Abstract

Firstly I will begin by talking about the theory behind a classifier. This includes explaining what a classifier is, what it does and how it does it. I will also talk about chain codes and fourier transforms. Then I will explain how I undertook creating my classifier. This will be a step by step going through my code showing examples and explaining what each snippet does. Finally I will present my results along with how I tested and optimised the classifier.

Theory

A classifier is a program that takes an input and outputs a class label. In this project that means taking in an image and the output being the type of image it is.

Chain codes are an important part of a classifier . The chain code of an image is a way to represent a binary object but encoding just the boundary. Chain code is made up of numbers between 1 and 7 relating to the transition between two consecutive pixels on the boundary. Chain code only contains the object's shape, the position of the object is lost. (Hendricks, 2010).

Fourier transforms are also crucial to a classifier. A fourier transform transforms a signal into the frequencies that make it up. This is so important as the chain code will be noisy with variations that need to be removed. This can be fixed by removing the variations that are high frequency. Fourier transforms also allows chain codes of different lengths to be compared.
Then we have a feature vector, which is simply a vector that contains information that can describe the important characteristics of an object.

Method

The first thing I did before I started was to split the images classes in half so there for each type of image half were in a training set and half were in a testing set.

I then created create a matrix, 'output', that has the dimensions such that the number of rows corresponds to the number of images in the training set and the number of columns such that it corresponds to the frequencies to keep.

Then a for loop goes through every image in the training set, calculates the chain code, performs a fourier transform and gets a feature vector. That feature vector is then added to the output matrix. Once all the images have been looped over, the output matrix will contain contain one row per image that holds that image's feature vector.

```
frequenciesToKeep = 7;
output = zeros(42,frequenciesToKeep);
for a = 1:42;
    im = imread(sprintf('Images/AlienTraining/alien%1d.gif', a));
    im = logical(im); %Convert the original intensity values into logical 1s and 0s
    c = chainCode(im);
    %% filter using the FT of the angles of the chaincode
    angles = c(3,:)*(2*pi/8);
    anglesFFT = fft(angles); %fast fourier transform
    filteredFFT = anglesFFT(1:frequenciesToKeep); % Apply the filter by scalar
multipliacation
    % feature vector
    output(a,:) = abs(filteredFFT)/100;
end
```

I then take the mean and covariance of the output matrix. The mean matrix needs to be transposed for later use.

```
themean = mymean(output);
thecovariance = cov(output,1);
transposedMean = (themean)';
```

I then run the chain code and fourier transform again on the test image to get a feature vector for it. This is also transposed.

To classify the image I used the formula for maximum a posteriori estimation:

$$\arg\max \theta\ p(\theta|x) = \ln(p(\theta)) - 0.5 \ln(\det C) - 0.5 (x - \mu) >C-1 (x - \mu)$$

```
map = log(42/227) - 0.5*log(det(thecovariance)) -
0.5*((transposedTraining-transposedMean)'*(thecovariance^-1)*(transposedTraining-transposedMean));
```

Where:

- ln(p(θ)) is the log of the posterior probability.
- det C is the determinant of the covariance matrix.
- x is the feature vector of the test image

- μ is the mean of the output matrix

The result of this formula is the likelihood of the tester image belonging to the class of training images where the greater the number the greater the likelihood.

I have 4 of these classes, one for each Alien, Butterfly, Face and Star.

To actually carry out the classification of all the test images I created a new class called main.

I first create a 4 x 4 confusion matrix filled with zeros. Then I loop through every image in the test folder and call the Alien, Butterfly, Face and Star class with the test image. This returns 4 numbers which are all stored in variables.

```
confusionMatrix = zeros(4);
imageDirectory = ('Images/OtherTest');
filePattern = fullfile(imageDirectory, '*.gif');
files = dir(filePattern);
for k = 1:length(files)
    baseFileName = files(k).name;
    fullFileName = fullfile(imageDirectory, baseFileName);
    tester = fullFileName;

    theAlien = alien(tester);
    theButterfly = butterfly(tester);
    theFace = face(tester);
    theStar = star(tester);
```

Whichever class then returns the greatest number is what the test image is classified as.

