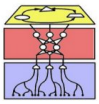

Universität Hamburg
Department Informatik, ABe NATS, WTM & TAMS
Wolfgang Menzel, Stefan Wermter, Jianwei Zhang

Kontakt: Cornelius Weber
Raum F-233
Tel 42883-2537

Exercises to the module: Algorithmisches Lernen

SS 2012 Sheet 5

Due: 9.05.2012

Note: the exercises of AL part II are optimised for the Python code of the "Machine Learning" book by Stephen Marsland. They may be solved by any software, such as:

- PyBrain: http://pybrain.org
- Emergent: http://grey.colorado.edu/emergent/index.php/Main_Page
- Encog: http://www.heatonresearch.com/encog
- LENS: http://tedlab.mit.edu/~dr/Lens

However, we have not yet tested them all. When choosing a tool, please make sure it supports Multi-Layer Perceptrons (MLP), Elman Networks (simple recurrent network), Kohonen's Self-Organizing Map (SOM).

Task 5.1

The following numbers characterise the two largest parts of the mouse brain, cerebral cortex and hippocampus:

- volume of cortex and hippocampus: 2 x 90 cubic millimeters
- cell density: 90,000 cells per cubic millimeter
- synapse density: 700,000,000 synapses per cubic millimeter
- total axon length per neuron: 40 millimeters
- range covered by the axons: 1 millimeter
- total dendrite length per neuron: 4 millimeters
- range covered by the dendrites: 0.2 millimeters

From these numbers, compute the following:

- number of neurons
- total axon length per cubic millimeter
- distance of cells between each other
- number of synapses per axon length
- number of synapses per dendrite length
- estimate the probability that two neighboring neurons share a synapse

The number of sensory fibres to these brain parts is smaller than 1 million in total. What does this tell about how cortex and hippocampus work?

— see reverse —

Wolfgang Menzel
(menzel@informatik...)
Stefan Wermter
(wermter@informatik...)
Jianwei Zhang
(zhang@informatik...)

Vorlesung:
Mi 8:15-11:45, F-009
Übungen:
Mi 12:15-13:45, F-009

Dienstgebäude
Universität Hamburg
Vogt-Kölln-Str. 30
22527 Hamburg

Öffentliche Verkehrsmittel
Buslinien 181, 281
Haltestelle: Informatikum

Task 5.2 Perzeptron and XOR-Classification

To classify the XOR problem successfully with a perceptron, extend the XOR data by a third input dimension. The value of this new x_3 coordinate may be computed from the x_1 and x_2 values as follows:

1. $x_3 = x_1 \cdot x_2$
2. $x_3 = x_1 + x_2$
3. $x_3 = x_1 - x_2$
4. $x_3 = e^{-(x_1 - x_2)^2}$
5. $x_3 = random(0.0, 1.0)$

In which cases can the perceptron correctly classify the XOR data? Why, if not? Why is case 5 not well chosen?

Are there further possibilites to make a perceptron classify the XOR data correctly?

Task 5.3 Perceptron and AND-Classification

Train a perceptron on the logical AND function.

Consider the code from Stephen Marsland's book "Machine Learning - An Algorithmic Perspective" at: http://seat.massey.ac.nz/personal/s.r.marsland/MLBook.html

The class `pcn`, which implements the perceptron, can be found in the file pcn_logic_eg.py (in "Chapter 2").

What does the method pcnfwd therein do? Characterise the matrices: inputs, outputs, weights.

The following is the code (from the book) that trains the perceptron from this class with the AND data:

```
from numpy import *
import pcn_logic_eg
inputs = array([[0,0],[0,1],[1,0],[1,1]])          # the input data
targets = array([[0],[0],[0],[1]])                 # the outputs for AND
p = pcn_logic_eg.pcn(inputs,targets)
p.pcntrain(inputs,targets,0.25,6)
```

Run the program (or any perceptron of your choice) on these data. Has the neuron learnt all AND data?

How many parameters did the programmer set; how many parameters did the network learn?

Print the confusion matrix (use the corresponding method in the class pcn)!

Read out the weights and threshold, and compute the line that forms the decision boundary!

Can this problem be solved with negative weights, or negative bias?