Philadelphia University
Faculty of Engineering

Marking Scheme

Exam Paper

BSc CE

Neural Networks and Fuzzy Logic (630514)

Final Exam First semester Date: 31/01/2016

Section 1

Weighting 40% of the module total

Lecturer: Dr. Qadri Hamarsheh

Coordinator: Dr. Qadri Hamarsheh

Internal Examiner: Dr. Mohammed Mahdi

Marking Scheme

Neural Networks and Fuzzy Logic (630514)

The presented exam questions are organized to overcome course material through 6 questions.
The *all questions* are compulsory requested to be answered.

Marking Assignments

Question 1 This question is attributed with 12 marks if answered properly; the answers are as following:

1) What are the advantages of biological neural networks (**BNNs**) compared to conventional **Von Neumann** computers?

(i) BNNs have the ability to learn from examples.
(ii) BNNs have a high degree of parallelism.
(iii) BNNs require a mathematical model of the problem.
(iv) BNNs can acquire knowledge by "trial and error".
(v) BNNs use a sequential algorithm to solve problems.

a) **(i), (ii), (iii), (iv) and (v)** **b)** **(i), (iii) and (iv)**
c) **(i) , (ii) and (iii)** **d)** **(i), (ii) and (iv)**

2) A multi-layer feedforward network has **5** input units, a first hidden layer with **4** units, a second hidden layer with **3** units, and **2** output units. How many weights does this network have?

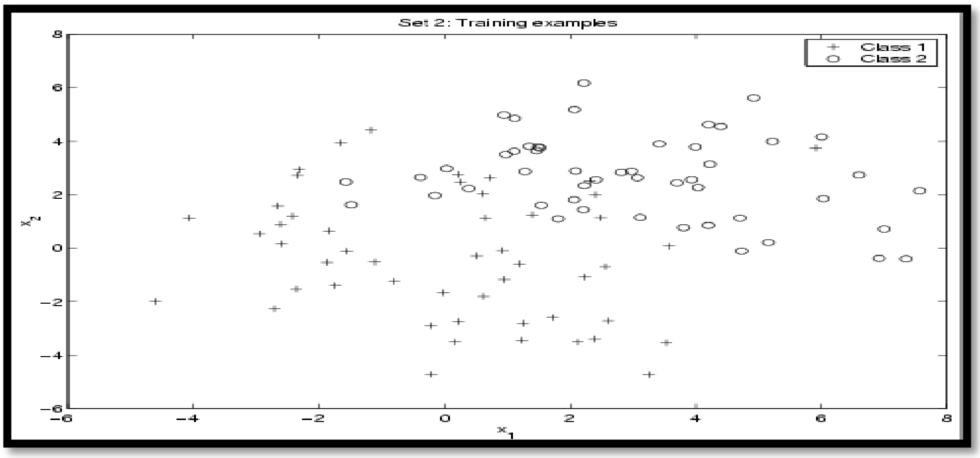
a) **18** **b)** **20**
c) **26** **d)** **38**

3) Which of the following equations is the best description of **Hebbian learning**?

a) $\Delta W_k = \eta y_k X$
b) $\Delta W_k = \eta\,(X - W_k)$
c) $\Delta W_k = \eta\,(d_k - y_k)X$
d) $\Delta W_j = \eta_j\,(X - W_j)$, where $\eta_j < \eta$ and $j \neq k$

Where X is the input vector, η is the learning rate, W_k is the weight vector, d_k is the target output, and y_k is the actual output for unit k.

