Exercise set 1: Deep learning 1

Bonus submission deadline: 30.03.2021 23:59

The exercises marked with **BONUS** should be returned in Moodle by the submission deadline. Successfully solving it will give you bonus points to boost your final grade. During the exercise session, the solution of the exercises will be discussed.

Exercise 1: backpropagation (BONUS)

Figure 1 shows the fundamental unit neural networks, the perceptron. The output y of the perceptron in the figure is given by $y = \sigma(w_1 x_1 + w_2 x_2 + b)$, where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function, x_1, x_2 are the input values and w_1, w_2, b are the parameters to be learned.

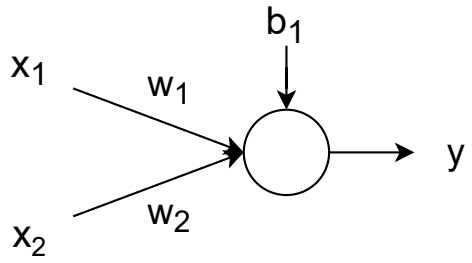
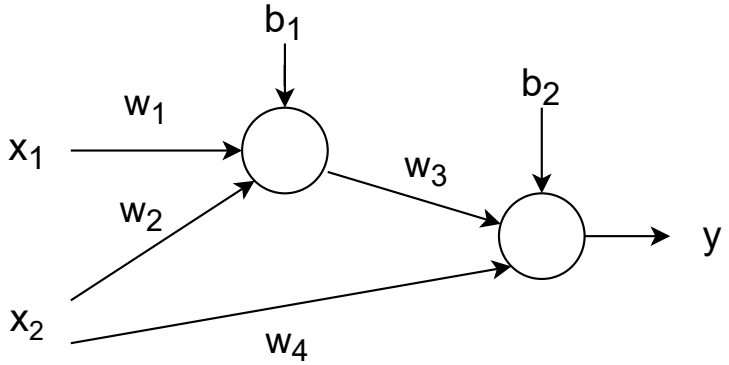


Figure 1: Perceptron

Consider the following small network, which has input values $x_1 = 1, x_2 = 2$ and weight values $w_1 = 1, w_2 = -1, b_1 = 1, w_3 = 4, w_4 = -1, b_2 = 0$



- Compute the value of the output y of the network with the given values. Assume the target output is $t = 1$, compute the value of the loss

function $E = \frac{1}{2}(y-t)^2$.

- For each parameter p compute the value of the partial derivative $\frac{\partial E}{\partial p}$ with the given values.
  Hint! Computing the expression for each partial derivative and evaluating it might be a lot of work. Instead, you may want to introduce the intermediate variables $z_1 = w_1x_1 + w_2x_2 + b_1$, $y_1 = \sigma(z_1)$ and $z_2 = w_3y_1 + w_4x_2 + b_2$ and compute the derivatives with the chain rule. Remember to recycle your calculations! You may also find useful to know that $\frac{\partial \sigma(x)}{\partial x} = \sigma(x)(1-\sigma(x))$.

- Update the value of each parameter using the update rule $p = p - \frac{\partial E}{\partial p}$. Compute the value of the network output and loss function using the updated parameters values, what do you observe?

Exercise 2: Linear Discriminant Analysis

Given a sample represented by a feature vector $\mathbf{x} \in \mathbb{R}^n$, a linear classifier has the following structure

$$y = \begin{cases} 1, & \text{if } \mathbf{w} \cdot \mathbf{x} \geq c \\ 0, & \text{if } \mathbf{w} \cdot \mathbf{x} < c \end{cases} \tag{1}$$

training the linear classifier now means to compute the weigth vector $\mathbf{w}$. Suppose you have split your training set into the matrices $X_0 \in \mathbb{R}^{m_0 \times n}$ containing the sample for the class 0 and $X_1 \in \mathbb{R}^{m_1 \times n}$. The *Fisher classifier* (or LDA classifier) is defined as follows

$$\mathbf{w} = (\Sigma_0 + \Sigma_1)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \tag{2}$$

where μ, Σ denote the mean vector and covariance matrix of the two classes. It is also common practice to normalize the result afterwards. A common choice for the threshold is $c = \frac{1}{2}(\mathbf{w} \cdot \boldsymbol{\mu}_0 + \mathbf{w} \cdot \boldsymbol{\mu}_1)$

- In the script `ex2.m2`, we load data belonging to two classes. Each row is a feature vector. Implement the Fisher classifier

- What is the geometrical interpretation of computing $\mathbf{w} \cdot \mathbf{x}$?

- Plot the histograms of the values $\mathbf{w} \cdot \mathbf{x}$ for the two classes. Try changing the values in $\mathbf{w}$. What do you observe? Can you use the histograms to motivate the choice of the threshold?

Exercise 3: Precision and recall

- You have trained a classifier to find the images that contain a cat. You give your classifier 100 images, out of which 80 contain a cat. You classifier returns 50 images, out of which only 40 actually contain a cat. Write down the confusion matrix of the results.
- compute the precision and recall of the results
- The following table reports the probabilities that a picture contains a cat computed by the classifier, as well as the ground truth. Plot the precision-recall curve and compute the Area Under Curve (AUC)

score	ground truth
0.9	1
0.8	1
0.7	0
0.6	1
0.5	0
0.4	1
0.3	0
0.2	1
0.1	0