Introduction to Artificial Intelligence
Autumn Semester 2017
Sept 25, 2017

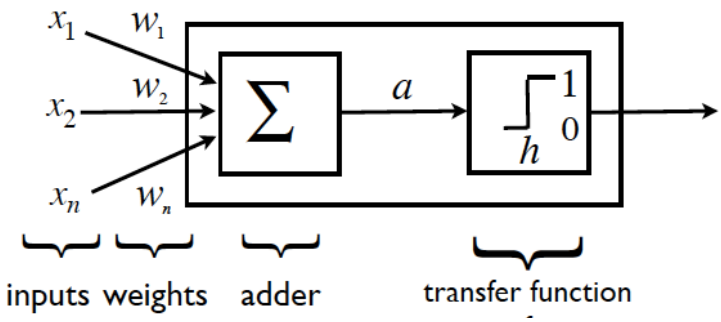
The University of Birmingham
School of Computer Science
Jeremy L Wyatt

Exercise 1 - Neural Networks

1 Hand in requirements

This is an optional exercise, so you have the opportunity to obtain feedback on a typical assignment before attempting exercises that contribute to your module mark. **The mark awarded is indicative and will *not* count towards your overall module mark**. You need to submit your answers on not more than six sides of A4 and submit them as a pdf file via Canvas. You should type the text for your answers. Figures and text must be legible. We recommend you draw figures using a simple drawing program. You can plot graphs using any program you like, e.g. Excel. For graphs you should always include axes, axis labels, a title, and wherever possible units. The deadline is 12 noon on Tuesday 10th October. You will receive a mark and feedback within two weeks of submission. You must submit your work anonymously, with your Student ID number and **NOT** your name on the top.

Q1 A simple neural model is shown graphically here:



The activation is given by the following equation:

$$a = \sum_{i=1}^{n} w_i x_i$$

Let $\vec{x}^T = \langle 1, 0, 1 \rangle$
Let $\vec{w}^T = \langle 0.2, -1, 0.5 \rangle$

Where T stands for the vector transpose operator (which flips a row vector to become a column vector and vice versa).

i) What is the activation?
ii) If the threshold $h = 0.5$ what is the output y ?
iii) What would the output be if the threshold h=0.8?

(1 mark for this question)

Q2. Suppose the learning rate $\alpha = 0.1$, and the target output $t = 0$. Assume that the threshold h=0.5. The update equation in general is

$$w_i' = w_i + \alpha(t - y)x_i$$

The starting weight vector is again $\vec{w}^T = \langle 0.2, -1, 0.5 \rangle$
The set of patterns is

	Input Patterns			Target Output
	x_1	x_2	x_3	t
pattern p_1	-1	0	0	1
pattern p_2	-1	0	1	1
pattern p_3	-1	1	0	1
pattern p_4	-1	1	1	0

For pattern p_1 the activation is

$$a = \sum_{i=1}^{n} w_i x_i = 0.2 \times -1 + -1 \times 0 + 0.5 \times 0 = -0.2$$

So y = 0. The update for w_1 for pattern p_1 is:

$$\begin{aligned} w_1' &= w_1 + \alpha(t - y)x_1 \\ &= 0.2 + 0.1 \times (1 - 0) \times -1 \\ &= 0.1 \end{aligned}$$

i) Now write out the specific update equations for pattern p_1 every other weight in $\vec{w}$

ii) Write down the weight vector after this single step of learning on pattern p_1

iii) Now, for incremental learning, show the weight changes, in the first epoch of training, for the patterns until p_4

iv) Why does learning not change the weight vector for pattern p_4?

v) Will the incremental LMS rule still make a weight change for pattern p_4?

vi) Why?

(3 marks for this question)

Q3 Using the demonstration applet for Neural Networks in AISpace (www.aispace.org) website, complete the following questions. Start the demo neural network program. Complete the tutorials until you are confident how to use the applet.

i) Enter the data below as the training set and as the test set.

	Input Patterns		Target Output
	x_1	x_2	t
pattern p_1	0	0	0
pattern p_2	1	0	1
pattern p_3	0	1	1
pattern p_4	1	1	0

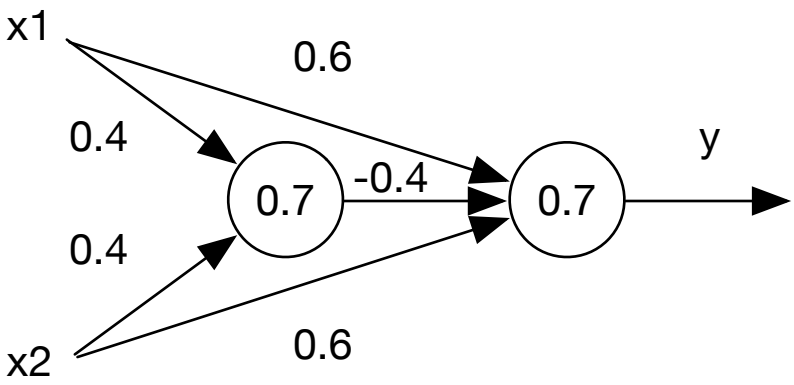
ii) Now initialise the neuron's weights randomly. Record the initial test error. Run training for 50 steps at a time by pressing the Step 50 x button. Every 50 steps you can check the number of correctly classified test examples in the "Summary statistics" panel. Run for at least 200 steps, or until you see no improvement in the test performance. Then reset the weights and repeat the experiment five times (i.e. perform five trials). Record your experimental results as follows:

a) The start and final values of the weight vector on each trial.
b) The proportion of 'correctly' classified test examples on each trial, every 50 steps. Create a plot of the average test performance over time. (Alternatively, just show five error plots from the neural network simulator).

ii) Does the single neuron converge to classify all test patterns correctly?

iii) Give a reason for your answer to ii).

iv) Will the following network solve XOR?



v) Explain your answer to part iv)

vi) Design a network that will solve XOR. Draw it. Implement it in the Neural Networks applet. Initialise the weights randomly. Use “Step to Target Error” to initialise learning. Stop learning if it runs past 50,000 steps. Run ten trials from different initial weights. Reproduce a representative error graph (or graphs) in your report.

vii) What do you notice about the error on different trials. Explain your observations.

(3.5 marks for this question)

Q4 (optional for 2 bonus marks up to a maximum of 7.5): Derive the learning rule for least mean squares by finding the partial derivative of the error with respect to a weight w_i

(2 marks for this question)

(This question can only be answered if you have some experience of differentiation and the chain rule. If you don’t, don’t worry and I will show the derivation later.)