# Informatics 1 Cognitive Science – Tutorial 2

Frank Keller, Carina Silberer, Frank Mollica

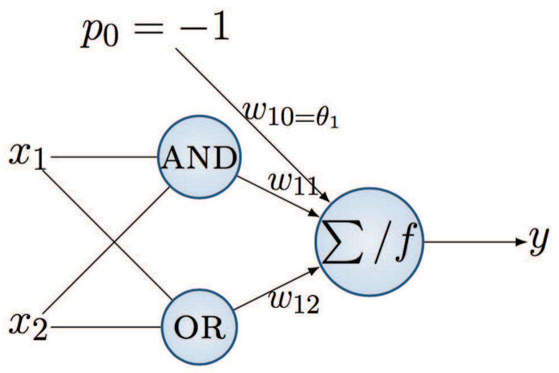
Week 3

The goal of this tutorial is to start building an in-depth understanding of how perceptrons work and the backpropagation algorithm used for training multilayer perceptrons. Before working on the exercises, you should make sure that you have revised the slides for lectures 5 and 6.

## 1 Perceptrons

In the lecture you learned that the primitive boolean functions AND and OR can be represented by a single perceptron each. You also saw that a single perceptron does not have the representational power to represent the boolean function XOR. A network of perceptrons two levels deep can in fact represent every boolean function.

**Exercise 1** Consider the two-level network illustrated below. It is composed of three perceptrons. The two perceptrons of the first level implement the AND and OR functions, respectively.



Determine the weights $w_{11}$, $w_{12}$ and threshold $\theta_1 (= w_{10})$ such that the network implements the XOR function. The initial weights are set to zero, i.e., $w_{11} = w_{12} = w_{10} = 0$, and the learning rate $\eta$ is set to 0.1 (Feel free to choose other initial values for $w_{1.}$ and $\eta$).

### Notes

- The input function for the perceptron on level $j$ is the weighted sum $\sum$ of its input.

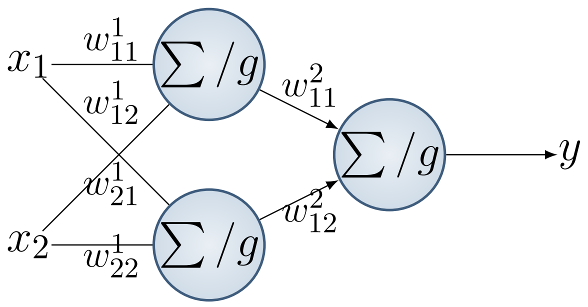
- The activation function $f$ for a perceptron is a step function:

$$f = \begin{cases} 1 & if \sum > 0 \\ 0 & otherwise \end{cases}$$

- The threshold $\theta$ is considered as a weight, with input $p_0 = -1$.
- Assume that the weights for the perceptrons of the first level are given, meaning that these perceptrons already compute AND and OR correctly and their weights do not change.

## 2 Backpropagation in Multilayer Perceptrons

**Exercise 2** Consider the following multilayer perceptron.



The network should implement the XOR function. Perform one epoch of backpropagation as introduced in the lecture on multilayer perceptrons. That is, for each training example:

1. Compute the activations of the hidden and output neurons
2. Compute the error of the network; backpropagate the error to determine $\Delta w_{ij}$ for all weights $w_{ij}$
3. Update the weight $w_{ij}$

What was is error before and after updating the weights through one round of training (or epoch)?

We will break this into several steps below. It will be very time consuming to do the computations for backpropatgation by hand. Instead, you can use the Excel spreadsheet provided with the tutorial, or implement backpropagation in a programming language of your choice.

1. Write down the equations for the output of each unit.
2. Write down the equation for the error of the network based on its output using the MSE we discussed in lecture. We don't use this directly until we're reporting the error at the end – in updating the weights we want the gradient of the error.

3. Write down the derivative of the activation function (you don't have to derive this by hand; it's in the slides).

4. Write down the formulas for the changes in the weights ($\Delta w$), and the $\delta$ terms they rely on.

5. Compute the error for the initial weights. As noted above, use the spreadsheet to do this. Otherwise it's painful.

6. Compute the values for $\Delta w$ for all weights.

7. Compute the new error after updating the weights.

**Notes**

- The activation function $g$ for this perceptron is the sigmoid function: $g(x) = \frac{1}{1+e^{-x}}$.

- The thresholds are not shown in the network. The threshold nodes are set to $-1$, and their weights are denoted with $\theta$.

- Use the following initial parameter values, where each row corresponds to the unit that each weight/bias feeds into:

$$
\begin{array}{lll}
w^1_{11} = 6 & w^1_{12} = 8 & \theta^1_1 = 2 \\
w^1_{21} = -6 & w^1_{22} = -8 & \theta^1_2 = -1 \\
w^2_{11} = 6 & w^2_{12} = -6 & \theta^2 = -2
\end{array}
$$

- The learning rate is set to $\eta = 0.7$.