CHALMERS, GÖTEBORGS UNIVERSITET

EXAM for
ARTIFICIAL NEURAL NETWORKS

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

Time:	October 27, 2023, at $14^{00} - 18^{00}$
Place:	Johanneberg
Teachers:	Bernhard Mehlig, 073-420 0988 (mobile)
	Teacher visits at 14^{30} and 17^{30}
Allowed material:	Book B. Mehlig, *Machine Learning with Neural Networks, CUP*
Not allowed:	Any other written material, calculator

Maximum score on this exam: 15 points.
Maximum score for homework problems: 9 points.
To pass the course it is necessary to score at least 6 points on this written exam.
CTH >13.5 passed; >17 grade 4; >21.5 grade 5,
GU >13.5 grade G; > 19.5 grade VG.

1. Hopfield model. Figure 1 shows a Hopfield model with three neurons s_1, s_2, and s_3, and symmetric weights w_{ij}.
(a) Write down the energy function for this network (assume that the thresholds are zero). *Hint*: the general form is $H = -\frac{1}{2}\sum_{ij} w_{ij} s_i s_j$. (**0.5**p).
(b) Show that the energy cannot increase if you update the first neuron using $s_1' = \text{sgn}(\sum_j w_{1j} s_j)$. (**1.5**p).
(c) Now consider a synchronous update, updating all neurons at the same time with $s_i' = \text{sgn}(\sum_j w_{ij} s_j)$. Show that the energy can increase. (**1**p).

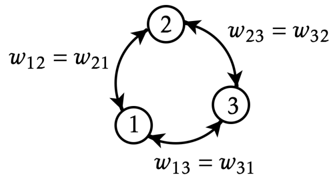


Figure 1: Hopfield model with three neurons. Question 1.

Solution: (a)

$$\begin{aligned} H &= -\tfrac{1}{2}\sum_{ij} w_{ij}s_is_j \\ &= -\tfrac{1}{2}\Big[w_{12}s_1s_2 + w_{13}s_1s_3 + w_{23}s_2s_3 + w_{32}s_2s_1 + w_{31}s_3s_1 + w_{32}s_3s_2\Big] \\ &= -\Big[w_{12}s_1s_2 + w_{13}s_1s_3 + w_{23}s_2s_3\Big]\,. \end{aligned}$$

(b) We use the update rule $s_1' = \mathrm{sgn}(w_{12}s_2 + w_{13}s_3)$. Let

$$H' = -\Big[w_{12}s_1's_2 + w_{13}s_1's_3 + w_{23}s_2s_3\Big]\,.$$

We consider two cases. First, if $s_1' = s_1$ then $H' = H$. Second, if $s_1' = -s_1$ then

$$\begin{aligned} H' - H &= \Big[w_{12}s_1s_2 + w_{13}s_1s_3 - w_{23}s_2s_3\Big] + \Big[w_{12}s_1s_2 + w_{13}s_1s_3 + w_{23}s_2s_3\Big] \\ &= 2w_{12}s_1s_2 + 2w_{13}s_1s_3 = 2s_1(w_{12}s_2 + w_{13}s_3) < 0\,, \end{aligned}$$

since the update rule implies that $w_{12}s_2 + w_{13}s_3$ has the same sign as s_1', opposite of s_1. We conclude: the energy cannot increase.

(c) Now consider

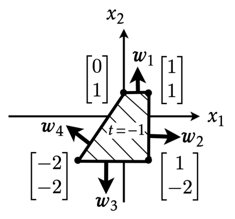
$$H' = -\Big[w_{12}s_1's_2' + w_{13}s_1's_3' + w_{23}s_2's_3'\Big]\,.$$

Assume $s_1' = -s_1, s_2' = -s_2, s_3' = s_3$. In this case

$$H' - H = 2s_3(w_{13}s_1 + w_{23}s_2) > 0\,,$$

since the update rule implies that $w_{13}s_1 + w_{23}s_2 = w_{31}s_1 + w_{32}s_2$ has the same sign as s_3', the same as s_3. So in this case the energy can increase.

