Foundations of Deep Learning **Fall 2024**

Exercise 1: Neural Networks: Basic Components

Lecturer: Aurelien Lucchi

Problem 1 (Activation functions):

The softplus function s_0 and the sigmoid function s_1 are common activation functions used in neural networks:

$$s_0 : \mathbb{R} \to \mathbb{R},\ x \mapsto \ln(1 + e^x);$$
$$s_1 : \mathbb{R} \to \mathbb{R},\ x \mapsto \frac{1}{1 + e^{-x}}.$$

a) Plot (manually/digitally) the softplus function s_0 and add the asymptotes for $x \to \pm\infty$. Show that $s_0' = s_1$.

b) Plot (manually/digitally) the sigmoid function s_1 and add the asymptotes for $x \to \pm\infty$. Show that

$$s_1'(x) = s_1(x)(1 - s_1(x)).$$

c) The Rectified Linear Unit (ReLU) l_0 and the Heaviside step function l_1 are also common activation functions used in neural networks:

$$l_0 : \mathbb{R} \to \mathbb{R},\ x \mapsto \max\{0, x\};$$
$$l_1 : \mathbb{R} \to \mathbb{R},\ x \mapsto \begin{cases} 1 & ,\ x \geq 0 \\ 0 & ,\ x < 0. \end{cases}$$

Name one similarity and one difference between s_0 and l_0 (resp. between s_1 and l_1).

Problem 2 (Sigmoid vs. Hyperbolic Tangent):

For any activation function $\sigma : \mathbb{R} \to \mathbb{R}$, we extend its definition to multivariate case by entry-wise evaluation:

$$\sigma : \mathbb{R}^n \to \mathbb{R}^n,\ (x_i)_{i=1}^n \mapsto (\sigma(x_i))_{i=1}^n.$$

Consider the following two-layer feedforward network with $\left(\tanh : \mathbb{R} \to \mathbb{R},\ x \mapsto \frac{e^x - e^{-x}}{e^x + e^{-x}}\right)$ as the activation function:

$$F : \mathbb{R}^d \to \mathbb{R}^m,\ \mathbf{x} \mapsto \mathbf{W}_2 \tanh(\mathbf{W}_1 \mathbf{x} + b_1) + b_2,$$

where $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{W}_1 \in \mathbb{R}^{n \times d}$, $b_1 \in \mathbb{R}^n$, $\mathbf{W}_2 \in \mathbb{R}^{m \times n}$, $b_2 \in \mathbb{R}^m$.

a) Show that there exists a two-layer network

$$G(\mathbf{x}) = \mathbf{W}_2' s_1(\mathbf{W}_1' \mathbf{x} + b_1') + b_2',$$

which computes exactly the same function as $F(x)$ using weight matrices and bias vectors of the same dimension, but with the sigmoid activation function s_1. Write down the weights $\mathbf{W}_1', \mathbf{W}_2', b_1', b_2'$ in terms of $\mathbf{W}_1, \mathbf{W}_2, b_1, b_2$.
(Hint: You can first show that $\tanh(x) = 2s_1(2x) - 1$, where s_1 is the Sigmoid function.)

b) Does the same statement hold if s_1 is replaced by l_1? Why?

Problem 3 (Gradient for Three Layer network with softmax activation and cross-entropy loss):

Consider a three layer neural network with a softmax activation at the output defined as

$$f_{\mathbf{W},\boldsymbol{\Theta}}(\mathbf{x}) = \sigma^{\max}(\mathbf{W}\mathbf{x}; \Theta),$$

where the softmax function converts the logits $\mathbf{W}x$ into a probability distribution via

$$\sigma_i^{\max}(\mathbf{x}; \boldsymbol{\Theta}) = \frac{\exp(\mathbf{x}^\top \boldsymbol{\theta}_i)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \boldsymbol{\theta}_j)}, \quad \boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k].$$

The cross-entropy loss is chosen to train the network, i.e.

$$\ell(\mathbf{x}, \mathbf{y}; \mathbf{W}, \boldsymbol{\Theta}) = -\mathbf{y}^\top \ln(\sigma^{\max}(\mathbf{W}\mathbf{x}; \boldsymbol{\Theta}))$$

a) Derive the derivative with respect to each $\boldsymbol{\theta}_i$, i.e. $\frac{\partial \ell(\mathbf{x},\mathbf{y};\mathbf{W},\boldsymbol{\Theta})}{\partial \boldsymbol{\theta}_i}$.

b) Derive the derivative with respect to $\mathbf{W}$, i.e. $\frac{\partial \ell(\mathbf{x},\mathbf{y};\mathbf{W},\boldsymbol{\Theta})}{\partial \mathbf{W}}$.
Hint: Make use of the matrix calculus to simplify the computation. Alternatively, calculate the the derivative entry-wise and combine it into a matrix expression.