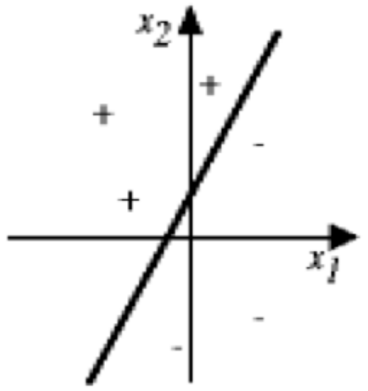
# Lab 5: 16th April 2012

## Exercises on Neural Networks

1. What are the values of weights $w_0$, $w_1$, and $w_2$ for the perceptron whose decision surface is illustrated in the figure? Assume the surface crosses the $x_1$ axis at -1 and the $x_2$ axis at 2.



**Solution**

The output of the perceptron is

$$o = sgn(w_0 + w_1 x_1 + w_2 x_2)$$

The equation of the decision surface (the line) is

$$w_o + w_1 x_1 + w_2 x_2 = 0$$

We know the coordinates of 2 points of this line: A=(-1,0) and B=(0,2). Therefore, the equation of the line is

$$\frac{x_1 - x_{1A}}{x_{1B} - x_{1A}} = \frac{x_2 - x_{2A}}{x_{2B} - x_{2A}} \quad \rightarrow \quad \frac{x_1 - (-1)}{0 - (-1)} = \frac{x_2 - 0}{2 - 0} \quad \rightarrow \quad x_1 + 1 = \frac{x_2}{2} \quad \rightarrow \quad 2 + 2x_1 - x_2 = 0$$

So 2, 2, -1 are possible values for the weights $w_0$, $w_1$, and $w_2$ respectively. To check if their signs are correct, consider a point on one side of the line, for instance the origin O=(0,0). The output of the perceptron for this point has to be negative, but the output of the perceptron using the candidate weights is positive. Therefore, we need to negate the previous values and conclude that $w_0 = -2, w_1 = -2, w_2 = 1$.

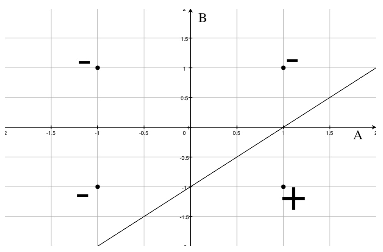
2. (a) Design a two-input perceptron that implements the Boolean function A∧¬B. (b) Design the two-layer network of perceptrons that implements A XOR B.

**Solution (a)**

The requested perceptron has 3 inputs: A, B, and the constant 1. The values of A and B are 1 (true) or -1 (false). The following table describes the output O of the perceptron:

| A | B | O = A∧¬B |
|---|---|---|
| -1 | -1 | -1 |
| -1 | 1 | -1 |
| 1 | -1 | 1 |
| 1 | 1 | -1 |

One of the correct decision surfaces (any line that separates the positive point from the negative points would be fine) is shown in the following picture.



The line crosses the A axis at 1 and the B axis -1. The equation of the line is

$$\frac{A-0}{1-0}=\frac{B-(-1)}{0-(-1)} \rightarrow A = B+1 \rightarrow 1-A+B=0$$

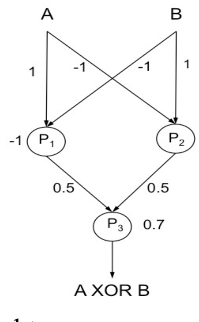
So 1, -1, 1 are possible values for the weights $w_0$, $w_1$, and $w_2$, respectively. Using this values the output of the perceptron for A=1, B=-1 is negative. Therefore, we need to negate the weights and therefore we can conclude that $w_0=-1, w_1=1, w_2=-1$.

**Solution (b)**

A XOR B cannot be calculated by a single perceptron, so we need to build a two-layer network of perceptrons. The structure of the network can be derived by:

- Expressing A XOR B in terms of other logical connectives:
  A XOR B = (A∧¬B) ∨(¬A∧B)
- Defining the perceptrons $P_1$ and $P_2$ for (A∧¬B) and (¬A∧B)
- Composing the outputs of $P_1$ and $P_2$ into a perceptron $P_3$ that implements o($P_1$) ∨ o($P_2$)

Perceptron $P_1$ has been defined above. $P_2$ can be defined similarly. $P_3$ is defined in the course slides[1]. In the end, the requested network is the following:



NB. The number close to each unit is the weight $w_0$.

[1] It is defined for 0/1 input values, but it can be easily modified for -1/+1 input values.

3. Consider two perceptrons A and B defined by the threshold expression $w_0+w_1x_1+w_2x_2>0$. Perceptron A has weight values $w_0=1$, $w_1=2$, $w_2=1$ and perceptron B has weight values $w_0=0$, $w_1=2$, $w_2=1$. Is perceptron A *more_general_than* perceptron B? A is *more_general_than* B if and only if $\forall$ instance $<x_1,x_2>$, $B(<x_1,x_2>)=1 \rightarrow A(<x_1,x_2>)=1$.

