Mathematics of Learning – Worksheet 11 - – Discussion on Jan 11th/12nd, 2024

- The exercise sheets will be uploaded every Monday. Solution sketches will be uploaded one week later.
- You can hand in your own solutions via StudOn and we correct them - this is not mandatory. Please hand in small groups of 2-3 students.
- For questions, please use the forum on StudOn since other students may have similar questions. If you have a more personal question about the exercises please send an email to ehsan.waiezi@fau.de or lars.weidner@fau.de respectively.

Exercise 1 [More about neural networks - theoretical and difficult].
Consider F_I^L the set of Lipschitz continuous functions mapping from a compact real interval I to the real numbers. Lipschitz continuous means,

$$\exists L > 0 : |f(x) - f(y)| \leq L|x - y| \text{ for all } x, y \in I.$$

Consider the set F_I of functions which can be linearly combined of right shifted Heavyside step functions, i.e.

$$F_I = \{f : I \to \mathbb{R} : \ \exists n \in \mathbb{N}, \lambda, b \in \mathbb{R}^n : f(x) = \sum_{i=1}^{n} \lambda_i H(x + b_i) \text{ for all } x \in I\}.$$

H denotes the Heavyside step function, i.e., $H(x) = 1$ if x is nonnegative and 0 otherwise.
Prove or disprove: The closure, corresponding to the $||\cdot||_\infty$ norm on functions, of F_I, is a superset of F_I^L.

Solution. We have to show that we find for an arbitrary Lipschitz function $f \in F_I^L$ a sequence $f_n \in F_I$ which converges uniformly to f. Hence, let f be an arbitrary Lipschitz function on I with constant L.
We know the following: since f is continuous, and $I := [a, b]$ for some $a, b \in \mathbb{R}$ is compact, $\min_{x \in I} f(x)$ and $\max_{x \in I} f(x)$ exist. Furthermore, the difference of maximum and minimum is bounded by the Lipschitz inequality, since for arbitrary $x, y \in I$ it holds that $|f(x) - f(y)| \leq L|x - y| \leq L(b - a)$.
This also applies in case we split the interval in n smaller sub intervals, i.e. $I = \bigcup_{i=1}^{n} I_i^{(n)}$ with $I_i^{(n)} := \left[a + \frac{(b-a)(i-1)}{n}, a + \frac{(b-a)i}{n}\right]$.
For these it holds that $\max_{x \in I_i^{(n)}} f(x) - \min_{x \in I_i^{(n)}} f(x) \leq L\frac{1}{n}(b - a)$. We exploit that property to define the sequence of functions

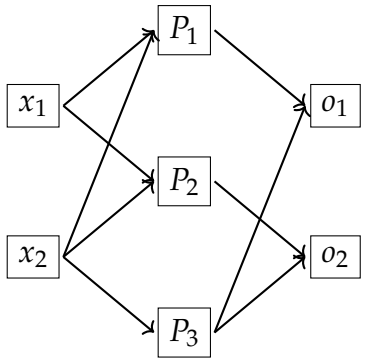
$$f_n(x) = \min_{y \in I_i^{(n)}} f(y), \ \text{ if } x \in \left[a + \frac{(b-a)(i-1)}{n}, a + \frac{(b-a)i}{n}\right),$$

which can be expressed as finite sums of Heavyside functions. The limit

$$\lim_{n\to\infty} ||f_n - f||_\infty = \lim_{n\to\infty} \sup_{x\in I} |f_n(x) - f(x)| \leq \lim_{n\to\infty} L\frac{(b-a)}{n} = 0,$$

which is the desired result.

