The duration of this exam is four hours.

CHALMERS, GÖTEBORGS UNIVERSITET

EXAM for
ARTIFICIAL NEURAL NETWORKS

COURSE CODES: **FFR 135, FIM 720 GU, PhD**

Time:	October 28, 2019, at $08^{30} - 12^{30}$
Place:	Lindholmen-salar
Teachers:	Bernhard Mehlig, 073-420 0988 (mobile)
	Marina Rafajlovic, 076-580 4288 (mobile)
Allowed material:	Mathematics Handbook for Science and Engineering
Not allowed:	Any other written material, calculator

Maximum score on this exam: 12 points.
Maximum score for homework problems: 12 points.
To pass the course it is necessary to score at least 5 points on this written exam.
CTH $\geq$14 passed; $\geq$17.5 grade 4; $\geq$22 grade 5,
GU $\geq$14 grade G; $\geq$ 20 grade VG.

---

1. Energy function in a neural network.
(a) Fig. 1 shows a neural network with two neurons with asymmetric weights, $w_{12} = 2$, and $w_{21} = -1$. The states of the neurons, denoted by S_1 and S_2, are either $+1$ or -1. Show that the energy function

$$H = -\frac{w_{12} + w_{21}}{2} S_1 S_2 \tag{1}$$

can increase under the asynchronous deterministic McCulloch-Pitts rule for updating the second neuron $S_2' = \text{sgn}(w_{21}S_1)$, but not under the deterministic McCulloch-Pitts rule for updating the first neuron $S_1' = \text{sgn}(w_{12}S_2)$. (**0.5**p).
(b) For the network shown in Fig. 1, show that the energy (1) cannot stay constant after a single step of synchronous update rule $S_i' = \text{sgn}(w_{ij}S_j)$, for $i = 1, 2$. (**0.5**p).
(c) Now consider a neural network with N neurons. The states of the neurons, denoted by n_i ($i = 1, \ldots, N$) are either 0 or 1. The weights w_{ij} are symmetric $w_{ij} = w_{ji}$ for $i \neq j$, and $w_{ii} > 0$ for $i = 1, \ldots, N$. Show that the energy function

$$H = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} n_i n_j + \sum_{i=1}^{N} \mu_i n_i \tag{2}$$

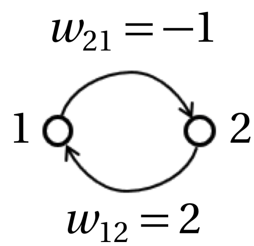

Figure 1: Question 1. Neural network with two neurons and asymmetric weights.

cannot increase under the asynchronous update rule

$$n'_m = \theta_{\rm H}(b_m), \text{ with } b_m = \sum_{j=1}^{N} w_{mj} n_j - \mu_m \ . \tag{3}$$

Here

$$\theta_{\rm H}(b_m) = \begin{cases} 1, & \text{for } b_m > 0, \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

is the Heaviside step function. (**1**p).

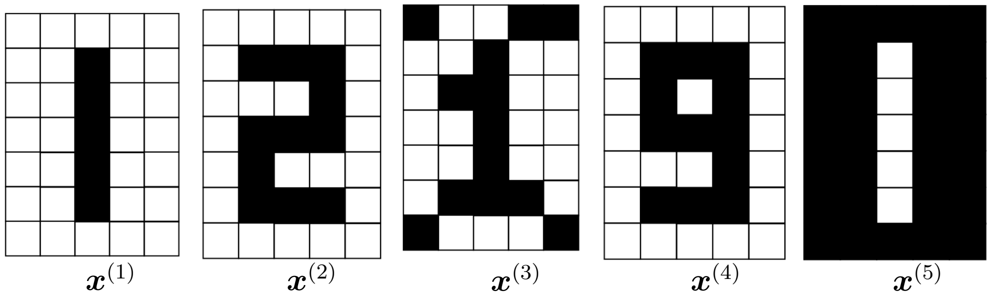

Figure 2: Question 2. Each of the five patterns consists of 35 bits $x_i^{(\mu)}$. A black pixel i in pattern μ corresponds to $x_i^{(\mu)} = 1$, a white one to $x_i^{(\mu)} = -1$.

2. Recognising digits with a Hopfield network. Fig. 2 shows five patterns, each with $N = 35$ bits. Store the patterns $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ in a Hopfield network using Hebb's rule $w_{ij} = \frac{1}{N}\sum_{\mu=1}^{2} x_i^{(\mu)} x_j^{(\mu)}$ with $i, j = 1, \ldots, N$. Use the update rule

$$S_i \leftarrow \operatorname{sgn}\Big(\sum_{j=1}^{N} w_{ij} S_j\Big) . \tag{5}$$

Feed the patterns into the network. To determine their fate, follow the steps outlined below.