4) Is the following statement true or false? "A **perceptron** is trained on the data shown below, which has two classes (the two classes are shown by the symbols '**+**' and '**o**' respectively). After many epochs of training, the perceptron will converge and the decision line will reach a steady state."



a) **TRUE**. **b)** **FALSE.**

5) How many hidden layers are there in an **autoassociative** Hopfield network?

a) **None (0).** **b)** **One (1).**
c) **Two (2).** **d)** **unlimited**

6) The maximum number of fundamental memories M_{max} (**All perfectly retrieved**) that can be stored in the **n-neuron** Hopfield network is limited by

a) $M_{max} = 0.15\,n$ **b)** $M_{max} = \frac{n}{2\ \ln n}$

c) $M_{max} = \frac{n}{4\ \ln n}$ **d)** **None of above**

7) An input vector **x** and two prototype vectors $\mathbf{p_1}$ and $\mathbf{p_2}$ are given by

$$\mathbf{x} = [-1.40,\ 2.30,\ 0.20]^T$$
$$\mathbf{p_1} = [-1.00,\ 2.20,\ 0.10]^T$$
$$\mathbf{p_2} = [-4.00,\ 7.00,\ 0.60]^T$$

Which prototype is nearest to **x** in terms of **squared Euclidean distance**?

a) $\mathbf{p_1}$ **b)** $\mathbf{p_2}$

8) Which of the following statements is **NOT** true for a self-organizing map (**SOFM**)?

a)	**The size of the neighbourhood is decreased during training.**
b)	**The units are arranged in a regular geometric pattern such as a square or ring.**
c)	**The weights of the winning unit k are adapted by $\Delta w_k = \eta\,(x - w_k)$, where x is the input vector.**
d)	**The weights of the neighbours j of the winning unit are adapted by $\Delta w_j = \eta_j\,(x - w_j)$, where $\eta_j > \eta$ and $j \neq k$.**

9) What is the equation for **probabilistic or**?

a) **Probor (a,b) = a-b + ab** **b)** **Probor (a,b) = a+b - ab**

c) **Probor (a,b) = ab + ab** **d)** **Probor (a,b) = a/b x ab**

10) What is the **input** and **output** of step 2 of fuzzy logic - **Apply Fuzzy Operator**?

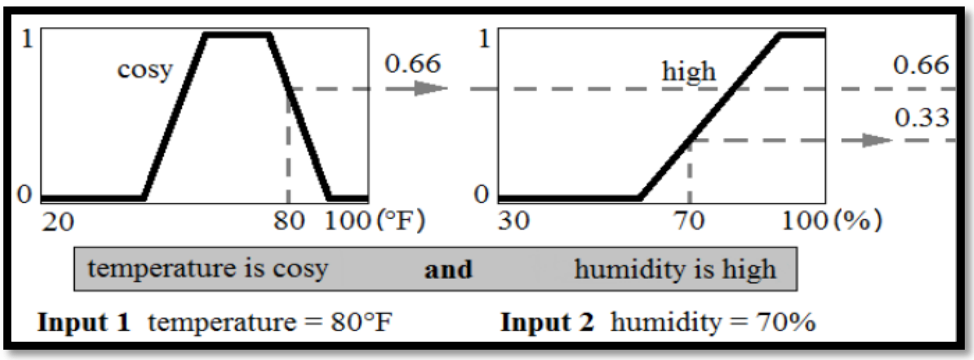
a) **The input is a single truth value and the output has two or more values.**

b) **The input is a value greater than one and the output is a value less than the input.**

c) **The input and output have both the same values.**

d) **The input has two or more values and the output has a single truth value.**

11) The result of **fuzzy operator** shown in the following figure is



a) **0.33** **b)** **0.66**

c) **0.23** **d)** **1**

12) What is the following sequence of steps taken in designing a fuzzy logic machine?

a) **Fuzzification->Rule evaluation->Defuzzification**

b) **Rule evaluation->Fuzzification->Defuzzification**

c) **Fuzzy Sets->Defuzzification->Rule evaluation**

d) **Defuzzification->Rule evaluation->Fuzzification**

Question 2 This question is attributed with 9 marks if answered properly; the answers are as following:

a) *(5 marks)*

Solution

The Self-Organizing Map algorithm can be broken up into 9 steps.

Step 0:

- Initialize synaptic weights w_{ij} to small random values, say in an interval [0, 1].
- Set topological neighborhood parameters.
- Set learning rate parameters (small positive value).