Exercise 2 [Neural networks - Calculations].
Given the following network:



The initial weights are all 1, the biases of the output layers are 0, the biases of P_i are 0. Activation functions for all layers are $\psi(t) := \frac{1}{1+e^{-t}}$. The input points are
$X = \left\{ \begin{pmatrix}2\\5\end{pmatrix}, \begin{pmatrix}3\\2\end{pmatrix}, \begin{pmatrix}7\\-2\end{pmatrix}, \begin{pmatrix}8\\4\end{pmatrix}, \begin{pmatrix}-4\\-2\end{pmatrix}, \begin{pmatrix}4\\3\end{pmatrix} \right\}, Y = \left\{ \begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}, \begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix} \right\}.$

1. Look at the data and formulate a guess what the targets Y express.
   Solution. The output seems to decide which of the entries of x_i is larger. If the first entry is larger, then it is mapped to $\begin{pmatrix}1\\0\end{pmatrix}$, if the second is larger, it is mapped to $\begin{pmatrix}0\\1\end{pmatrix}$.

2. Do feedforward passes for all given data points.

   Solution. First, we calculate a feedforward pass for $x^1 = \begin{pmatrix}2\\5\end{pmatrix}$. We start with the first layer and compute for each neuron the value a_{P_i} and apply the activation function to this value (note that the following numbers are all rounded, but we still use " = "):

$$\begin{aligned}
a_{P_1} &= 1 \cdot x_1^1 + 1 \cdot x_2^1 + 0 = 2 + 5 = 7 \\
z_{P_1} = \psi(a_{P_1}) &= \frac{1}{1+e^{-7}} = 0.99909 \\
a_{P_2} &= 1 \cdot x_1^1 + 1 \cdot x_2^1 + 0 = 2 + 5 = 7 \\
z_{P_2} = \psi(a_{P_2}) &= \frac{1}{1+e^{-7}} = 0.99909 \\
a_{P_3} &= 1 \cdot x_2^1 + 0 = 5 \\
z_{P_3} = \psi(a_{P_3}) &= \frac{1}{1+e^{-5}} = 0.99331
\end{aligned}$$

Using these outputs we can now compute the outputs for the neurons in the output layer:

$$
\begin{aligned}
a_{o_1} &= 1 \cdot \psi(a_{P_1}) + 1 \cdot \psi(a_{P_3}) + 0 = 0.99909 + 0.99331 = 1.99240 \\
z_{o_1} = \psi(a_{o_1}) &= \frac{1}{1 + e^{-1.9924}} = 0.88000 \\
a_{o_2} &= 1 \cdot \psi(a_{P_2}) + 1 \cdot \psi(a_{P_3}) + 0 = 0.99909 + 0.99331 = 1.99240 \\
z_{o_2} = \psi(a_{o_2}) &= \frac{1}{1 + e^{-1.9924}} = 0.88000
\end{aligned}
$$

Analogously, we can compute the feedforward passes for all other points. The results are given as follows (with $z = \psi(a)$ in every node):

- For $x^2 = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$:

$$
\begin{aligned}
a_{P_1} &= 5, & \psi(a_{P_1}) &= 0.99331 \\
a_{P_2} &= 5, & \psi(a_{P_2}) &= 0.99331 \\
a_{P_3} &= 2, & \psi(a_{P_3}) &= 0.88080 \\
a_{o_1} &= 1.87411, & \psi(a_{o_1}) &= 0.86693 \\
a_{o_2} &= 1.87411, & \psi(a_{o_2}) &= 0.86693
\end{aligned}
$$

- For $x^3 = \begin{pmatrix} 7 \\ -2 \end{pmatrix}$:

$$
\begin{aligned}
a_{P_1} &= 5, & \psi(a_{P_1}) &= 0.99331 \\
a_{P_2} &= 5, & \psi(a_{P_2}) &= 0.99331 \\
a_{P_3} &= -2, & \psi(a_{P_3}) &= 0.11920 \\
a_{o_1} &= 1.11251, & \psi(a_{o_1}) &= 0.75260 \\
a_{o_2} &= 1.11251, & \psi(a_{o_2}) &= 0.75260
\end{aligned}
$$