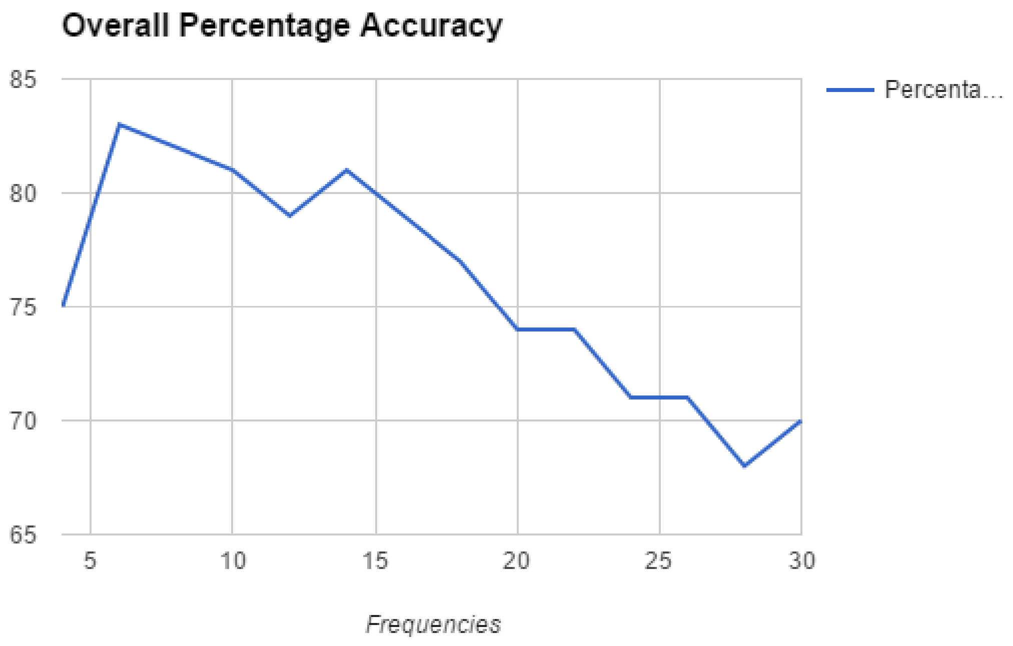
I then increment the corresponding value in my confusion matrix depending on what the image actually was and what the classifier thought it was. The way I am checking is by comparing the file names. In addition to this I have printed the number of correct and incorrect classifications.

Results

The first test of the classifier was using 30 frequencies. This is the confusion matrix produced from this test:

	Alien	Butterfly	Face	Star
Alien	**37**	5	0	0
Butterfly	4	**46**	0	0
Face	28	12	**59**	0
Star	12	2	0	**6**

In order to optimise my classifier I tested the frequencies beginning at 4 and increasing by 2 up to 30. I put the data from all of those different frequencies in to their own confusion matrices and plotted graphs to see which frequencies performed best. From this I aimed to find the optimal frequency for the classifier as well as seeing what the optimal frequency was for classifying certain images.

