1 ACML Homework: Neural Networks

In this lab you will implement a neural network. The implementation can be made in any language of your choice: Java, MatLab, Octave, Python, Prolog (for the extra challenge) ... your choice, but you can not (yet) rely on high-level libraries such as Keras, PyTorch, Theano, Tensorflow, Deeplearning4J, the Matlab Neural Network Toolbox, etc. You need to implement your own backpropagation for this assignment. You should solve this assignment in pairs. (Hint: Things get easier when you vectorise.)

The goal of the implementation is to produce and train a network with 3 layers: input - hidden - output. Both the input layer and the output layer will have 8 nodes, the hidden layer only 3 nodes (+biases).

The learning examples will each have 7 zeros and 1 one in them (so there will be only 8 different learning examples, and you will have to repeat them) and the output the network should learn is exactly the same as the input. So when the input layer is given $< 0, 0, 0, 1, 0, 0, 0, 0 >$ as input, the output to aim for is also $< 0, 0, 0, 1, 0, 0, 0, 0 >$.

Try to get your network to learn this reproducing function on the 8 different learning examples.

Then study the weights and the activations of the hidden nodes of your network and try to interpret them. Your deliverables for this week's assignment are (1) a runnable (compilable and runnable) demo of your network (this means including any extra files such as learning examples if needed) and (2) a short report where you (i) describe your software and how to use it (like a README file), (ii) a brief description of the learning performance of your network (how many examples does it need to converge, how long does that take, does it converge every time or do you have to be lucky with the parameters[1], etc.) and (iii) your interpretation of the learned weights.

Handing in

To hand in, register both students as part of an assignment pair and upload your[2] zipped code, report (in PDF format only) and **a short (2-5 minutes) video** tracing your code and report separately (so 3 different files, namely zip (with the code), pdf and video) through the student portal. Late submissions will NOT be graded (you will not get credit for your work). Both students upload and both mention the other student's name in the comments of the submission.

Note: a Jupyter Notebook that combines code and report can be also submitted. However, you must submit not only the ipynb file but also a pdf version. Therefore, the three files to be submitted will be an ipynb, a pdf and the video.

Hint on the video preparation

The video must be short, we will stop checking it after minute 5:00. To decide what to prioritize in the video, we highly recommend that you base your explanation on the rubric provided within the assignment in Canvas.

The video can be prepared by one or both students.

[1] Since this was apparently not obvious in the past, this implies actually trying different parameter values.

[2] There are probably more NN implementations available on the internet than I can count on my 10 binary fingers. You can of course use these for inspiration, but please do not steal an implementation and turn this in as your own. Not only is this in violation of our rules, but you will learn so much less doing this, and you (or someone else on your behalf) are paying for you not to learn as much as you possibly can.