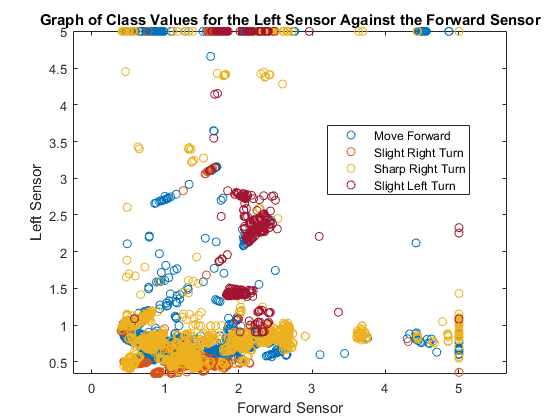
INCA Summative

# Section A:

The problem is to guide a mobile robot around a room in a wall following navigation task. The data was collected in four rounds from 24 sensors attached in a clockwise direction around the robot. There are 5456 total rows of data with each row of data having 24 columns of sensor data and one of four class labels off:

* Move-Forward
* Slight-Right-Turn
* Sharp-Right-Turn
* Slight-Left-Turn

As the problem is to input a row of sensor data and output a class label this is a classification problem. As the robot is travelling clockwise round, the most important sensors will be the forward sensor and the left hand sensor. Therefore as an initial test, the left sensors value is plotted against the forward sensors value. It is then coloured differently for each class label. There is no clear distinction between the data. Therefore is also a non-linearly separable classification problem.



As this is a non-linearly separable classification problem, there are different types of architecture available to choose from. The obvious choice is a multi-layer perceptron (referred to as a MLP from here on). In addition to a MLP, a recurrent neural network (referred to as a RNN) is also suitable for this problem.

|  |  |
| --- | --- |
| Multi-Layer Perceptron | |
| Technical Features | |
|  | |
| Advantages | **Disadvantages** |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Recurrent Neural Network | |
| Technical Features | |
|  | |
| Advantages | **Disadvantages** |
|  |  |
|  |  |
|  |  |

The input data is 5456 rows of 24 columns of sensor data. The sensor data is all in double form. As the meta-data accompanying the data set describes each sensor data is valid. The data was verified again according to the meta-data. To take an example for the first sensor “US 1” meta-data states:

|  |  |  |  |
| --- | --- | --- | --- |
| Max | Min | Mean | SD |
| 5.0000 | 0.40000 | 1.47162 | 0.80280 |

This was calculated again after importing into MatLab - no value is missing. The meta-data describes that four rounds took place however, does not state if these rounds were of equal size or not. Therefore, it will be assumed that the rounds are of equal size.

|  |  |  |  |
| --- | --- | --- | --- |
| Round 1 | Round 2 | Round 3 | Round 4 |
| 1:1364: | 1365:2728 | 2729:4092 | 4093:5456 |

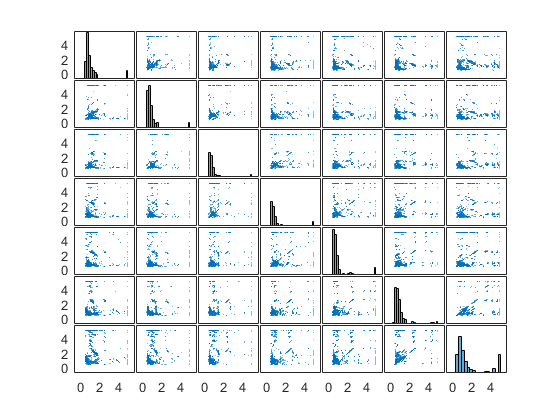
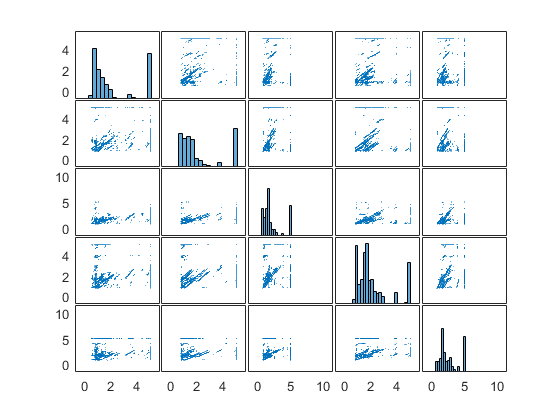
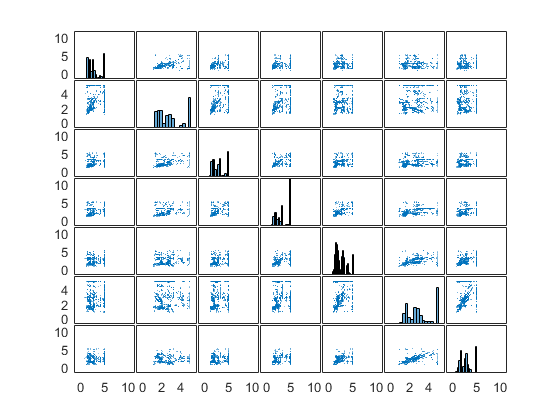
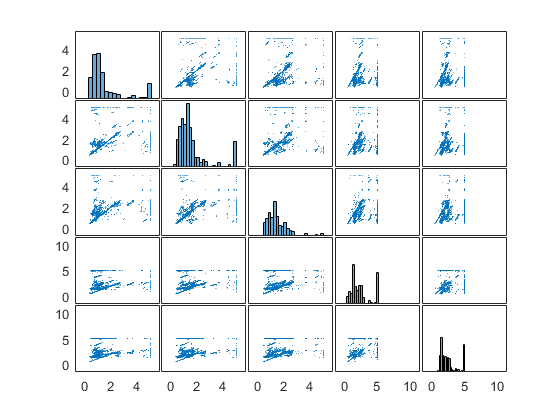
The target data was initially a string – the class labels described above, therefore this data needed to be one hot encoded. Each class label was assigned a number from one to four. These integers were then one hot encoded. The table shows the conversion.