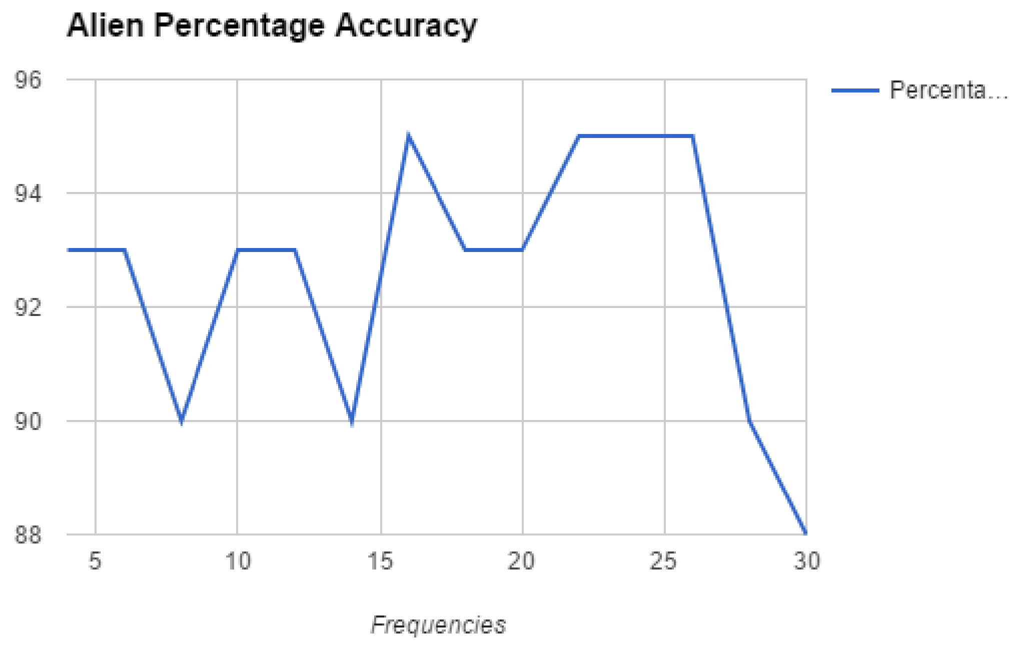


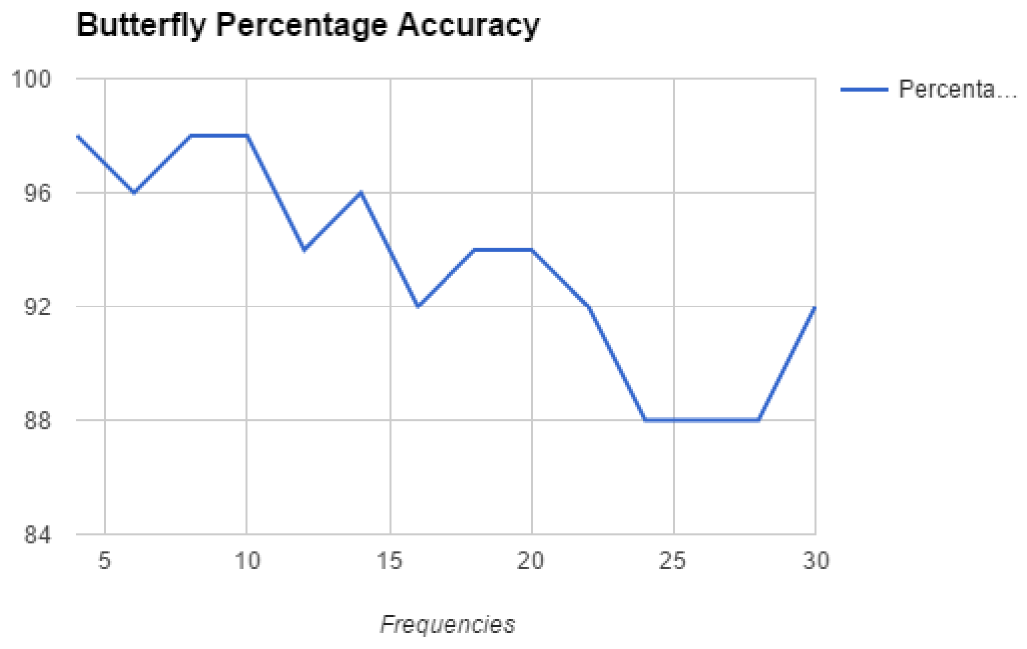
As the graph shows, the optimal number of frequencies for classification is 6 which produced a result of 83% of images being correctly classified.

