UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

B.Sc. Examination 2007

COMPUTING AND INFORMATION SYSTEMS

IS53002A (CIS311) Neural Networks

Duration: 2 hours 15 minutes

Date and time:

- *Full marks will be awarded for complete answers to THREE questions. Do not attempt more than THREE questions on this paper. Although each question carries 25 marks, and therefore the result from three questions sums up to 75 marks, the final result will be additionally scaled to 100.*

- *Electronic calculators may be used. The make and model should be specified on the script. The calculator must not be programmed prior to the examination. Calculators which display graphics, text or algebraic equations are not allowed.*

THIS EXAMINATION PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

Question 1.

a) Explain the difference between the two modes of training neural networks: supervised and unsupervised training? **[4]**

b) Which of the following boolean functions can be realised by Perceptron networks: AND, NOT, XOR, OR ? Explain your answer. **[5]**

c) i) How can we perform classification into 3 classes using Perceptron networks with thresholded activation functions? How can we train these networks assuming that their activation functions are: *Threshold* (s) = 1 if $s > 0$ and –1 otherwise? **[4]**

ii) What can be done with unthresholded Perceptron networks, using linear activation functions, to perform classification into 3 classes? **[4]**

d) Give the names and the formulae of the two most commonly used differentiable non-linear activation functions in the nodes of multilayer neural networks. Compare the training effects from using each of these formulae. **[8]**

Question 2.

a) Define the gradient descent training rule for unthresholded Perceptron networks. Explain the meaning of each term in this training rule. **[4]**

b) What is the difference between batch and incremental training of unthresholded Perceptrons? **[3]**

c) Design and train an unthresholded Perceptron to learn from the training examples given below.

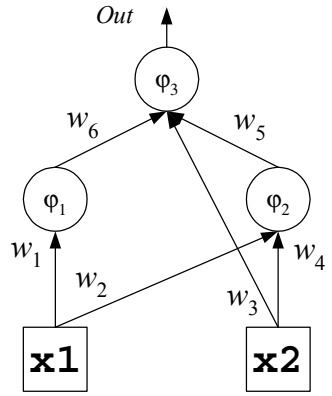
x_1	x_2	x_3	y
0.1	0.5	-0.3	1.1
0.2	0.4	-0.1	1.2
-0.1	0.3	0.6	0.3

i) Draw the topology of the Perceptron network and label each component in it. **[6]**

ii) Train this Perceptron network in an incremental manner with all given examples starting with initial weights (w_1, w_2, w_3) = (-0.1, 0.2, 0.3), and using learning rate η=0.12. **[12]**

Question 3.

Consider the multilayer neural network with irregular topology illustrated in the figure below. This network has two hidden nodes, and one output node, all of which use sigmoidal activations. There are two inputs to the network (x_1, x_2), and six weights as illustrated in the figure.



Start neural network training from the following initial weights:

$w_1 = -0.1$ $w_2 = -0.2$ $w_3 = 0.3$ $w_4 = 0.4$ $w_5 = 0.5$ $w_6 = -0.6$

Train this multilayer network with the backpropagation algorithm using learning rate $\eta=0.13$, and zero momentum, using the following input vector:

x_1	x_2	y
1	1	1

Show the output of each node, the error effects β (beta), the weight updates *delta-w*, and finally the modified weights. **[25]**

Question 4.

a) Describe the unsupervised training algorithm for self-organizing Kohonen networks. **[6]**

b) Before training self-organizing Kohonen networks the input vectors and the weights should be normalized. The normalization of a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is performed by multiplying its elements with a positive number, c.

i) Give the formula for computing the positive normalizer, c. **[3]**

ii) Normalize the following vector $\mathbf{x} = (1.5, -2.4, 3.5, 4.6)$. **[6]**

c) Suppose that a self-organizing Kohonen network with two neurons is given. Each neuron has three inputs. One input vector is given for training $(x_{11}, x_{12}, x_{13}) = (0.8, 0.33, 0.5)$. Train this network starting from initial weights $\mathbf{w}_1 = (-0.3, -0.51, 0.8)$, and $\mathbf{w}_2 = (-0.77, 0.57, 0.27)$. Use learning rate $\eta = 0.2$, and distance function $h = 1$.

i) Determine the index of the neuron to fire. **[5]**

ii) Calculate the updated weight vector of the chosen neuron. **[5]**

Question 5.