Step 1: While stopping condition is false, do Steps 2-8.

Step 2: For each input vector x chosen at random from the set of training data and presented to the network, do Steps 3-5.

Step 3: For each j, compute:

$$D(j) = \sum_i (w_{ij} - x_i)^2.$$

Euclidean distance is a measurement of similarity between two sets of data. (Every node in the network is examined to calculate which ones' weights are most like the input vector).

Step 4: Find index J such that D(J) is a minimum (The winning node is commonly known as the Best Matching Unit (BMU)).

$$j_{\mathbf{X}}(p) = \min_j \|\mathbf{X} - \mathbf{W}_j(p)\| = \left\{ \sum_{i=1}^{n} [x_i - w_{ij}(p)]^2 \right\}^{1/2},$$
$$j = 1, 2, \ldots, m$$

Where: n is the number of neurons in the input layer,

m is the number of neurons in the Kohonen layer.

Step 5: Learning (Update the synaptic weights):

For all units j within a specified neighborhood of J, and for all i:

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

Where $\Delta w_{ij}(p)$ is the weight correction at iteration p.

The weight correction is determined by the competitive learning rule:

$$\Delta w_{ij}(p) = \begin{cases} \alpha [x_i - w_{ij}(p)], & j \in \Lambda_j(p) \\ 0, & j \notin \Lambda_j(p) \end{cases}$$

Where α is the learning rate parameter, and $\Lambda_j(p)$ is the neighborhood function centered around the winner-takes-all neuron j_X at iteration p.

Any nodes found within the radius of the BMU are adjusted to make them more like the input vector. The closer a node is to the BMU, the more its' weights are altered.

Step 6: Update learning rate (α is a slowly decreasing function of time).

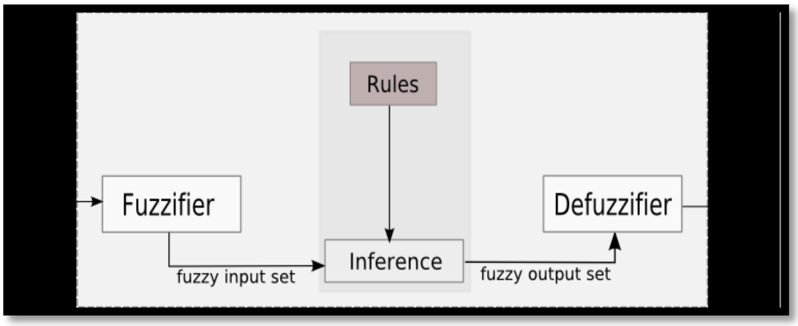
Step 7: Reduce radius of topological neighborhood at specified times. The radius of the neighborhood of the BMU is calculated. This value starts large (typically it is set to be the radius of the network). The radius of the neighborhood around a cluster unit also decreases as the clustering process progresses.

Step 8: Test stopping condition.

b) *(4 marks)*

Solution

- Fuzzy logic system (FLS) can be defined as the nonlinear mapping of an input data set to a scalar output data. A FLS consists of four main parts:
    - Fuzzifier (Fuzzification).
    - Rules.
    - Inference engine.
    - Defuzzifier (Defuzzification).
- These components and the general architecture of a FLS are shown in Figure.



- The process of fuzzy logic:
  - A crisp set of input data are gathered and converted to a fuzzy set using fuzzy linguistic variables, fuzzy linguistic terms and membership functions. This step is known as fuzzification.
  - An inference is made based on a set of rules.
  - The fuzzy output is mapped to a crisp output using the membership functions, in the defuzzification step.
- Fuzzification- The process of generating membership values for a fuzzy variable using membership functions.
- Defuzzification- The process of transforming a fuzzy output of a fuzzy inference system into a crisp output.

