

# Computational Intelligence WS1718

## Introduction to Artificial Neural Networks

Max Springenberg, 177792

### 1 Abstraction And Models

A neural net tries to replicate the behaviour of a neuron.

A neuron consists of dendrites that carry the signal input, a nucleus/ cell body that processes the signal and a axon and synapse that carry out signals.

Now, there are different methods to replicate such behaviour.

#### 1.1 McCulloch-Pitts Neuron 1943

We take input values  $x_1, \dots, x_n$  with  $x_i \in \mathbb{B}, 0 < i \leq n$ , apply them to a function and use the output.

obviously such a "neuron" is either active or inactive.

the function "fires" a 1, whenever a threshold  $\Theta$  is reached.

therefore gates like OR and AND, as well as NOT can be replicated with such neurons.

In consequence all logical functions can be replicated, using such neurons.

Weighted Neural Nets are NN, that multiply input values, before handing them to the function. Those are equivalent, since for every weighted net a unweighted net, that takes the input values multiple times and adjust the threshold, can be constructed.

##### 1.1.1 conclusion

PROS	CONS
feed forward (computes any logical /boolean function)	almost identical to logical circuits
recursive (can simulate any DFA)	difficulties in construction
	no good learning algorithm available

## 1.2 The Perceptron, Rosenblatt 1958 Minsky-Papert 1969

The perceptron by Rosenblatt is a rather complex model, that has been reduced by Minsky and Papert to it's core neccassities, resulting in the Minsky-Papert perceptron.

The essential difference of the MPP to the MCP is, that its input values are within  $\mathbb{R}$ , rather than  $\mathbb{B}$

### 1.2.1 What can a single MPP do?

Let  $W$  be the set of weights applied to the input signals, with  $W \subseteq \mathbb{R}$   
The function of a MPP with binary output can be defined as

$$f(W, X) = (w_1 * x_1 + \dots + w_n * x_n \geq \Theta)$$

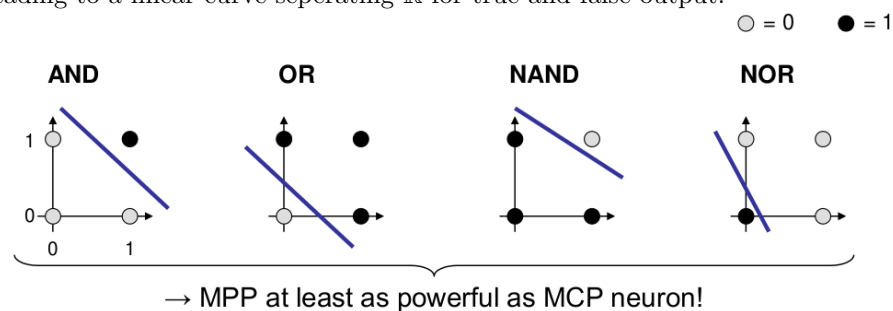
Let us isolate an  $x_i$

$$x_i = \frac{\Theta - (w_1 * x_1 + \dots + w_{i-1} * x_{i-1} + w_{i+1} * x_{i+1} + \dots + w_n * x_n)}{w_i}$$

for  $n = 2$  we get

$$x_2 = \frac{\Theta - w_1 * x_1}{w_2}$$

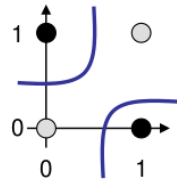
leading to a linear curve seperating  $\mathbb{R}$  for true and false output.



To solve the xor functions a nonlinear function and an additional "neuron" is required.

**XOR**

$$g(x_1, x_2) = 2x_1 + 2x_2 - 4x_1x_2 - 1 \quad \text{with} \quad \theta = 0$$



$$g(0,0) = -1$$

$$g(0,1) = +1$$

$$g(1,0) = +1$$

$$g(1,1) = -1$$