a) Present the training algorithm for radial-basis function (RBF) networks. **[6]**

b) Is it necessary to train the radial-basis function centers $\mathbf{x}_i$ and variances σ_i^2 ? Explain briefly your answer. **[4]**

c) What is the most distinguishing characteristic of Hopfield neural networks with respect to the connectivity pattern? **[4]**

d) Give the Widrow-Hoff learning rule for Hopfield neural networks. **[5]**

e) Consider the following weight matrix for a Hopfield neural network:

$$\mathbf{W} = \begin{bmatrix} 0 & -0.2 & 0.8 & -0.3 \\ -0.2 & 0 & 0.4 & 0.5 \\ 0.8 & 0.4 & 0 & -0.1 \\ -0.3 & 0.5 & -0.1 & 0 \end{bmatrix}$$

Compute the output from the first neuron using the input pattern [0 1 1 1]. Assuming that the target is negative, decide whether to train the network. If it is necessary to conduct training, use the Widrow-Hoff rule to correct the weights. **[6]**

UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

B.Sc. Examination 2007

COMPUTING AND INFORMATION SYSTEMS

IS53002A (CIS311) Neural Networks

Duration: 2 hours 15 minutes

Date and time:

- *Full marks will be awarded for complete answers to THREE questions. Do not attempt more than THREE questions on this paper. Although each question carries 25 marks, and therefore the result from three questions sums up to 75 marks, the final result will be additionally scaled to 100.*

- *Electronic calculators may be used. The make and model should be specified on the script. The calculator must not be programmed prior to the examination. Calculators which display graphics, text or algebraic equations are not allowed.*

THIS EXAMINATION PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

Solutions CIS311 INTERNAL

Question 1.

a) Explain the difference between the two modes of training neural networks: supervised and unsupervised training? **[2+2=4]**

The supervised mode of training neural networks assumes that the given batch of training examples involves both the inputs and the desired targets, that is each training input comes with a corresponding output.

The unsupervised mode of training neural networks assumes that the desired targets are not given with the inputs, that is the training algorithm should also learn the targets.

b) Which of the following boolean functions can be realised by Perceptron networks: AND, NOT, XOR, OR ? Explain your answer. **[3+2=5]**

A Perceptron network can learn only linearly separable functions, that is why, it can realise only the three linear Boolean functions AND, NOT, and OR.
The Perceptron can not realise the XOR function because it is nonlinear.

c) i) How can we perform classification into 3 classes using Perceptron networks with thresholded activation functions? How can we train these networks assuming that their activation functions are: *Threshold* (s) = 1 if $s > 0$ and –1 otherwise? **[4]**

We can take 2 such Perceptrons networks using thresholded activation functions to achieve classification into 3 classes, and train them so that each Perceptron is trained to produce thresholded output 1 from sums $s_c > 0$ from the data from the corresponding class, and thresholded output -1 from sums $s_c <= 0$ from the remaining data not in this class.

ii) What can be done with unthresholded Perceptron networks, using linear activation functions, to perform classification into 3 classes? **[4]**

Having the possibility to use Perceptron networks that behave like linear functions, we can take 2 such networks to achieve classification into 3 classes. Each Perceptron should be trained to produce output $s_c > 0$ from the data from the corresponding class, and $s_c <= 0$ from the remaining data not in this class.

d) Give the names and the formulae of the two most commonly used differentiable non-linear activation functions in the nodes of multilayer neural networks. Compare the training effects from using each of these formulae. **[2x3+2=8]**

The training algorithm for multilayer networks requires differentiable continuous nonlinear activation functions. The most popular such functions are:

- the *sigmoid*, or *logistic function*: $Output = \sigma (s) = 1 / (1 + e^{-s})$, where s is the sum: $s = \Sigma_{i=0}^{d} w_i x_i$ of products from the weights w_i and the inputs x_i;
- the *hyperbolic tangent function*: $Output = \tanh(s) = (e^{s} - e^{-s}) / (e^{s} + e^{-s})$, where s is the sum: $s = \Sigma_{i=0}^{d} w_i x_i$ of products from the weights w_i and the inputs x_i.

Training multilayer networks using the hyperbolic tangent function is faster because this function has a steeper shape than the sigmoid, and this facilitates the learning process.

Question 2.

a) Define the gradient descent training rule for unthresholded Perceptron networks. Explain the meaning of each term in this training rule. **[4]**

The gradient descent training rule for unthresholded Perceptrons is: $w_i = w_i + \eta\ (\ y_e - o_e\ )\ x_{ie}$

where y_e is the desired output, o_e is the network output, x_i are the inputs, and η is the learning rate.

