Homework 3: Deep Learning

Out May 9; Due May 15, 12 a.m.*

Kristian Kersting, Dominik Hintersdorf, Quentin Delfosse
{kersting, dominik.hintersdorf, quentin.delfosse}@cs.tu-darmstadt.de

1. Implement the following network in PyTorch.

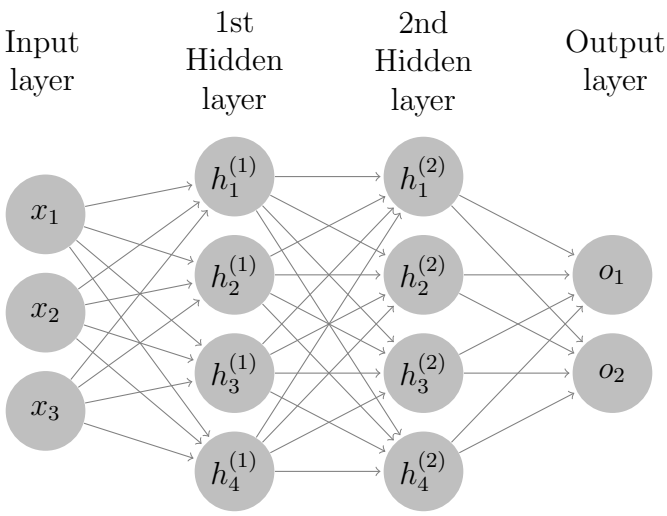

Figure 1: A feed-forward neural network architecture consisting of two hidden layers.

- The sigmoid activation function is used for computing hidden activations.
- No activation function in the output layer.
- In this homework, please use the attached notebook.

And the squared loss function:

$$\mathcal{L}(\theta; \mathbf{x}, \mathbf{y}) = \frac{1}{M} \sum_{n=1}^{M} \frac{1}{2} \| f_\theta(\mathbf{x}_n) - \mathbf{y}_n \|^2 \tag{1}$$

where $f_\theta(x)$ denotes the outputs of the network given inputs $\mathbf{x}$, $\mathbf{y}$ are the targets, $\mathbf{M}$ are the number of samples in the batch and θ denotes a set of the parameters.

*We will discuss the solutions in the exercise session. It is my suggestion that you try to address at least 50% of the exercise questions. Simply try hard to solve them. This way, you will get familiar with the technical terms and with the underlying ideas of the lecture.

2. Compute the forward pass using PyTorch, print the output values of every layer.

3. Compute the gradients using PyTorch, i.e., loss.backward(), param.grad and show:

$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_1}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{b}_1}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{W}_2}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{b}_2}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{W}_3}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{b}_3}$

4. Did you get the same results as in your previous homework?

Some references:
`https://pytorch.org/tutorials/beginner/blitz/autograd_tutorial.html`
`https://github.com/yunjey/pytorch-tutorial`