**Solution**

$B(<x_1,x_2>) = 1 \rightarrow 2x_1+x_2 > 0 \rightarrow 1+2x_1+x_2 > 0 \rightarrow A(<x_1,x_2>) = 1$

4. Derive a gradient descent training rule for a single unit with output *o*, where

$$o=w_0+w_1x_1+w_1x_1^2+\ldots+w_nx_n+w_nx_n^2$$

**Solution**

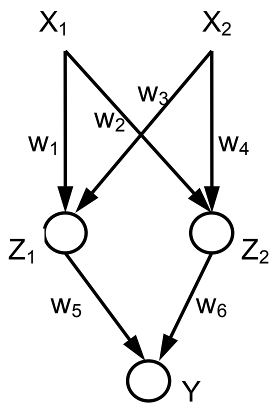
The gradient descent training rule specifies how the weights are to be changed at each step of the learning procedure so that the prediction error of the unit decreases the most. The derivation of the rule for a linear unit is presented on pages 91-92 of the Mitchell, and on pages 4-6 of the course slides (*ml_2012_lecture_07*). We can adapt that derivation and consider the output *o*.

$$\frac{\partial E}{\partial w_i} = \sum_{\mathbf{x}\in X}\left(out_{\mathbf{x}} - o_{\mathbf{x}}\right)\frac{\partial}{\partial w_i}\left(out_{\mathbf{x}} - \left(\mathrm{w}_0+\mathrm{w}_1\mathrm{x}_{1\mathbf{x}}+\mathrm{w}_1\mathrm{x}_{1\mathbf{x}}^2+\ldots+\mathrm{w}_n\mathrm{x}_{n\mathbf{x}}+\mathrm{w}_n\mathrm{x}_{n\mathbf{x}}^2\right)\right) = \sum_{\mathbf{x}\in X}\left(out_{\mathbf{x}} - o_{\mathbf{x}}\right)\left(-x_{i\mathbf{x}} - x_{i\mathbf{x}}^2\right)$$

Therefore, the gradient descent training rule is

$$\frac{\partial E}{\partial w_i} = \sum_{\mathbf{x}\in X}\left(out_{\mathbf{x}} - o_{\mathbf{x}}\right)\frac{\partial}{\partial w_i}\left(out_{\mathbf{x}} - \left(\mathrm{w}_0+\mathrm{w}_1\mathrm{x}_{1\mathbf{x}}+\mathrm{w}_1\mathrm{x}_{1\mathbf{x}}^2+\ldots+\mathrm{w}_n\mathrm{x}_{n\mathbf{x}}+\mathrm{w}_n\mathrm{x}_{n\mathbf{x}}^2\right)\right) = \sum_{\mathbf{x}\in X}\left(out_{\mathbf{x}} - o_{\mathbf{x}}\right)\left(-x_{i\mathbf{x}} - x_{i\mathbf{x}}^2\right)$$

5. Consider a two-layer feed-forward neural network that has the topology shown in the figure.
   - $X_1$ and $X_2$ are the two inputs.
   - $Z_1$ and $Z_2$ are the two hidden neurons.
   - $Y$ is the (single) output neuron.
   - $w_i$, i=1..4, are the weights of the connections from the inputs to the hidden neurons.
   - $w_j$, j=5..6, are the weights of the connections from the hidden neurons to the output neuron.



Explain the three phases (i.e., input signal forward, error signal backward, and weight update) of <u>**the first training iteration**</u> of the Backpropagation algorithm for the current network, given the training example: ($X_1$=x$_1$, $X_2$=x$_2$, $Y$=y). Please use the following notations for the explanation.

- $Net_1$, $Net_2$, and $Net_3$ are the net inputs to the $Z_1$, $Z_2$, and $Y$ neurons, respectively.
- $o_1$, $o_2$, and $o_3$ are the output values for the $Z_1$, $Z_2$, and $Y$ neurons, respectively.
- $f$ is the activation function used for every neuron in the network, i.e., $o_k=f(Net_k)$, k=1..3.
- $E(\mathbf{w}) = (y - o_3)^2 / 2$ is the error function, where $y$ is the desired network output.
- $\eta$ is the learning rate
- $\delta_1$, $\delta_2$, and $\delta_3$ are the error signals for the $Z_1$, $Z_2$, and $Y$ neurons, respectively.

## Solution

*Propagate the input forward through the network*

1. Input the instance ($x_1$,$x_2$) to the network and compute the network outputs $o_3$

- $Net_1= w_1x_1+ w_2x_2 \rightarrow o_1=f(Net_1)$
- $Net_2= w_3x_1+ w_4x_2 \rightarrow o_2=f(Net_2)$
- $Net_3= w_5 f(Net_1)+w_6 f(Net_2) \rightarrow o_3=f(w_5 f(Net_1)+w_6 f(Net_2))$

*Propagate the error backward through the network*

- $E(\mathbf{w}) = (y - o_3)^2 / 2 = (y-f(w_5 f(Net_1)+w_6 f(Net_2)))^2$

2. Calculate the error term of out unit Y

- $\delta_3=f'(f(w_5 f(Net_1)+w_6 f(Net_2)))*(y-f(w_5 f(Net_1)+w_6 f(Net_2)))$

3. Calculate the error of the 2 hidden units

- $\delta_2=f'(f(Net_2)\delta_3 \quad \delta_1=f'(f(Net_1)\delta_3$

4. Update the network weights

- $w_1 \leftarrow w_1+ \eta\delta_1 w_1$
- $w_2 \leftarrow w_2+ \eta\delta_2 w_2$
- $w_3 \leftarrow w_3+ \eta\delta_1 w_3$
- $w_4 \leftarrow w_4+ \eta\delta_2 w_4$
- $w_5 \leftarrow w_5+ \eta\delta_3 w_5$
- $w_6 \leftarrow w_6+ \eta\delta_3 w_6$

6. In the Back-Propagation learning algorithm, what is the object of the learning? Does the Back-Propagation learning algorithm guarantee to find the global optimum solution?

**Solution**

The object is to learn the weights of the interconnections between the inputs and the hidden units and between the hidden units and the output units. The algorithms attempts to minimize the squared error between the network output values and the target values of these outputs.
The learning algorithm does not guarantee to find the global optimum solution. It guarantees to find at least a local minimum of the error function.