FMCS1: Neural Computation
Lab 2

Mark van Rossum & Gaurav Malhotra

February 1, 2006

MATLAB provides a useful toolbox to construct neural networks and test them out on a set of data. This toolbox abstracts away from the MATLAB data structures and provides a user-interface to design elementary neural network structures like layers, connections (with weights) and inputs.

In order to run the tool, type in nntool at the matlab command prompt. This would open a user-interface that allows you to design input, target datasets and networks, as well as to test them. Play around with it, until you feel you are familiar with the interface and its functions. Use 'Help' to find out more.

1. Using nntool construct a perceptron and try to make it learn the following logical operations *[Hint: In order to create a perceptron you would have to change the Network type in network configuration window]*:
   (a) **AND** *[Hint: dataset = Input[0 1 0 1;0 0 1 1] Output[0 0 0 1]]*
   (b) **OR**
   (c) **XOR**
2. Now use a feedforward network instead and try to make it work for the same data sets.
3. Try to change the *transfer function*. Does it change the output?
4. Change the learning rate parameter μ and see how learning converges. *[Don't forget to re-initialise the weights]*
5. Now make a new network with a hidden layer of 2 units and try to make it learn the following data. This is an example of the *parity problem* – the network's output depends on whether the number of 1's in the input are odd, or even:
   - Input patterns: [-1 -1 1], [-1 1 1], [1 -1 1], [1 1 1]
   - Output patterns: [-1 1 1 -1]

6. Now clear all variables and make a new network for making it learn the mapping between a binary number and its corresponding decimal number, for the binary numbers $001, 010, 011$. After you succeed in doing this, try increasing the number of mappings - say, make it remember 8 patterns. What does this tell you about the network capacity?