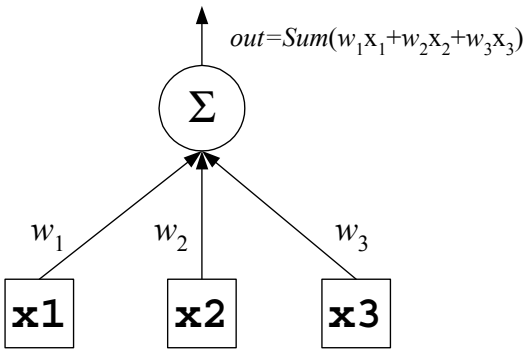
b) What is the difference between batch and incremental training of unthresholded Perceptrons? **[3]**

The batch training algorithm for unthresholded Perceptrons updates the network weights at the end of the training epoch with all given examples, while incremental training algorithm updates the network weights after each training example.

c) Design and train an unthresholded Perceptron to learn from the training examples given below.

x_1	x_2	x_3	y
0.1	0.5	-0.3	1.1
0.2	0.4	-0.1	1.2
-0.1	0.3	0.6	0.3

i) Draw the topology of the Perceptron network and label each component in it. **[6]**



ii) Train this Perceptron network in an incremental manner with all given examples starting with initial weights (w_1, w_2, w_3) = (-0.1, 0.2, 0.3), and using learning rate η=0.12. **[3x4=12]**

Example: `( 0.1, 0.5, -0.3 ) | 1.1`

s = (-0.1)*0.1 + 0.2*0.5 + 0.3*(-0.3) = 0.0
w_1 = (-0.1) + 0.12 * (1.1 - 0.0) * 0.1 = -0.0868
w_2 = (0.2) + 0.12 * (1.1 - 0.0) * 0.5 = 0.266
w_3 = (0.3) + 0.12 * (1.1 - 0.0) * (-0.3) = 0.2604

Example: `( 0.2, 0.4, -0.1 ) | 1.2`

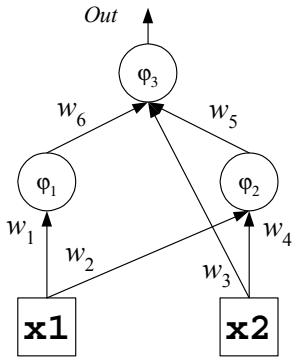
s = (-0.0868)*0.2 + 0.266*0.4 + 0.2604*(-0.1) = 0.063
w_1 = (-0.0868) + 0.12 * (1.2 - 0.063) * 0.2 = -0.0595
w_2 = (0.266) + 0.12 * (1.2 - 0.063) * 0.4 = 0.3206
w_3 = (0.2604) + 0.12 * (1.2 - 0.063) * (-0.1) = 0.2468

Example: `( -0.1, 0.3, 0.6 ) | 0.3`

s = (-0.0595)*(-0.1) + 0.3206*0.3 + 0.2468*0.6 = 0.2502
w_1 = (-0.0595) + 0.12 * (0.3 - 0.2502) * (-0.1) = -0.0601
w_2 = (0.3206) + 0.12 * (0.3 - 0.2502) * 0.3 = 0.3224
w_3 = (0.2468) + 0.12 * (0.3 - 0.2502) * 0.6 = 0.2504

Question 3.

Consider the multilayer neural network with irregular topology illustrated in the figure below. This network has two hidden nodes, and one output node, all of which use sigmoidal activations. There are two inputs to the network (x_1, x_2), and six weights as illustrated in the figure.



Start neural network training from the following initial weights:

$w_1 = -0.1$ $w_2 = -0.2$ $w_3 = 0.3$ $w_4 = 0.4$ $w_5 = 0.5$ $w_6 = -0.6$

Train this multilayer network with the backpropagation algorithm using learning rate 0.13, and zero momentum, using the following input vector:

x_1	x_2	y
1	1	1

Show the output of each node, the error effects β (beta), the weight updates *delta-w*, and finally the modified weights. **[4+4+12+5=25]**