(a) Compute $\sum_{j=1}^{N} x_j^{(\mu)} x_j^{(\nu)}$, for $\mu = 1$, $\nu = 1, \ldots, 5$, and also for $\mu = 2$, $\nu = 1, \ldots, 5$. *Hint*: the result can be read off from the Hamming distances between the patterns shown in Figure 2. (**0.5**p).

(b) Consider the quantity $b_i^{(\nu)} = \sum_{j=1}^{N} w_{ij} x_j^{(\nu)}$, where w_{ij} are the weights obtained by storing patterns $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$. Compute $b_i^{(\nu)}$ for $\nu = 1, \ldots, 5$. Express your result as linear combinations of $x_i^{(1)}$ and $x_i^{(2)}$. *Hint*: use your answer to the first part of this question. (**1**p).

(c) Feed the patterns in Figure 2 to the network. Which of the patterns remain the same after one synchronous update according to (5)? (**0.5**p).

3. Linearly inseparable problem. A classification problem is specified in Fig. 3, where a grey triangle in input space is shown. The aim is to map input patterns $x^{(\mu)}$ to outputs $O^{(\mu)}$ as follows: if a point $x^{(\mu)}$ lies inside the triangle it is mapped to $O^{(\mu)} = +1$, but if $x^{(\mu)}$ is outside the triangle it is mapped to $O^{(\mu)} = -1$. How patterns on the boundary of the triangle are classified is not important.
(a) Show that this problem is not linearly separable by constructing a counter-example using four input patterns. (**0.5**p).
(b) The problem can be solved by a perceptron with one hidden layer with three neurons ($j = 1, 2, 3$)

$$V_j^{(\mu)} = \operatorname{sgn}\Big(-\theta_j + \sum_{k=1}^{2} w_{jk} x_k^{(\mu)}\Big) \tag{6}$$

and output

$$O^{(\mu)} = \operatorname{sgn}\Big(-\Theta + \sum_{j=1}^{3} W_j V_j^{(\mu)}\Big) \; . \tag{7}$$

Here w_{jk} and W_j are weights and θ_j and Θ are thresholds. In Fig. 3, the orientation of weight vectors $\boldsymbol{w}_j = (w_{j1}, w_{j2})^{\mathsf{T}}$ corresponding to hidden nodes $j = 1, 2, 3$ is indicated. Using this, find values of w_{jk} and θ_j and that solve the classification problem. (**1**p).
(c) Based on your result in (b), illustrate the problem in the hidden space encoding the outputs, draw a decision boundary that solves the problem, and compute W_j and Θ corresponding to the decision boundary you drew. (**0.5**p).

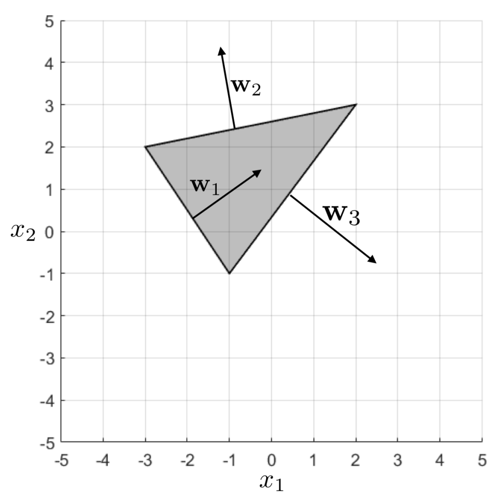

Figure 3: Question 3. Classification problem. Input space is the $x_1 - x_2$-plane. Depicted are the orientations of the weight vectors $\boldsymbol{w}_j$ ($j = 1, 2, 3$) corresponding to the hidden neurons.

μ	$x_1^{(\mu)}$	$x_2^{(\mu)}$	$x_3^{(\mu)}$	$t^{(\mu)}$
1	0	1	0	-1
2	1	0	1	-1
3	0	0	0	$+1$
4	1	0	0	-1
5	1	1	0	-1
6	0	0	1	-1
7	0	1	1	-1
8	1	1	1	$+1$

Table 1: Question 4. Inputs and target values for the problem specified in question 4.

4. Decision boundary. Consider the problem in Table 1.

a) Illustrate the problem graphically. Explain whether or not it can be solved by a simple perceptron with three input units, and one output unit $O^{(\mu)} = \text{sgn}(\sum_{i=1}^{3} w_i x_i^{(\mu)} - \theta)$, where w_i is the weight from unit i to the output with threshold θ. (**0.5**p)

b) Show that this problem can be solved following the three steps below.