2. Linearly inseparable problem. Figure 2 shows a classification problem where input patterns $\boldsymbol{x}^{(\mu)}$ inside the hashed region have targets $t^{(\mu)} = -1$ and patterns outside of the hashed region have targets $t^{(\mu)} = 1$. Design a fully-connected neural network with two inputs, a hidden layer with M neurons, and an output layer with one neuron that solves the classification problem. Use $g(b) = \mathrm{sgn}(b)$ for all neurons. Denote the weights leading from the input to the hidden layer by w_{ij}, the thresholds of the hidden layer by θ_i, the weights leading from the hidden layer to the output layer by W_i and the threshold of the output neuron by Θ. Orient the hidden weight vectors $\boldsymbol{w}_i$ as shown in Figure 2. Clearly write down all the parameter values chosen for the network. (**2**p).



x_2
x_1
w_1
w_2
w_3
w_4
$t=-1$
$\begin{bmatrix}0\\1\end{bmatrix}$
$\begin{bmatrix}1\\1\end{bmatrix}$
$\begin{bmatrix}-2\\-2\end{bmatrix}$
$\begin{bmatrix}1\\-2\end{bmatrix}$

Figure 2: Classification problem for question 2. Targets in the hashed region (▨) are $t = -1$, outside $t = 1$.

Solution: The decision boundaries are given by the following weights and thresholds:

$$\boldsymbol{w}_1 = \begin{bmatrix}0\\1\end{bmatrix}, \theta_1 = 1; \boldsymbol{w}_2 = \begin{bmatrix}1\\0\end{bmatrix}, \theta_2 = 1; \boldsymbol{w}_3 = \begin{bmatrix}0\\-1\end{bmatrix}, \theta_3 = -2; \boldsymbol{w}_3 = \begin{bmatrix}-3\\2\end{bmatrix}, \theta_4 = 1\,.$$

The network output should evaluate to unity for all hidden-neuron states, except when all $V_j = -1$. In that case the network output should be -1. A possible choice of output weights and threshold is: $W_1 = W_2 = W_3 = W_4 = 1$ and $\Theta = -3$.

3. Backpropagation. Figure 3 shows an autoencoder with a bottleneck that has just one single neuron that computes the latent variable $z = g(b)$ with $b = \boldsymbol{w} \cdot \boldsymbol{x} - \theta$. The outputs compute $O_i = g(B_i)$ with $B_i = W_i z - \Theta_i$. Derive the learning rules for the bottleneck weights $\boldsymbol{w}$ and threshold θ using the energy function $H = \frac{1}{2}\sum_i (x_i - O_i)^2$. (**2**p).

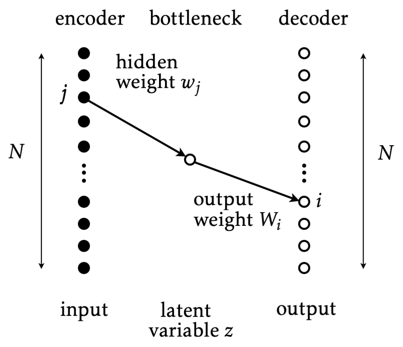


Figure 3: Network layout for question 3. Not all connections are shown.

Solution: The energy function reads $H = \frac{1}{2}\sum_i (x_i - O_i)^2$. The neurons calculate

$$z = g(b) \quad \text{with} \quad b = \sum_j w_j x_j - \theta$$

$$O_i = g(B_i) \quad \text{with} \quad B_i = W_i z - \Theta_i\,.$$

To derive the learning rule for the hidden weights, we use $\delta w_j = -\eta \partial H / \partial w_j$. We have

$$\begin{aligned}
\frac{\partial H}{\partial w_j} &= -\sum_i (x_i - O_i) \frac{\partial O_i}{\partial w_j} \\
\frac{\partial O_i}{\partial w_j} &= g'(B_i)\frac{\partial B_i}{\partial w_j} = g'(B_i) W_i \frac{\partial z}{\partial w_j} = g'(B_i) W_i g'(b) \frac{\partial b}{\partial w_j} \\
&= g'(B_i) W_i g'(b) x_j\,.
\end{aligned}$$

This gives

$$\delta w_j = \eta \sum_i (x_i - O_i) g'(B_i) W_i g'(b) x_j\,.$$

In an analogous way one obtains the learning rule for the threshold:

$$\delta\theta = -\eta \sum_i (x_i - O_i) g'(B_i) W_i g'(b)\,.$$

4. Feature map. Figure 4 shows two patterns. Design a convolutional network with one convolution layer with one single 3×3 kernel with ReLU activation function (equal to zero for $b < 0$ and equal to b for $b \geq 0$), zero threshold, and stride [1,1]. The resulting feature map is fed into a 3×3 max-pooling layer with stride [1,1]. Finally there is a fully connected output layer with a single output neuron with Heaviside activation function.