Example: `(0,1) | 1`

Outφ1 = Sigma(-0.1 * 1) = Sigma(-0.1) = 0.475

Outφ2 = Sigma(-0.2 * 1 + 0.4 * 1) = Sigma(-0.2) = 0.4502

Outφ3 = Sigma(0.3 * 1 + 0.5 * 0.4502 - 0.6 * 0.475) = Sigma(0.2401) = 0.5597

β-φ3 = 0.5597 * (1 - 0.5597) * (1 - 0.5597) = 0.1085
delta-w3 = 0.13 * (0.1085) * 1 = 0.0141
delta-w5 = 0.13 * (0.1085) * 0.4502 = 0.0063
delta-w6 = 0.13 * (0.1085) * 0.475 = 0.0067

β-φ2 = 0.4502 * (1 - 0.4502) * [(0.1085) * (0.5)] = 0.0134
delta-*w*2 = 0.13 * (0.0134) * 1 = 0.0017
delta- *w*4 = 0.13 * (0.0134) * 1 = 0.0017

β-φ1 = 0.475 * (1 - 0.475) * [(0.1085) * (- 0.6)] = - 0.0162
delta- *w*1 = 0.13 * (- 0.0162) * 1 = - 0.0162

---

*w*1 = (-0.1) + (- 0.0021) = - 0.1021

*w*2 = 0.25 + 0.0017 = 0.2517

*w*3 = 0.3 + 0.0141 = 0.3141

*w*4 = 0.4 + 0.0017= 0.4017

*w*5 = 0.5 + 0.0063 = 0.5063

*w*6 = (-0.6) + 0.0067 = - 0.5933

Question 4.

a) Describe the unsupervised training algorithm for self-organizing Kohonen networks. **[6]**

The algorithm for unsupervised training of for self-organizing Kohonen neural networks is:

1) *Initialize* the weights with small random values

2) *Repeat*

Draw a training vector $\mathbf{x}_e$ with a certain probability

- *Calculate* the output of each neuron

$$s_m = \sum_{i=1}^{d} w_i \, \mathbf{x}_{e\,i} \qquad i <= m <= M$$

- *Determine* the winner index

$$i(\mathbf{x}_e) = \arg\min_m (s_m)$$

- *Isolate* a neighborhood of neurons ($n < M$)

$$h(n, i) = \exp(- l_{n\,i}^2 / 2\sigma^2), \quad l_{n\,i} = \| \mathbf{x}_e - \mathbf{w}_n \|$$

- *Update* the weights

$$\mathbf{w}_n = \mathbf{w}_n + \eta \, h(n, i)(\mathbf{x}_e - \mathbf{w}_n)$$

Until the changes become less than the predefined threshold.

b) Before training self-organizing Kohonen networks the input vectors and the weights should be normalized. The normalization of a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is performed by multiplying its elements with a positive number, c.

i) Give the formula for computing the positive normalizer, c. **[3]**

$c = 1 / \text{sqrt}(x_1^2 + x_2^2 + \ldots + x_n^2)$, where sqrt is the square root function.

ii) Normalize the following vector: $\mathbf{x} = (1.5, -2.4, 3.5, 4.6)$. **[6]**

$c = 1 / \text{sqrt}(1.5^2 + (-2.4)^2 + 3.5^2 + 4.5^2) = 0.1571$

$\mathbf{x} = (1.5 * 0.1571, \ (-2.4)* 0.1571, \ 3.5* 0.1571, \ 4.6* 0.1571)$

$= (0.2356, \ - 0.377, \ 0.5498, 0.7227)$

c) Suppose that a self-organizing Kohonen network with two neurons is given. Each neuron has three inputs. One input vector is given for training $(x_{11}, x_{12}, x_{13}) = (0.8, 0.33, 0.5)$. Train this network starting from initial weights $\mathbf{w}_1 = (-0.3, -0.51, 0.8)$, and $\mathbf{w}_2 = (-0.77, 0.57, 0.27)$. Use learning rate $\eta = 0.2$, and distance function $h = 1$.

i) Determine the index of the neuron to fire. **[5]**

The summation block is computed as follows:

$$\mathbf{s}^t = \mathbf{W}\mathbf{x}^t = \begin{vmatrix} -0.3, & -0.51, & 0.8 \\ -0.77, & 0.57, & 0.27 \end{vmatrix} * (0.8, 0.33, 0.5)^t = \begin{vmatrix} - 0.0083 \\ - 0.2929 \end{vmatrix}$$

Therefore the selected neuron index is: $i = 1$.

ii) Calculate the updated weight vector of the chosen neuron. **[5]**

The weight $\mathbf{w}_1$ has to be changed as follows:

$\mathbf{w}_1 = (-0.3, -0.51, 0.8) + 0.2 * [(0.8, 0.33, 0.5) - (-0.3, -0.51, 0.8)]$

$= (-0.3, -0.51, 0.8) + 0.2 * (1.1, 0.84, -0.3)$

$= (-0.08, -0.342, 0.74)$

Question 5.

a) Present the training algorithm for radial-basis function (RBF) networks. **[6]**

The algorithm for training Radial-basis function networks is:

1) Initialize the examples, and *determine*:

- the network structure with a number n of basis functions φ_i, i=1,2,...,n
- the basis function centers $\mathbf{x}_i$, i=1,2,...,n (using for example the *k-means* clustering algorithm)
- their variances σ_i^2, i=1,2,...,n (by setting σ_i^2 equal to the average distance from $\mathbf{x}_i$ to its 5 nearest neighbors)

2) *Train*

- *calculate the outputs* from each e^{th} example with the Gaussian basis functions:

  $\varphi_{ei} = \exp(- \| \mathbf{x}_e - \mathbf{x}_i \|^2 / 2\sigma_i^2)$ // *at each* i^{th} *hidden unit*where i=1,2,...,n; e=1,2,...,N

- *compute the correlation matrix*: $\mathbf{\Phi}^T \mathbf{\Phi}$, and perform the summation: $\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \mathbf{I}$
- *invert the matrix*: $(\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \mathbf{I})^{-1}$
- compute the vector: $\mathbf{\Phi}^T \mathbf{y}$
- *estimate and the weights*: $\mathbf{w} = (\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \mathbf{I})^{-1} \mathbf{\Phi}^T \mathbf{y}$

b) Is it necessary to train the radial-basis function centers $\mathbf{x}_i$ and variances σ_i^2 ?
Explain briefly your answer. **[4]**

The radial-basis function centers $\mathbf{x}_i$ and variances σ_i^2 may be trained along with the output weights but this takes a long time. That is why, they are often determined in advance.

c) What is the most distinguishing characteristic of Hopfield neural networks with respect to the connectivity pattern? **[4]**

This Hopfield network is a kind of a recurrent neural network with feedback connections, which distinguish it from the so called feedforward networks like the multilayer perceptron. The feedback connections are loops from node outputs toward the inputs of the nodes.

d) Give the Widrow-Hoff learning rule for Hopfield neural networks. **[5]**

The Widrow-Hoff learning rule suggests to compute the summation block of the i-th neuron:
$s_i = \sum_{j=1}^{n} w_{ji} x_j$, and to modify the weights depending on the result. There are two cases:

- if $s_i < 0$ and $x_i = 1$ the output should be made negative by: $w_{ji} = w_{ji} - (0.1 + s_i) / n$
- if $s_i >= 0$ and $x_i = 0$ the output should be made positive by: $w_{ji} = w_{ji} + (0.1 - s_i) / n$

e) Consider the following weight matrix for a Hopfield neural network:

$$\mathbf{W} = \begin{bmatrix} 0 & -0.2 & 0.8 & -0.3 \\ -0.2 & 0 & 0.4 & 0.5 \\ 0.8 & 0.4 & 0 & -0.1 \\ -0.3 & 0.5 & -0.1 & 0 \end{bmatrix}$$

Compute the output from the first neuron using the input pattern [0 1 1 1]. Assuming that the target is negative, decide whether to train the network. If it is necessary to conduct training, use the Widrow-Hoff rule to correct the weights. **[6]**

The output of the first neuron is computed as follows:
$s_1 = w_{11} x_1 + w_{12} x_2 + w_{13} x_{13} + w_{13} x_{13} = 0*0 + (-0.2)*1 + 0.8*1 + (-0.3)*1 = 0.3$,
therefore 1, so, we have to subtract from the weights the quantity:

$(0.1 + s_1) / 4 = (0.1 + 0.3) / 4 = 0.1$

$w_{11} = 0$

$w_{12} = (-0.2) - 0.1 = -0.3$

$w_{13} = 0.8 - 0.1 = 0.7$

$w_{14} = (-0.3) - 0.1 = -0.4$

$$\mathbf{W} = \begin{bmatrix} 0 & -0.3 & 0.7 & -0.4 \\ -0.3 & 0 & 0.4 & 0.5 \\ 0.7 & 0.4 & 0 & -0.1 \\ -0.4 & 0.5 & -0.1 & 0 \end{bmatrix}$$