1. Transform the inputs $(x_1, x_2, x_3)^{\mathsf{T}}$ to two-dimensional coordinates $(g_1, g_2)^{\mathsf{T}}$ using the following functions:

$$g_1(\boldsymbol{x}^{(\mu)}) = \exp(-|\boldsymbol{x}^{(\mu)} - \boldsymbol{w}_1|^2), \text{ with } \boldsymbol{w}_1 = (0,0,0)^{\mathsf{T}} , \tag{8}$$
$$g_2(\boldsymbol{x}^{(\mu)}) = \exp(-|\boldsymbol{x}^{(\mu)} - \boldsymbol{w}_2|^2), \text{ with } \boldsymbol{w}_2 = (1,1,1)^{\mathsf{T}} . \tag{9}$$

   Here $\boldsymbol{x}^{(\mu)} = (x_1^{(\mu)}, x_2^{(\mu)}, x_3^{(\mu)})^{\mathsf{T}}$, and $|\cdots|$ denotes the norm of a vector. Plot the positions of the eight input patterns in the transformed space $(g_1, g_2)^{\mathsf{T}}$, encoding the different target outputs. (*To compute* $g_i(\boldsymbol{x}^{(\mu)})$ *use the following approximations:* $\exp(-1) \approx 0.37$, $\exp(-2) \approx 0.14$, $\exp(-3) \approx 0.05$.) (**0.5**p)

2. Use the transformed input data as inputs to a simple perceptron with $\text{sgn}(\cdots)$ activation function. In the plot you drew in the previous step, draw also a decision boundary that solves the problem when a simple perception is applied to the transformed data. (**0.5**p)

3. Compute the weight vector and the threshold for the simple perceptron corresponding to the decision boundary you drew in the previous step. (**0.5**p)

5. Training a multi-layer perceptron by gradient descent. To train a multi-layer perceptron by gradient descent one needs update formulae for weights and thresholds. Derive these update formulae for sequential training

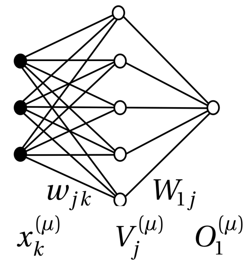

Figure 4: Question 5. Multi-layer perceptron with three input terminals, one hidden layer, and one output.

using backpropagation for the network shown in Fig. 4. The weights for the hidden layer are denoted by w_{jk}, and those for the output layer by W_{1j}. The corresponding thresholds are denoted by θ_j and Θ_1, and the activation function by $g(\cdots)$. The target value for input pattern $\boldsymbol{x}^{(\mu)}$ is $t_1^{(\mu)}$, and the pattern index μ ranges from 1 to p. The energy function is $H = \frac{1}{2}\sum_{\mu=1}^{p}(t_1^{(\mu)} - O_1^{(\mu)})^2$. (**2**p).

6. Number of parameters of a convolutional net. A convolutional net has the following layout (Fig. 5): an input layer of size $31 \times 31 \times 3$, a convolutional layer with ReLU activations with 10 kernels with local receptive fields of size 3×3, stride $(2, 2)$, and padding $= (0, 0, 0, 0)$, a max-pooling layer with local receptive field of size 5×5, stride $= (5, 5)$, padding $= (0, 0, 0, 0)$, a fully connected layer with 10 neurons with sigmoid activations, and a fully connected output layer with 5 neurons. In one or two sentences, explain the function of each of the layers. Specify the values of the parameters $x1, y1, z1, x2, \ldots, y5$ depicted in Fig. 5 and determine the number of trainable parameters (weights and thresholds) for the connections into each layer of the network. (**2**p)

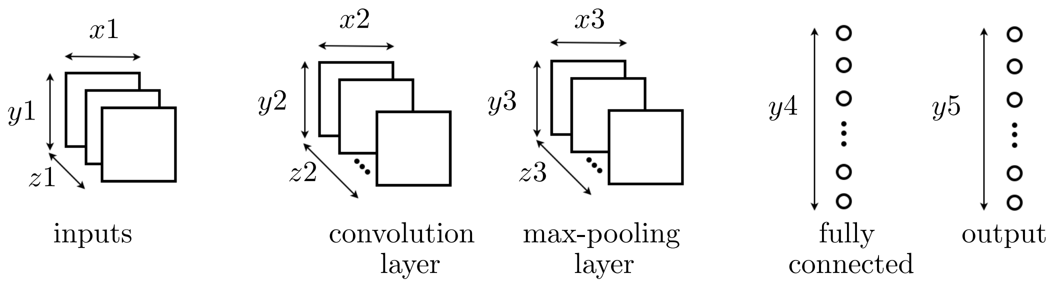

Figure 5: Question 6. Layout of convolutional net in Question 6.