Find suitable weights of the feature map, as well as weights and threshold of the output neuron that together allow the network to distinguish the digits, by assigning output 1 to the input "2", and output 0 to input "8". For both patterns, determine the resulting feature map, the output of the max-pooling layer, and the network output. (**3**p).

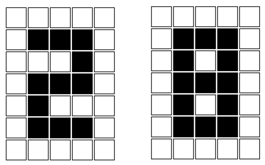


Figure 4: Input patterns for question 4.

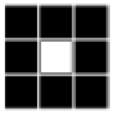


Figure 5: Kernel used for solution of question 4, where ■ corresponds to a unit weight, and □ to a zero weight.

Solution: A possible choice for the 3×3 kernel is shown in Figure 5. Using this kernel together with the ReLU activation function, we obtain the following feature maps for the two input patterns:

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 7 & 4 \\ 2 & 4 & 2 \\ 4 & 7 & 4 \\ 2 & 3 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 2 & 4 & 2 \\ 4 & 8 & 4 \\ 3 & 6 & 3 \\ 4 & 8 & 4 \\ 2 & 4 & 2 \end{bmatrix} .$$

Next, applying the max-pooling layer yields

$$\begin{bmatrix} 7 \\ 7 \\ 7 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 8 \\ 8 \\ 8 \end{bmatrix} .$$

Nownset the output weights to

$$\boldsymbol{W} = \begin{bmatrix} -1 & -1 & -1 \end{bmatrix} ,$$

and the output threshold to $\Theta = -22$. Then the network output evaluates to 1 when feeding the pattern '2', and to 0 upon feeding the pattern '8'.

5. Recurrent network. Figure 6 shows a simple recurrent network with one hidden neuron $V(t)$, one input $x(t)$ and one output $O(t)$. The network learns a time series of input-output pairs $[x(t), y(t)]$ for $t = 1, 2, 3, \ldots, T$. Here t is a discrete time index and $y(t)$ is the target value at time t (the targets are denoted by y to avoid confusion with the time index t). The hidden unit is initialised to a value $V(0)$ at $t = 0$. This network can be trained by backpropgation by *unfolding it in time*.

(a) Draw the unfolded network, and label the connections using the labels shown in Figure 6. (**0.5**p).

(b) Write down the dynamical rules for this network, the rules that determine $V(t)$ in terms of $V(t-1)$ and $x(t)$, as well as $O(t)$ in terms of $V(t)$. Assume that both $V(t)$ and $O(t)$ have the same activation function $g(b)$. (**0.5**p).

(c) Derive the learning rule for $w^{(ov)}$ for gradient descent on the energy function

$$H = \frac{1}{2} \sum_{t=1}^{T} E(t)^2 \quad \text{where } E(t) = y(t) - O(t) . \tag{1}$$

Denote the learning rate by η. (**1**p).
(d) Derive the learning rule for $w^{(vx)}$. (**1**p).

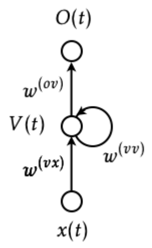


Figure 6: Recurrent network, question 5.

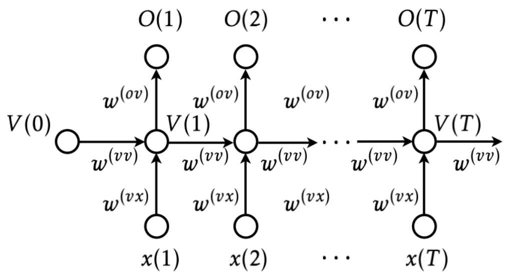


Figure 7: Unfolded network, question 5.

Solution: (a) The unfolded network is drawn in Figure 7.