|  |  |  |
| --- | --- | --- |
| String | Integer | One hot encoded |
| Move-Forward | 1 | 1000 |
| Slight-Right-Turn | 2 | 0100 |
| Sharp-Right-Turn | 3 | 0010 |
| Slight-Left-Turn | 4 | 0001 |

**In addition as using a RNN will need to turn the data into time series. 5456 timesteps with each timestep having a vector of the 24 sensor values. This will be done for the one hot encoded targers as well.**

Was also necessary to see if it was possible to reduce the number of inputs. This will be done by seeing if any of the input fata is correlated. This will be done using the matlab function plotmatrix “plotmatrix([X](http://uk.mathworks.com/help/matlab/ref/plotmatrix.html#inputarg_X),[Y](http://uk.mathworks.com/help/matlab/ref/plotmatrix.html" \l "inputarg_Y)) creates a matrix of sub-axes containing scatter plots of the columns of X against the columns of Y. If X is *p*-by-*n* and Y is *p*-by-*m*, then plotmatrix produces an *n*-by-*m* matrix of subaxes." [ADD REF]. There are four plots:

* The first is those sensors to the front:
  + US23, US24, US1, US2, US3.
* The second is those sensors to the right
  + US4, US5, U6, U7, US8, US9, U10
* The third is those sensors to the back
  + US11, US12, US13, US14, US15.
* The fourth is those sensors to the left:
  + US16, US17, US18, US19, US20, US21, US22



The image demonstrates that no sensors are correlated.

**Principal component analysis was also performed**

**STEP BY STEP**

### Training

* Which training algorithm used
  + What training alg using
* How selected
  + How did I select this ?
    - Why ?
    - Did I test multiple times
* Sufficient detail for someone else to re-implement
  + E.g info from matlab website plus others
  + Need references

### Evaluations

* Num neurons/layers
* How split data
* Explain how compare networks
* How decided on the final network
* Graph of all four data sets
* What this means for nns / input data that ca n be discarded
* What this means in real terms – good bad etc
* Test again but with changing feeback and delay size
* Pick the best
* Test again
* What this means

### Network

* Describe best network
* What is this best network
* Someone else should be able to re-implement
* Describe all of its layesr training functions etc
* Picture
* Graphs for just that data
* Matlab defaults

### Results

* Relate these back to problem
* What do my results mean
* Does my NN solve the problem
* What does this mean in terms of the problem

# Section B:

* 2.75 pages
* NNs in recent lit for vision processing for robots
* Paper 1
  + Intro
  + Main features of the architecture
  + How network is trained
  + How much data is needed
  + How data is pre processed
  + Effectiveness for a robot vision system
* Paper 2
  + Intro
  + Main features of the architecture
  + How network is trained
  + How much data is needed
  + How data is pre processed
  + Effectiveness for a robot vision system

Overall approach is OK.

You removed the subject number and test time, for a good reason. However, the brief specified that you should create two datasets, one with only voice data and one with everything, to show the effect of leaving them in. Make sure you follow the spec.

Take care in using PCA. PCA does not discard the original variables! It produces a new set of variables, from which you can discard the least useful.

You need to be more careful in presenting results. When quoting MSEs be very clear about which dataset they relate to, and put them in the context of the problem – is the network doing anything useful given the MSEs you have. It is a regression problem – why not use regression plots?

You need to be more explicit about the network details and implementation, to ensure your work is reproducible.

## Bibliography:

**PLOT MATRIX**

http://uk.mathworks.com/help/matlab/ref/plotmatrix.html

* Paper 1
* Paper 2
* MATLAB refs