Computer Science Project Progress Report

Name: Michael Painter

E-mail: mp703@cam.ac.uk

Project title: Spectral Image Analysis for Medical Imaging

Supervisors: Dr Pietro Lio', Dr Gianluca Ascolani

Director of Studies: Dr John Fawcett

Overseers: Prof John Daugman, Dr David Greaves

The project was broken down into the following stages in the project proposal (reworded):

1. Implement infrastructure (data structures etc) and reading in raw data;
2. Create a tool used to indicate areas of interest on images that contain the the training data;
3. Implement the main machine learning algorithm(s), implement an out the box solution (using a machine learning library), to solve a 'toy problem' in both cases;
4. Extend the implementation to work for noisy 'toy images' and then real images.

Currently stages 1 and 3 have been completed and stage 2 hasn't, so the project is a little behind schedule. This was due to illness during the Christmas holidays and in the first week of Lent. As stages 3 and 4 can be implemented independently of stage 2 and constitute most of the technical challenges of the project, I will implement stage 2 after implementing stage 4 (which will be implemented as planned during weeks 3/4 of Lent) using the slack block "Lent Week 7-8".

I have built a Random Forests library in Java. Given an instance $\mathbf{x}$ we traverse a binary decision tree using a *weak learner* function $h(\theta; \mathbf{x})$ to make decisions at each node, where θ is a set of learnt parameters local to the tree node. If $h(\theta; \mathbf{x}) = 0$ then we traverse 'left', if $h(\theta; \mathbf{x}) = 1$ then we traverse 'right'. In each leaf node a distribution of posterior probabilities ($\mathbb{P}(\mathcal{C}_i|\mathbf{x})$) are stored. When we wish to classify a vector $\mathbf{x}$ we traverse each tree in the forest, then average the probability distributions giving an overall probability distribution of what class $\mathbf{x}$ belongs to. To learn θ I've used the *information gain ratio* from partitioning the training sequence as an *objective function* to determine the best parameters θ for the weak learner, at each node.

In addition to a Random Forests library, I have used the Java machine learning library Encog. I have provided a few additional functions to train a network and to save neural networks after they have been trained.

We define a *spectral image* or *data cube* as an image which we have obtained a spectrum of intensities for each pixel, opposed to just three values, one for each of red, green and

blue. We index a spectral image using (x, y, f) where x, y are coordinates and f is an index for a frequency band. There is a `DataCube` class which contains functionality to load in a `DataCube` from an RGB image, or from a series of grey-scale images (one for each frequency band).

Finally the machine learning algorithms have been applied to produce a pixel labelling of a spectral image. This simply loops through each pixel and classifies the associated spectrum to provide a label for the pixel.

Initially I found it difficult to consider the question of how can I allow the algorithms to indicate a level of certainty in the label they output, which will be needed when considering a real dataset, where we might be identifying unhealthy tissues (it would be silly to claim that a label is 100% certain to be correct). This problem was solved by introducing posterior probability distributions in the decision trees and taking an average, so we can have a probabilistic output. Prior to this each tree used to vote on a classification and the majority vote was the definite class. A second difficulty I encountered was installing and setting up OpenCV, which became a time consuming and it quickly became obvious that I would be better off using an alternative library for machine learning, which lead to me finding and using Encog.