(b) The dynamical rules are

$$V(t) = g\big(w^{(vv)}V(t-1) + w^{(vx)}x(t) - \theta^{(v)}\big)\,, \tag{2a}$$
$$O(t) = g\big(w^{(ov)}V(t) - \theta^{(o)}\big) \tag{2b}$$

for $t = 1, 2, \ldots$.

(c) Gradient descent using (1) yields

$$\delta w^{(ov)} = \eta \sum_{t=1}^{T} E(t) \frac{\partial O(t)}{\partial w^{(ov)}} = \eta \sum_{t=1}^{T} E(t) g'(B(t)) V(t) = \eta \sum_{t=1}^{T} \Delta(t) V(t)\,, \tag{3}$$

where $\Delta(t) = E(t)g'(B(t))$ is the output error, $B(t) = w^{(ov)}V(t-1) - \theta^{(o)}$ is the local field of the output neuron at time t, and we used Equation (2b).

(d) Gradient descent using (1) yields

$$\delta w^{(vx)} = \eta \sum_{t=1}^{T} E(t) \frac{\partial O(t)}{\partial w^{(vx)}} = \eta \sum_{t=1}^{T} \Delta_t w^{(ov)} \frac{\partial V(t)}{\partial w^{(vx)}} \,. \tag{4}$$

Now evaluate the derivative $\partial V(t)/\partial w^{(vx)}$. Equation (2a) yields the recursion

$$\frac{\partial V(t)}{\partial w^{(vx)}} = g'(b(t)) \Big[x(t) + w^{(vv)} \frac{\partial V(t-1)}{\partial w^{(vx)}} \Big] \tag{5}$$

for $t \geq 1$. Since $\partial V(0)/\partial w^{(vv)} = 0$, Equation (5) implies:

$$\begin{aligned}
\frac{\partial V(1)}{\partial w^{(vv)}} &= g'(b(1))x(1)\,,\\
\frac{\partial V(2)}{\partial w^{(vv)}} &= g'(b(2))x(2) + g'(b(2))w^{(vv)}g'(b(1))x(1)\,,\\
&\vdots\\
\frac{\partial V(T-1)}{\partial w^{(vv)}} &= g'(b(T-1))x(T-1) + g'(b(T-1))w^{(vv)}g'(b(T-2))x(T-2) + \ldots\\
\frac{\partial V(T)}{\partial w^{(vv)}} &= g'(b(T))x(T) + g'(b(T))w^{(vv)}g'(b(T-1))x(T-1) + \ldots
\end{aligned}$$

The terms in this sum can be regrouped as described on p. 164 in the course book. Defining the errors as

$$\delta(t) = \begin{cases} \Delta(T) w^{(ov)} g'(b(T)) & \text{for } t = T,\\ \Delta(t) w^{(ov)} g'(b(t)) + \delta(t+1) w^{(vv)} g'(b(t)) & \text{for } 0 < t < T, \end{cases} \tag{6}$$

one can write the learning rule in the usual way:

$$\delta w^{(vx)} = \eta \sum_t \delta(t) x(t)\,. \tag{7}$$

Note that the sum involves x evaluated at t while the learning rule for $\delta w^{(vv)}$ involves V evaluated at $t-1$, consistent with Equation (2a).

6. Free energy of the Hopfield model. The free energy of the Hopfield model is defined as

$$F(\beta) = -\frac{1}{\beta}\log Z \quad \text{with} \quad Z = \sum_{\boldsymbol{s}} \mathrm{e}^{-\beta H(\boldsymbol{s})} \quad \text{and} \quad H(\boldsymbol{s}) = -\tfrac{1}{2}\sum_{i,j} w_{ij} s_i s_j \tag{8}$$

The s_j take values ± 1. In mean-field theory, one approximates the energy function $H(\boldsymbol{s})$ using

$$s_i s_j \approx s_i \langle s_j \rangle + \langle s_i \rangle s_j - \langle s_i \rangle \langle s_j \rangle\,. \tag{9}$$

where the average $\langle \cdots \rangle$ is over the Boltzmann distribution. (a) Use Hebb's rule $w_{ij} = N^{-1}\sum_\mu x_i^{(\mu)} x_j^{(\mu)}$ to write $H(\boldsymbol{s})$ as a function of the order parameters $m_\mu = N^{-1}\sum_j x_j^{(\mu)}\langle s_j\rangle$. Here N is the number of neurons in the network. *Hint*: the result is a linear function of s_i, with a constant term that does not depend on s_i, and a term linear in s_i. (**1**p).

