UNIVERSITÄT ZU LÜBECK
INSTITUTE FOR ROBOTICS
AND COGNITVE SYSTEMS

ADAPTIVE **AI** LAB
https://www.adaptiveailab.com

Prof. Dr. Sebastian Otte
Coşku Can Horuz

Winter Term 2024/2025

Deep Learning

Exercise Sheet 01

Release: October 24, 2024   Deadline: November 21, 2024 (11:59 am)

General remarks:

- Please download the exercise sheet and the corresponding Python code via **GitHub classroom**. This platform should be used for submitting your solutions as well.
- **Always add a documentation to your submission**. The documentation should contain: your pen & paper solutions (if applicable), explanations of your code (if needed), and results and discussions of your experiments (if applicable).
- When questions arise, start a forum discussion on Moodle or Webex. Alternatively, you can contact us via email:
  Coşku Can Horuz (cosku.horuz@uni-luebeck.de)
  Sebastian Otte (sebastian.otte@uni-luebeck.de)

1 Derivatives [24 points]

Compute the derivatives of following equations:

(a) Let $f(x) = \cos(x^3)\ln\left(\frac{x^4}{11}\right)$, $f : \mathbb{R} \to \mathbb{R}$. Compute $f'(x)$. [2 points]

(b) Let $f(x) = \exp(-\frac{1}{2\sigma^2}(x-\mu)^2)$, $f : \mathbb{R} \to \mathbb{R}$, where σ and μ are constant. Compute $f'(x)$. [2 points]

(c) Let $\sigma(x) = \frac{1}{1+e^{-x}}$, $\sigma : \mathbb{R}^1 \to \mathbb{R}^1$. Compute $\sigma'(x)$. [2 points]

(d) Let $f(\mathbf{x}) = \left(\sqrt{x_1^2 - x_3}, e^{x_2} cos(4x_3)\right)$, $f : \mathbb{R}^3 \to \mathbb{R}^2$. Compute the Jacobian matrix (the matrix of first-order partial derivatives) $\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$. [2 points]

(e) Let $a : \mathbb{R} \mapsto \mathbb{R}$ and $f(\mathbf{x}) = (a(x_1), ..., a(x_n))$, $f : \mathbb{R}^n \to \mathbb{R}^n$. Compute the Jacobian matrix $\frac{\partial f}{\partial \mathbf{x}}(\mathbf{x})$. What is the characteristic feature of the resulting matrix? [2 points]

(f) Consider the following functions:

$$
\begin{aligned}
f_1(\mathbf{x}) &= \sin(x_1)\cos(x_2), && \mathbf{x} \in \mathbb{R}^2 \\
f_2(\mathbf{x}, \mathbf{y}) &= \mathbf{x}^T \mathbf{y}, && \mathbf{x}, \mathbf{y} \in \mathbb{R}^2 \\
f_3(\mathbf{x}) &= \sin(\log(\mathbf{x}^T \mathbf{x})), && \mathbf{x} \in \mathbb{R}^n
\end{aligned}
$$

Compute the gradient $\nabla_{\mathbf{x}} f_i(\cdot)$ wrt. $\mathbf{x}$ for all three functions. What are their dimensions? [6 points]

(g) We define

$$
\begin{aligned}
f(\mathbf{z}, \mathbf{y}) &:= \log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}, \mathbf{y}) \\
\mathbf{z} &:= t(\mathbf{w}, \mathbf{y})
\end{aligned}
$$

the functions p, q, t are differentiable and $\mathbf{x} \in \mathbb{R}^D$, $\mathbf{z} \in \mathbb{R}^E$, $\mathbf{y} \in \mathbb{R}^F$, $\mathbf{w} \in \mathbb{R}^F$. Compute the gradient $\nabla_{\mathbf{y}} f(\mathbf{z}, \mathbf{y})$ by using the chain rule. [4 points]

(h) Let $f(\mathbf{y}) = \sin(\mathbf{y})$, and $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$, where $\mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{x} \in \mathbb{R}^N, \mathbf{b} \in \mathbb{R}^M$. Moreover, $\sin(\cdot)$ is an element-wise operation. Compute the derivative $\frac{df}{d\mathbf{x}}$ by exploiting the chain rule and give the dimension. You can write your answer in matrix notation. [4 points]

2 Computational Graph and Backpropagation [36 points]

Complete the following exercises for each equation.

$$
\begin{aligned}
\mathcal{A} &= \sqrt{a + b + c^2} + \log(a + b + c^2) + \frac{a + b + c^2}{bc^2} \\
\mathcal{B} &= \sum_{i=1}^{3} (w_0 + w_1 x_i - y_i)^2
\end{aligned}
$$

(a) Introduce intermediate variables $(t_1, t_2, t_3, ...)$ such that there is a single mathematical operation per assignment. Reuse intermediate variables whenever possible. [8 points]

(b) Draw the computation graph (forward pass) based on the variables introduced in (a). [16 points]

(c) For each node in the graph, write down the partial derivative products and sums to compute the gradient of the corresponding node. Your expressions only need to contain the direct children for the node and you do not need to symbolically calculate or simplify the derivatives. [12 points]

3 Autograd Implementation [40 points]

Please complete the exercises in `DL-Sheet01.ipynb`. Make yourself first familiar with the code until you understand every line.

(a) Complete the backward and forward passes for the given computation graph nodes (look for `# TODOs`). [30 points]

(b) Test your code with the equations $\mathcal{A}$ and $\mathcal{B}$. [10 points]