- For $x^4 = \begin{pmatrix} 8 \\ 4 \end{pmatrix}$:

$$
\begin{aligned}
a_{P_1} &= 12, & \psi(a_{P_1}) &= 0.99999 \\
a_{P_2} &= 12, & \psi(a_{P_2}) &= 0.99999 \\
a_{P_3} &= 4, & \psi(a_{P_3}) &= 0.98201 \\
a_{o_1} &= 1.98200, & \psi(a_{o_1}) &= 0.87889 \\
a_{o_2} &= 1.98200, & \psi(a_{o_2}) &= 0.87889
\end{aligned}
$$

- For $x^5 = \begin{pmatrix} -4 \\ -2 \end{pmatrix}$:

$$
\begin{aligned}
a_{P_1} &= -6, & \psi(a_{P_1}) &= 0.00247 \\
a_{P_2} &= -6, & \psi(a_{P_2}) &= 0.00247 \\
a_{P_3} &= -2, & \psi(a_{P_3}) &= 0.11920 \\
a_{o_1} &= 0.12167, & \psi(a_{o_1}) &= 0.53038 \\
a_{o_2} &= 0.12167, & \psi(a_{o_2}) &= 0.53038
\end{aligned}
$$

- For $x^6 = \begin{pmatrix} 4 \\ 3 \end{pmatrix}$:

$$\begin{aligned} a_{P_1} &= 7, & \psi(a_{P_1}) &= 0.99909 \\ a_{P_2} &= 7, & \psi(a_{P_2}) &= 0.99909 \\ a_{P_3} &= 3, & \psi(a_{P_3}) &= 0.95257 \\ a_{o_1} &= 1.95166, & \psi(a_{o_1}) &= 0.87563 \\ a_{o_2} &= 1.95166, & \psi(a_{o_2}) &= 0.87563 \end{aligned}$$

This completes the calculation of the feedforward passes.

3. Do 2 or 3 rounds of backpropagation passes (training) with the neural network, using either "pure" stochastic gradient descent (with batch size 1) or stochastic gradient descent with a larger batch size. (If you do not know about stochastic gradient descent methods, wait until the lecture on July 05)
Solution. Depending on the randomly chosen batch B we have to calculate the backpropagation passes of all pairs (x^i, y^i) such that the index i is contained in B. We show one round of the backpropagation pass for the first input output pair (x^1, y^1). The passes for all other pairs can be calculated analogously. First, we compute the derivative of the activation function ψ:

$$\psi'(t) = \frac{e^{-t}}{(1+e^{-t})^2} = \frac{1}{1+e^{-t}}\left(1 - \frac{1}{1+e^{-t}}\right) = \psi(t)(1-\psi(t))$$

We start with the output layer and compute the partial derivatives with respect to the biases and weights of the quadratic loss function $C(\theta) = \frac{1}{2}||f_\theta(x^1) - y^1||^2$, where f_θ denotes the output of the neural network and θ is a vector containing all biases and weights. We start with the biases of the neurons in the last layer:

$$\begin{aligned} \frac{\partial C}{\partial b_{o_1}} &= (z_{o_1} - y_1^1) \cdot \psi'_{o_1}(a_{o_1}) = (z_{o_1} - y_1^1) \cdot \psi(a_{o_1})(1 - \psi(a_{o_1})) \\ &= (0.88 - 0) \cdot 0.88 \cdot (1 - 0.88) = 0.0929, \end{aligned}$$

where z_n denotes the output of the corresponding neuron n. The derivative with respect to b_{o_2} can be calculated analogously:

$$\begin{aligned} \frac{\partial C}{\partial b_{o_2}} &= (z_{o_2} - y_2^1) \cdot \psi'_{o_2}(a_{o_2}) = (z_{o_2} - y_2^1) \cdot \psi(a_{o_2})(1 - \psi(a_{o_2})) \\ &= (0.88 - 1) \cdot 0.88 \cdot (1 - 0.88) = -0.0127. \end{aligned}$$