(b) Using this result, compute the free energy by averaging over $s_i = \pm 1$ with the Boltzmann distribution. (**1**p).

Solution: (a) Using Hebb's rule and Equation (9), we find that the energy function becomes

$$\begin{aligned}
H(s) = -\frac{1}{2}\sum_{i,j}\frac{1}{N}\sum_\mu x_i^{(\mu)}x_j^{(\mu)} s_i s_j &\approx -\frac{1}{2}\sum_{i,j}\frac{1}{N}\sum_\mu x_i^{(\mu)}x_j^{(\mu)} s_i\langle s_j\rangle \\
&\quad -\frac{1}{2}\sum_{i,j}\frac{1}{N}\sum_\mu x_i^{(\mu)}x_j^{(\mu)}\langle s_i\rangle s_j + \frac{1}{2}\sum_{i,j}\frac{1}{N}\sum_\mu x_i^{(\mu)}x_j^{(\mu)}\langle s_i\rangle\langle s_j\rangle \\
&= -\frac{1}{2}\sum_{i,\mu} x_i^{(\mu)} s_i m_\mu - \frac{1}{2}\sum_{j,\mu} x_j^{(\mu)} s_j m_\mu + \frac{N}{2}\sum_\mu m_\mu m_\mu \\
&= \frac{N}{2}\sum_\mu m_\mu m_\mu - \sum_{i,\mu} x_i^{(\mu)} s_i m_\mu . \qquad (10)
\end{aligned}$$

(b) Starting from the definition (8) of the partition function, we have

$$\begin{aligned}
Z = \sum_{\boldsymbol{s}} e^{-\beta H(\mathbf{s})} &\approx \sum_{\boldsymbol{s}} e^{-\beta\frac{N}{2}\sum_\mu m_\mu^2 + \beta\sum_{i,\mu} m_\mu x_i^{(\mu)} s_i} \\
&= e^{-\frac{\beta N}{2}\sum_\mu m_\mu^2}\sum_{\boldsymbol{s}} e^{\beta\sum_{i,\mu} m_\mu x_i^{(\mu)} s_i} . \qquad (11)
\end{aligned}$$

The sum over $\boldsymbol{s}$ includes all combinations $s_1 = \pm 1, \ldots s_N = \pm 1$. The next step is to rewrite the factor that contains this sum in the following way

$$\sum_{\boldsymbol{s}}\prod_i e^{\beta\sum_\mu m_\mu x_i^{(\mu)} s_i} . \qquad (12)$$

Now we can evaluate the sum over $\boldsymbol{s}$:

$$\sum_{\boldsymbol{s}}\prod_i e^{\beta\sum_\mu m_\mu x_i^{(\mu)} s_i} = \prod_i\sum_{s_i=\pm 1} e^{\beta\sum_\mu m_\mu x_i^{(\mu)} s_i} = \prod_i 2\cosh\big(\sum_\mu m_\mu x_i^{(\mu)}\big) . \qquad (13)$$

Inserting this expression into Equation (11) gives

$$F(\beta) = -\frac{1}{\beta}\log Z = \frac{N}{2}\sum_\mu m_\mu^2 - \frac{1}{\beta}\sum_i \log[2\cosh\big(\sum_\mu m_\mu x_i^{(\mu)}\big)] . \qquad (14)$$

See the solution to Exercise 3.1 to learn about the significance of the free energy.

