodology/classifications/whatisaclassification/
http://en.wikipedia.org/wiki/Kullback–Leibler_divergence