Now, we can go on with the partial derivatives with respect to the weights for the last layer:

$$\begin{aligned} \frac{\partial C}{\partial w_{o_1,P_1}} &= \frac{\partial C}{\partial b_{o_1}} \cdot z_{P_1} = 0.0929 \cdot 0.99909 = 0.0928 \\ \frac{\partial C}{\partial w_{o_1,P_3}} &= \frac{\partial C}{\partial b_{o_1}} \cdot z_{P_3} = 0.0929 \cdot 0.99331 = 0.0923 \\ \frac{\partial C}{\partial w_{o_2,P_2}} &= \frac{\partial C}{\partial b_{o_2}} \cdot z_{P_2} = -0.0127 \cdot 0.99909 = -0.0127 \\ \frac{\partial C}{\partial w_{o_2,P_3}} &= \frac{\partial C}{\partial b_{o_2}} \cdot z_{P_3} = -0.0127 \cdot 0.99331 = -0.0126 \end{aligned}$$

We move to the next layer and calculate again the derivatives with respect to the biases using the recursion formula given in the lecture:

$$\begin{aligned}
\frac{\partial C}{\partial b_{P_1}} &= \left(\frac{\partial C}{\partial b_{o_1}} \cdot w_{o_1,P_1}\right) \cdot \psi'_{P_1}(a_{P_1}) = (0.0929 \cdot 1) \cdot \psi(a_{P_1}) \cdot (1 - \psi(a_{P_1})) \\
&= 0.0929 \cdot 0.99909 \cdot (1 - 0.99909) = 8.446 \cdot 10^{-5} \\
\frac{\partial C}{\partial b_{P_2}} &= \left(\frac{\partial C}{\partial b_{o_2}} \cdot w_{o_2,P_2}\right) \cdot \psi'_{P_2}(a_{P_2}) = (-0.0127 \cdot 1) \cdot \psi(a_{P_2}) \cdot (1 - \psi(a_{P_2})) \\
&= -0.0127 \cdot 0.99909 \cdot (1 - 0.99909) = -1.155 \cdot 10^{-5} \\
\frac{\partial C}{\partial b_{P_3}} &= \left(\frac{\partial C}{\partial b_{o_1}} \cdot w_{o_1,P_3} + \frac{\partial C}{\partial b_{o_2}} \cdot w_{o_2,P_3}\right) \cdot \psi'_{P_3}(a_{P_3}) \\
&= (0.0929 \cdot 1 - 0.0127 \cdot 1) \cdot \psi(a_{P_3}) \cdot (1 - \psi(a_{P_3})) \\
&= 0.0802 \cdot 0.99331 \cdot (1 - 0.99331) = 5.329 \cdot 10^{-4}
\end{aligned}$$

We finish the first round of backpropagation passes by calculating the partial derivatives with respect to the remaining weights:

$$\begin{aligned}
\frac{\partial C}{\partial w_{P_1,x_1^1}} &= \frac{\partial C}{\partial b_{P_1}} \cdot x_1^1 = 8.446 \cdot 10^{-5} \cdot 2 = 1.6892 \cdot 10^{-4} \\
\frac{\partial C}{\partial w_{P_1,x_2^1}} &= \frac{\partial C}{\partial b_{P_1}} \cdot x_2^1 = 8.446 \cdot 10^{-5} \cdot 5 = 4.223 \cdot 10^{-4} \\
\frac{\partial C}{\partial w_{P_2,x_1^1}} &= \frac{\partial C}{\partial b_{P_2}} \cdot x_1^1 = -1.155 \cdot 10^{-5} \cdot 2 = -2.31 \cdot 10^{-5} \\
\frac{\partial C}{\partial w_{P_2,x_2^1}} &= \frac{\partial C}{\partial b_{P_2}} \cdot x_2^1 = -1.155 \cdot 10^{-5} \cdot 5 = -5.775 \cdot 10^{-5} \\
\frac{\partial C}{\partial w_{P_3,x_2^1}} &= \frac{\partial C}{\partial b_{P_3}} \cdot x_2^1 = 5.329 \cdot 10^{-4} \cdot 5 = 2.6645 \cdot 10^{-3}
\end{aligned}$$