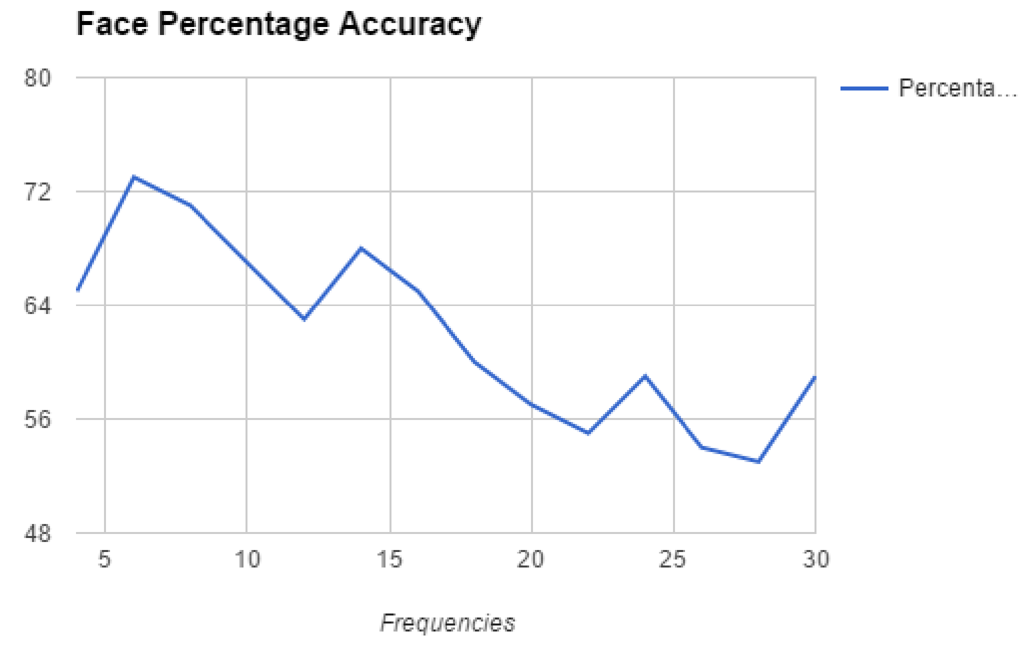
	Alien	Butterfly	Face	Star
Alien	**39**	3	0	0
Butterfly	2	**48**	0	0
Face	25	0	**73**	1
Star	2	2	0	**16**

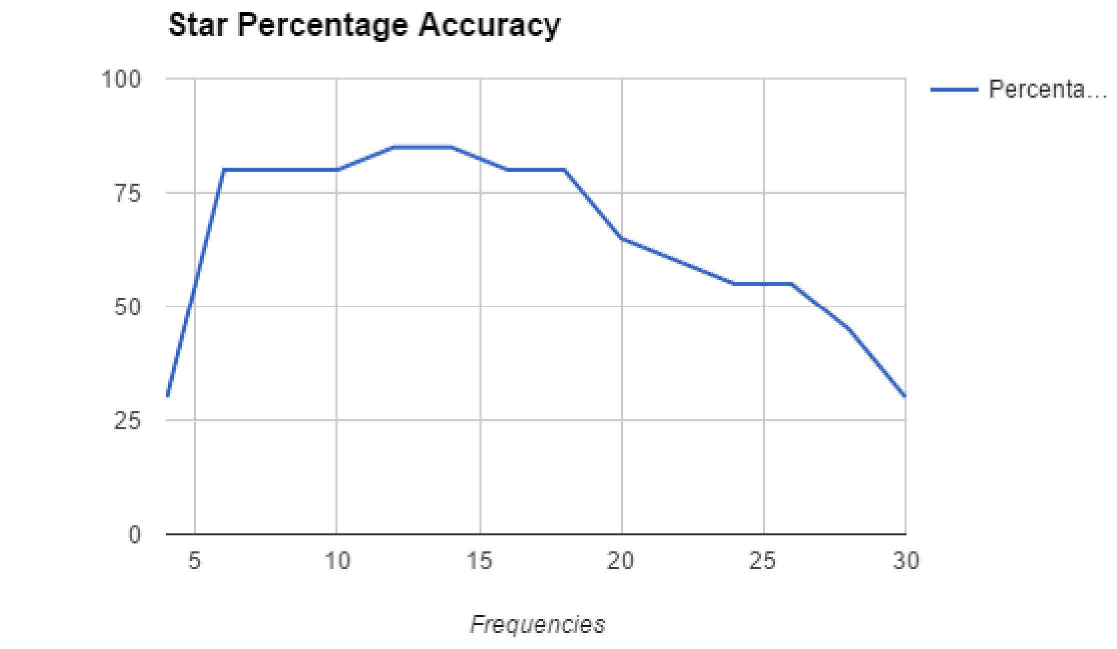
However, this didn’t mean that this frequency was optimal for every class, and obviously there are tradeoffs where you have to sacrifice the classification percentage of certain classes.