Errata for *Machine learning with neural networks*
Bernhard Mehlig, Cambridge University Press (2021)

p. 32	l. 3	‘$\partial H/\partial s_m$’ should be replaced by ‘$-\partial H/\partial s_m$’.
p. 32	l. 11	‘$w_{ii} > 0$’ should be replaced by ‘$w_{ii} = 0$’.
p. 32	l. 21	should read: ‘$H = -\frac{1}{2}\sum_{ij} w_{ij} g(b_i) g(b_j) + \sum_i \theta_i g(b_i) + \int_0^{b_i} \mathrm{d}b\, b g'(b)$, with $b_i = \sum_j w_{ij} n_j - \theta_i$, cannot increase…’.
p. 37	l. 16	replace ‘$\sqrt{N}$’ by ‘$N^{-1/2}$’.
	l. 17	replace ‘$\langle b_i(t)\rangle \sim N$’ by ‘$\langle b_i(t)\rangle = O(1)$’.
p. 48	eq. (3.46)	replace ‘$\langle n_i\rangle$’ by ‘$\langle s_i\rangle$’.
p. 54	eq. (4.5c)	replace ‘$-\beta b_m$’ by ‘$2\beta b_m$’.
p. 55	eq. (4.5d)	replace ‘βb_m’ by ‘$-2\beta b_m$’.
p. 61	eq. (4.18)	the sum should be over *distinct* patterns $\boldsymbol{x}$.
p. 67	alg. 3	add superscripts ‘(μ) ’ to ‘δw_{mn}’, ‘$\delta\theta_n^{(\mathrm{v})}$’, and ‘$\delta\theta_n^{(\mathrm{h})}$’.
p. 72	l. 12	the list should read ‘$1, 2, 4$, and 8’.
p. 85	fig. 5.11	switch the labels ‘10’ and ‘50’.
p. 86	fig. 5.12	permute the axis labels clockwise for consistency with fig. 5.8.
p. 93	fig. 5.22	switch the labels ‘1111’ and ‘1101’ in the right panel.
p. 97	eq. (6.6a)	insert ‘$V_n^{(\mu)}$’ before the ‘$\equiv$’ sign.
p. 106	l. 18	should read ‘a compromise, reducing the tendency of the network to overfit at the expense of training accuracy’.
p. 117	fig. 7.5	the hidden neurons should be labeled ‘$j = 0, 1, 2, 3$’ from bottom to top.
p. 118	fig. 7.6	exchange labels ‘1’ and ‘2’.
	eq. (7.9)	should read ‘$O_1 = \mathrm{sgn}(-V_0 + V_1 + V_2 - V_3)$’.
p. 121	fig. 7.10	change ‘$w^{(L-2)}$’ to ‘$w^{(L)}$’.
p. 122	eq. (7.17)	replace ‘$\mathbb{J}$’ by ‘$\mathbb{J}'$’, also in the two lines above the equation.
p. 123	eq. (7.19)	should read ‘$[\boldsymbol{\delta}^{(\ell)}]^{\mathsf{T}} = [\boldsymbol{\delta}^{(L)}]^{\mathsf{T}}\mathbb{J}_{L-\ell}$ with $\mathbb{J}_{L-\ell} = [\mathbb{D}^{(L)}]^{-1}\mathbb{J}'_{L-\ell}\mathbb{D}^{(\ell)}$’.
p. 131	eq. (7.45)	replace ‘O_l’ by ‘O_i’.
p. 139	l. 33	replace ‘the Lagrangian (7.57)’ by ‘$\frac{1}{2}\delta\boldsymbol{w}\cdot\mathbb{M}\delta\boldsymbol{w}$’.
p. 160	l. 15	delete ‘then $L_{ij} = \delta_{ij}$. In this case’.
p. 161	l. 19	replace ‘negative’ by ‘positive’, and ‘positive’ by ‘negative’ in the next line.
p. 171	l. 23	the upper limit of the second summation should be ‘M’.
p. 197	alg. 10	replace ‘$s_j = 0$’ by ‘$s_j = 1$’ in line 2 of Algorithm 10.
p. 202	l. 37	replace ‘positive’ by ‘non-negative’.
p. 203	l. 21	should read ‘Alternatively, assume that $\boldsymbol{w}^* = u + \mathrm{i}v$ can be written as an analytic function of $\boldsymbol{r} = r_1 + \mathrm{i}r_2\ldots$’.
	l. 27	add ‘See Ref. [2]’.
p. 204	l. 5	replace ‘$\sin(2\pi x_1)$’ by ‘$\sin(\pi x_1)$’. Same in caption of fig. 10.17.
p. 225	l. 5,6	replace ‘two’ by ‘two (three)’ and ‘lost’ by ‘lost (drew)’.

Gothenburg, October 20 (2023)