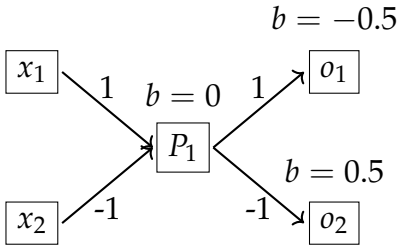
We collect all calculated partial derivatives in one single gradient vector and repeat this procedure for all pairs (x^i, y^i), where i is contained in the chosen batch. To obtain one single gradient we build the mean $\nabla\overline{C}$ of all these gradient vectors and use $\nabla\overline{C}$ to perform a gradient step with a given learning rate $\eta > 0$ and update the parameters with the following formula:

$$\theta^{new} = \theta - \eta\nabla\overline{C},$$

where θ consists of the initially given biases and weights. This completes the first round of training the network.
The second round can be performed analogously by just replacing the initial given biases and weights by the updated values.

4. Generate a few test data points on your own and check the classification accuracy.
Solution. Depending on if your calculations in 3. have been correct, the neural network should classify more or less accurate.

5. Propose an alternative, smaller network architecture, which does the same.
   Solution. Since we guessed that the output is just "which is the larger component", we just need a single neuron in the middle layer and two neurons in the output layer (all with heavyside activation function) to decide this.



Exercise 3 [Proof Exercise: Discriminatory Activation Functions].
Let $\Omega \subset \mathbb{R}^d$ be a compact set. A continuous function $\psi : \mathbb{R} \to \mathbb{R}$ is called discriminatory if for any signed measure $\mu \in \mathfrak{M}(\Omega)$ it holds

$$\int_\Omega \psi(w \cdot x + b)\, \mathrm{d}\mu(x) = 0, \quad \forall w \in \mathbb{R}^d,\, b \in \mathbb{R} \implies \mu = 0.$$

- Prove that the polynomials $\psi(t) = 1$ and $\psi(t) = t$ are *not discriminatory* by finding non-zero signed measures $\mu \in \mathfrak{M}([-1, 1])$ with

$$\int_{-1}^{1} \psi(wx + b)\, \mathrm{d}\mu(x) = 0, \quad \forall w, b \in \mathbb{R}.$$

- Using the general convergence theorem from the lecture, prove that polynomials are *not discriminatory*.

  Hint: Characterize the approximation space

$$\Sigma_d(\psi) = \left\{ \sum_{i=1}^{N} \alpha_i \psi(w_i \cdot x + b_i) \,:\, N \in \mathbb{N},\, w_i \in \mathbb{R}^d,\, b_i \in \mathbb{R} \right\}$$

  in the case that ψ is a polynomial of degree $p \in \mathbb{N}$.

  It suffices to argue in the one-dimensional case $d = 1$.

- Prove that $\mathrm{ReLU}(t) := \max(t, 0)$ is a discriminatory activation.

  Hint: You can cleverly combine two ReLU functions into a sigmoidal function, as defined in the lecture.

Solution. If we want to show that a function is not discriminatory, we are looking in general for measures, for which regardless of the values of w and b, i.e., we are looking for a signed measure $\mu \in \mathfrak{M}([-1, 1])$ (only in our case defined on the set $[-1, 1]$, in general this is different), such that

$$\int_{-1}^{1} \psi(wx + b)\, \mathrm{d}\mu(x) = 0.$$

Hence,

- If $\psi(t) = 1$ we can define (for example, there are so many possible measures here...) the non-zero signed measure $d\mu(x) = x\,\mathrm{d}x$ which yields

$$\int_{-1}^{1} \psi(wx+b)\,\mathrm{d}\mu(x) = \int_{-1}^{1} x\,\mathrm{d}x = 0, \quad \forall w, b \in \mathbb{R}.$$

For $\psi(t) = t$ we can use here a discrete measure which assigns mass 1 to $\{-1\}$ and $\{1\}$ and mass -2 to $\{0\}$, i.e., $\mu = \delta_{-1} + \delta_1 - 2 \cdot \delta_0$ while $\delta_\cdot$ denotes the dirac-measure.

