Week 7:

25th April 2017

Time	Topic	Reading
8.15-	Recap of tutorials last week	
	Lecture: Neural Networks in the Real World	Marsland Chapter 4
9.00-	Discussion exercise: Real Problems for Neural Networks	
11.45-	Lunch break	
12.15-	Questions and answers	
13.00-	Practice: Test your neural network on one real problem	

This PDF document is available in an HTML version at `http://www.hg.schaathun.net/FPIA/week07.html`.

1 Tutorial 7.1: Real Problems for Neural Networks

1.1 Binary Classification

We have worked mostly with the breast cancer data from Wisconsin. This is a very simple case, since we have two classes; we call it *binary classification*. Thus we only need one neuron in the output layer and the two-valued (0, 1) output is sufficient.

The input (feature vector) is already coded for us, so we do not have to think about that.

When we move to other problems we may have to think about coding, both of input and of output.

1.2 Multi-class problems (The iris data set)

We have also introduced the iris data set previously. The input is already floating point features, so we do not need to think about that. However, there are *three* classes, so we cannot

just use the single binary output neuron that we used to.

What options do we have for coding the output vectors? Discuss.

What are the advantages and disadvantages of each of the following options?

1. Change the threshold functions to allow three outputs from one neuron.
2. Use two binary output neurons, which gives four possible classes: (0,0), (0,1), (1,0), (1,1), where we use only three of the four.
3. Use three binary output neurons, one for each class. Thus the classes are represented as: (1,0,0), (0,1,0), (0,0,1)

1.3 Regression

The neural network discussed so far gives discrete output. Either the neuron fires, or it does not. This is suitable for *classification* problems, because classes er discrete entities.

Very often we have prediction problems, where we want to estimate some numeric characteristic from a set of features. Say, you want to estimate the price a picture will command at an auction, based on visual features. The price is a continuous variable, so the discrete output will not suite

Consider the wine quality data set from MCR. Open the CSV file and consider the data. Discuss the following:

1. Is this a regression or classification problem? Or could it be either?
2. What reasons/advantages exist for interpreting it as a regression problem?
3. What reasons/advantages exist for interpreting it as a classification problem?
4. How can you adapt a neural network to solve the problem?

Train and test a neural network to assess wine quality.

2 Tutorial 7.2: Randomising the order of the training items

In this exercise, we will randomise the order of the training items at each epoch.

2.1 Shuffling a list

First we focus on writing the `shuffle` function that can randomly shuffle a list.

```
shuffle :: TFGen -> [a] -> [a]
```

The first argument is a TFGen generator. You may need to use it to add randomness while shuffling. There are many different ways to implement this function. If you cannot find one, use the following hints. Although they can lead you to a simple solution, it is not an optimal one. You also can explore some good solutions here.

2.1.1 Hints

1. Implement a recursive function which takes a `TFGen` generator and an integer `n` as inputs and produces a list of `n` random `Word32`.

   ```
   randomList :: TFGen -> Int -> [Word32]
   ```

2. Complete `shuffle'`. This function can shuffle a list using a random stream of `Word32` and modular arithmetic.

   ```
   shuffle' :: [Word32] -> [a] -> [a]
   ...
   shuffle' (x:xs) ls = let (firsts, rest) = splitAt ((fromIntegral x `
                        in ...
   ```

3. Complete `shuffle` using the supporting functions you have just implemented.

2.2 Shuffling the training set at each epoch

Modify your `trainNetwork` function to shuffle the order of the training items at each epoch. Test your new function and compare the results.