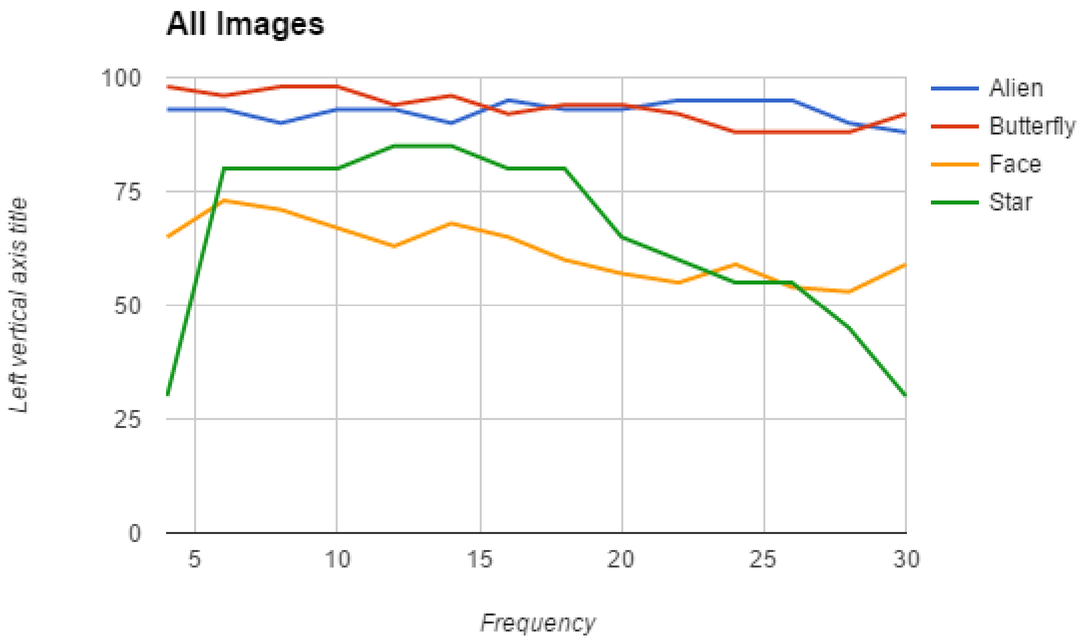
Here are four graphs which show which frequencies had the highest percentage accuracy for the Alien, Butterfly, Face and Star images.




Butterfly Percentage Accuracy
100
96
92
88
84
5
10
15
20
25
30
Frequencies
Percenta...


Face Percentage Accuracy
80
72
64
56
48
5
10
15
20
25
30
Frequencies
Percenta...





The graphs show that some frequency ranges are far better than others at successful classification. For example despite 6 giving the best overall performance, 12 and 14 perform better on the Star class. We can see that a frequency of 6 is only the best performing in one of the four classes on an individual basis whereas on the whole it is the best.

Conclusion

Completing this coursework opened up lots of new maths to me that I had never seen before. Fourier transforms were completely new to me and presented a tough challenge for me to get my head around and try and grasp the concept of, something that I now believe I have achieved. This coursework further reinforced my knowledge of matrices, something that I had not learnt before coming to university. It has also made me far more comfortable using MATLAB to program in.

Bibliography

Hendriks, C. (2010). *How to obtain the chain code.* Available: http://www.cb.uu.se/~cris/blog/index.php/archives/324. Last accessed 24th April 2015.