Practical work 06 – 28/03/2023
Batch-Normalization

Objectives

Main objective is to repeat certain findings of X. Glorot and J. Bengio (c.f. chapter 7 of the lecture notes) to understand the importance of the correct weight initialization for deep neural networks. In addition, batch normalization shall be implemented and its influence on the reduction of the *Internal Covariate Shift* shall be understood.

Submission

- **Deadline** : Tuesday 4th April, 15pm
- **Format** :
  - Completed Jupyter notebook with the blanks filled in (the sections to be completed marked as usual).
  - Comments and results (plot with weight histograms showing the results for the different parameter settings) either in the notebook or in a pdf-report.

  Submission of all files in a single zip-file using the naming convention (for team of two students #1, #2) :

  ```
  family name_given name #1- family name_given name #2.zip
  ```

Starting point for the following work is the Jupyter notebook `backprop-ext-stud.ipynb`, which is an extension to the notebook used to implement the backpropagation algorithm (PW05). You can directly insert your solution of PW05 into the corresponding voids. Below we will extend this notebook with proper weight initialization and with batch normalization.

Exercise 1 Weight Initialization

For all the following we will use the weight initialization schemes for the uniform distribution $U[-r, r]$ (as given in Fig.121 of the lecture notes).

— Extend the Dense Layer with 'completely wrong' weight initialization using a uniform distribution $U[-1, 1]$. Perform a few training steps and convince yourself, that the activations are highly saturated (**Test 1** in notebook). Explain in a few words, why these saturations occur.

— Now extend the Dense Layer with the improved but non-optimal weight initialization as used by X. Glorot and J. Bengio in their first trial $\sqrt{2/(n_{in} + n_{out})}$ (Equation 10 in lecture notes). Perform a few training steps and convince yourself, that the variances of the activations decrease constantly throughout the neural net (**Test 2** in notebook). The curves should look similar to Fig.6 top in X. Glorot and J. Bengio. Explain in a few words, why the variance decreases and by how much from layer to layer.

— Finally extend the Dense Layer with with proper *Xavier* weight initialization scheme i.e. $\sqrt{6/(n_{in} + n_{out})}$. Perform a few training steps and convince yourself, that the variances of the activations remain constant throughout the neural net (**Test 3** in notebook). The curves should look similar to Fig.6 bottom in X. Glorot and J. Bengio.

Exercise 2 Batch Normalization

— Now, implement propagation and backpropagation of the batch normalization layer in the notebook. We implement the batch normalization as an independent layer for reasons of clarity. The relevant formulas are given in chapter 8.1 of the lecture notes. Check your implementation with the corresponding unit tests at the end of the notebook.

— Further check the proper implementation of the batch normalization gradient of the MLP by running the gradient checking (in section 'Test' subsection 'Test analytical value of derivative using backpropagation ...'). This check iterates through all weights, computes the numeric approximation and tests for discrepancies larger than a given limit accuracy ($4.0 \cdot 10^{-7}$). Numeric output is provided only if this threshold is exceeded.

— Now again extend the Dense Layer with with weight initialization as used by X. Glorot and J. Bengio in their first trial $\sqrt{2/(n_{in} + n_{out})}$ (Equation 10 in lecture notes). Perform a few training steps using a layer architecture using a batch normalization layer after each dense layer and observe the variances throughout the neural net (**Test 4** in notebook). Explain in a few words the behaviour observed and the benefits of the batch normalization layer.