Fuzzy logic algorithm

1) Initialization process:
   - Define the linguistic variables.
   - Construct the fuzzy logic membership functions that define the meaning or values of the input and output terms used in the rules.
   - Construct the rule base (Break down the control problem into a series of IF X AND Y, THEN Z rules based on the fuzzy logic rules).
2) Convert crisp input data to fuzzy values using the membership functions (fuzzification).
3) Evaluate the rules in the rule base (inference).
4) Combine the results of each rule (inference).
5) Convert the output data to non-fuzzy values (defuzzification).

Question 3 This question is attributed with 6 marks if answered properly; the answers are as following:

Solution

a) *(1.5 marks)*

Input to top neuron = (0.35x0.1) + (0.9x0.8)=0.755. Output = 0.68.
Input to bottom neuron = (0.9x0.6) + (0.35x0.4) = 0.68. Output = 0.6637.
Input to final neuron = (0.3x0.68) + (0.9x0.6637) = 0.80133. Output = 0.69.

b) *(3 marks)*

Output error δ= (t-o) (1-o) o = (0.5-0.69) (1-0.69)0.69 = -0.0406.
New weights for output layer:
w_1+ = w_1+(δ x input) = 0.3 + (-0.0406x0.68) = 0.272392.
w_2+ = w_2+(δ x input) = 0.9 + (-0.0406x0.6637) = 0.87305.
Errors for hidden layers:
δ_1 = δ x w_1 = -0.0406 x 0.272392 x (1-o) o = $-2.406\text{x}10^{-3}$
δ_2= δ x w2 = -0.0406 x 0.87305 x (1-o) o = $-7.916\text{x}10^{-3}$
New hidden layer weights:
w_3+=0.1 + (-2.406 x 10-3 x 0.35) = 0.09916.
w_4+ = 0.8 + (-2.406 x 10-3 x 0.9) = 0.7978.
w_5+ = 0.4 + (-7.916 x 10-3 x 0.35) = 0.3972.
w_6+ = 0.6 + (-7.916 x 10-3 x 0.9) = 0.5928.

c) *(1.5 marks)*

Old error was -0.19. New error is -0.18205. Therefore error has reduced.

Question 4 This question is attributed with 4 marks if answered properly; the answers are as following:

Solution

$$A \cup B = \left\{ \frac{0.4}{a10} + \frac{1.0}{b52} + \frac{1.0}{b117} + \frac{0}{C5} + \frac{0}{C130} + \frac{0.6}{f4} + \frac{0.9}{f14} + \frac{0.8}{f15} + \frac{1.0}{f16} + \frac{0.7}{f111} + \frac{0}{KC130} \right\}$$

$$A \cap B = \left\{ \frac{0.4}{a10} + \frac{1.0}{b52} + \frac{0.4}{b117} + \frac{0}{C5} + \frac{0}{C130} + \frac{0.5}{f4} + \frac{0.6}{f14} + \frac{0.8}{f15} + \frac{0.3}{f16} + \frac{0.4}{f111} + \frac{0}{KC130} \right\}$$

$$\bar{A} = \left\{ \frac{0.7}{f16} + \frac{0.5}{f4} + \frac{0.6}{a10} + \frac{0.4}{f14} + \frac{0.7}{f15} + \frac{0.3}{f111} + \frac{0}{b117} + \frac{0}{b52} + \frac{1}{C5} + \frac{1}{C130} + \frac{1}{KC130} \right\}$$

$$\bar{B} = \left\{ \frac{0.6}{b117} + \frac{0.6}{f111} + \frac{0.4}{f4} + \frac{0.2}{f15} + \frac{0.1}{f14} + \frac{0}{f16} + \frac{1}{C5} + \frac{1}{C130} + \frac{1}{KC130} \right\}$$

Question 5 This question is attributed with 4 marks if answered properly; the answers are as following:

Solution

```
clear all ; clc; close all;
%Create training data as follows:
x = 0:0.02:1; % x-values for the function
y = x + 0.3*sin(2*pi*x); % Exact function values for each x
p = x; % x-values used for training of the network
t = y + 0.1*randn(size(y)); % Noisy measurements of the function values for p
%Plot the data
plot (p, t, 'x')
grid; xlabel('time (s)'); ylabel('output'); title(' function approximation ')
xt = 0:0.01:1; % x-values to use for testing
yt = xt + 0.3*sin(2*pi*xt); % Correct function values for xt.          (2 marks)
net = newff(minmax(p), [1,1], {'logsig', 'purelin'},'trainlm');
% Define learning parameters
net.trainParam.show = 50; % The result is shown at every 50th iteration.
net.trainParam.lr = 0.05; % Learning rate used in some gradient schemes
net.trainParam.epochs =1000; % Max number of iterations
net.trainParam.goal = 1e-3; % Error tolerance; stopping criterion
net = train(net, p, t);
z = sim(net, xt);
% Maximum fitting error
Maxfiterror = max (z- yt)                                              (2 marks)
```

Question 6 This question is attributed with 5 marks if answered properly; the answers are as following:

Solution 1

```
a=newfis(' water ');
a.andMethod = 'min';                 a.orMethod = 'max';
 a.impMethod = 'min';                a.aggMethod = 'max';      a.defuzzMethod = 'centroid';
a.input(1).name=' level ';                  a.input(1).range=[-1  1];  (1 mark)
a.input(1).mf(1).name='high';               a.input(1).mf(1).type='gaussmf';
a.input(1).mf(1).params=[1.5  -1];          a.input(1).mf(2).name='okay';
a.input(1).mf(2).type='gaussmf';  a.input(1).mf(2).params=[1.5  0];
a.input(1).mf(3).name='low';                a.input(1).mf(3).type='gaussmf';
a.input(1).mf(3).params=[1.5  1];  a.output(1).name='valve';          (1 mark)
a.output(1).range=[0  1];          a.output(1).mf(1).name='closefast'
a.output(1).mf(1).type='trimf';             a.output(1).mf(1).params=[0  0  0.4];
a.output(1).mf(2).name='nochange';          a.output(1).mf(2).type='trimf';      (1 mark)
a.output(1).mf(2).params=[0.1  0.5  0.9];   a.output(1).mf(3).name='openfast';
```

```
a.output(1).mf(3).type='trimf';                 a.output(1).mf(3).params=[0.6  1  1];
a.rule(1).antecedent=[2];                a.rule(1).consequent=[2];
a.rule(1).weight=1;                             a.rule(1).connection=1;
a.rule(2).antecedent=[3];                a.rule(2).consequent=[3];
a.rule(2).weight=1;                             a.rule(2).connection=1;
a.rule(3).antecedent=[1];                a.rule(3).consequent=[1];
a.rule(3).weight=1;                             a.rule(3).connection=1;
%FIS Evaluation
                                evalfis(0.8, a)                (2 marks)
```

Or

Solution 2

```
a=newfis('Water');
a.andMethod = 'min';          a.orMethod = 'max';
 a.impMethod = 'min';          a.aggMethod = 'max';   a.defuzzMethod = 'centroid';
a=addvar(a,'input', 'level',[-1 -1]);
a=addmf(a,'input',1,' high ','gaussmf',[ 1.5  -1]);
a=addmf(a,'input',1,' okay ','gaussmf',[ 1.5  0]);
a=addmf(a,'input',1,' low ','gaussmf',[1.5 1]);              (2.5 marks)
a=addvar(a,'output',' valve ',[0 1]);
a=addmf(a,'output',1,' closefast ','trimf',[ 0  0  0.4]);
a=addmf(a,'output',1,' nochange ','trimf',[ 0.1  0.5  0.9]);
a=addmf(a,'output',1,' openfast ','trimf',[ 0.6   1   1]);
ruleList=[ ...
2 2 1 1
3 3 1 1
1 1 1 1 ];
a=addrule(a,ruleList);
%FIS Evaluation
evalfis(0.8, a)                                              (2.5 marks)
```