INFO 3406 Assignment 1

Image Classification

# Introduction

## Aim – 0.5

The aim of this assignment is to classify images without human input, using only labelled images to train the classifier. The aim is also to report on the accuracy and methodology used to classify images.

This is an important task in programming, and one which has not been completely solved (modern methods to classify images in the cifar-10 dataset have 11% error). Image classification is important for tasks such as computer vision, handwriting recognition, and also for self-driving cars to identify hazards.

## Methods - 5

### Similarity metrics

The method used in my assignment is as follows;

1. For each of the training and testing images convert it too a 32x32x3 array (length \* width \* rgb).
2. Use SVD to reduce the dimension of the dataset. Only keep the 10 most important vectors after svd (assume that they suitably represent the image).
3. Group all the training images of each class.
4. Calculate the average of the vector which represents each image to get a class image.
5. Calculate the Manhattan distance to each of the test images to each of the class vectors.
6. Weight these distances and then calculate a score.
7. The class with the lowest score (closest vector) is the class which the image is classified as. Robust

One possible means to make the classifier robust is to use flipped and rotated copies of the test image and then take the average of each of them. This would mean that regardless of the orientation of an image it would always be treated the same. However it is doubtful whether this would improve the accuracy/performance of the classifier.

## Analysis and discussion

### Types of images with errors

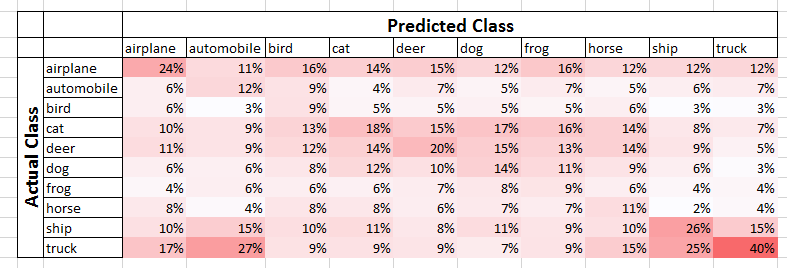
From a survey of the data if became clear that the classifier matched more on some features than others. What I mean by this is that for some classes colour was a very important feature. For example in the frogs data class in cifar10 the frogs which were green were generally far better classified than the frogs which were coloured brown. Similarly the ships were often able to be classified because they had blue colour.

For other classes shape or viewing angle was more important. Birds which were in flight were poorly matched compared to stationary birds.

### Accuracy scores

|  |  |
| --- | --- |
| Dataset | Accuracy (%) |
| Cifar 10 | 18.22 |
| Cifar 100 – coarse classification | 9.67 |
| Cifar 100 – fine classification | 3.0 |

### Confusion matrix



### Speed-Accuracy

* I spent 3 hours writing multithreaded code and it ran slower (too many function calls and bad use of shared memory)
* Time increases linearly with number of vectors used, but after 10 vecors accuracy stops rising
* With more than 5000 training images the classifier doesn’t improve
* Compare to each vector or compare to average of a class is faster than comparing to every other image of that class in the training set

## Conclusion

I created a classifier that satisfies the aim of the assignment. However the accuracy of the classifier is poor, and a better solution should be used if available. Furthermore the image processing time is too high to be convenient and practical (700 seconds is too long for only 30000 images).

Whilst some minor improvements could be made to this code (some configurations had 23.5% accuracy for the cifar10 dataset which is 5% above my submission) I believe that the general method is flawed, and that a vastly different approach should be used if accuracies above 50% are desired/necessary.

## Discussion

Improvements;

When computing the average vector of a class I should use the median not average. This would mean that the outliers had no effect on the data, and it would put the vector closer to the best vector to represent the class.

When classifying flying/not flying birds’ only one vector is used to represent both of these states. This means that you can’t represent both of these states in the one class. The solution would be to use multiple subclasses for each class to represent similar clusters of images in each class.

### Posterior Probability

I’m unsure if this is appropriate for my code;

P(Bird)=0.1

P(chosen as bird|bird)=.18

P(chosen as bird| not bird)=.1

Posterior Probability;

P(Bird|chosen as bird)=0.166

## References – change to IEEE

1. Image SVD methodology - <http://www.frankcleary.com/svdimage/>
2. Bayesian classifier - <https://ludios.org/bayes/>
3. Classifier accuracy <https://www.cs.toronto.edu/~kriz/cifar.html>

## Appendix – folder setup