This simplifies the integration from -1 to 1 to an easy sum, i.e.,

$$\int_{-1}^{1} \psi(wx+b)d\mu(x) = \psi(-w+b) - 2 \cdot \psi(b) + \psi(w+b).$$

In the case where ψ is the identity function on real numbers, this sum evaluates to $-w + b - 2 \cdot b + w + b = 0$, no matter of w and b.

- We argue in one dimension $d = 1$ (for $d > 1$ it is more technical, but analogous). We remember the universal approximation theorem from the lecture:

Theorem 1 *Let $\psi\colon \mathbb{R} \to \mathbb{R}$ be a continuous discriminatory funciton. Then finite sums of form*

$$G(x) = \sum_{j=1}^{N} a_j \psi(w_j^T x + b_j)$$

are dense in $C(I_n)$. This means: For an arbitrary continuous function $f \in C(I_n)$ and an arbitrary $\varepsilon > 0$ there exists an N and a function G as above, such that:

$$|G(x) - f(x)| < \varepsilon \quad \forall x \in I_n.$$

In the case that ψ is a polynomial of degree $p \in \mathbb{N}$ it holds

$$\psi(w_i x + b_i) = \sum_{j=0}^{p} \beta_j (w_i x + b_i)^j = \sum_{j=0}^{p} \beta_{ij} x^j,$$

for suitable coefficients β_{ij}.

Hence, the approximation space $\Sigma_d(\psi)$ (this is the set of functions which can be linearly combined of functions $\psi(w_j x + b_j)$, $j = 1, ..., N$ as the $G(x)$ in the theorem above) coincides with

$$\begin{aligned}\Sigma_d(\psi) &= \left\{ \sum_{i=1}^{N} \alpha_i \sum_{j=0}^{p} \beta_{ij} x^j \;:\; \alpha_i,\, \beta_{ij} \in \mathbb{R} \right\} \\ &= \left\{ \sum_{j=0}^{p} \gamma_j x^j \;:\; \gamma_j \in \mathbb{R} \right\},\end{aligned}$$

which is simply the space of all polynomials with degree at most p.

This space is a closed subspace of the continuous functions,[1] which means that it is not dense. Hence, inverting the statement of the theorem, polynomials are

[1] since sums and limits of polynomials with degree at most p are polynomials with degree at most p

not discriminatory (if they would be discriminatory, their approximation space would be dense in the continuous functions, what is not the case).

Remark: Closed subsets have the property, that any converging sequence in the closed subset has the limit in the closed subset (take for example an open interval: there is a sequence, which converges to the border, but the border is not in the interval; hence this is not closed).
The definition of "dense" is, that a subset lies dense in a superset, if for every arbitrary element in the superset there exists a converging sequence in the subset which converges to this element (the standard example is $\mathbb{Q}$ is dense in $\mathbb{R}$.)
Taking both together: If a subset is closed and not equal to the superset, then it is not dense.

- To show that ReLU is discriminatory we argue as follows: Assume that

$$\int_\Omega \mathrm{ReLU}(w \cdot x + b)\, \mathrm{d}\mu(x) = 0, \quad \forall w \in \mathbb{R}^d,\, b \in \mathbb{R}.$$

We can define the sigmoidal function

$$\psi(x) = \mathrm{ReLU}(x) - \mathrm{ReLU}(x-1) = \begin{cases} 0, & x < 0, \\ x, & 0 \le x < 1, \\ 1, & x > 1. \end{cases}$$

and compute

$$\begin{aligned} \int \psi(w \cdot x + b)\, \mathrm{d}\mu(x) &= \int \mathrm{ReLU}(w \cdot x + b)\, \mathrm{d}\mu(x) - \int \mathrm{ReLU}(w \cdot x + b - 1)\, \mathrm{d}\mu(x) \\ &= 0 - 0 = 0, \quad \forall w \in \mathbb{R}^d,\, b \in \mathbb{R}. \end{aligned}$$

Since ψ is sigmoidal it is discriminatory according to the lecture. This implies $\mu = 0$ and hence ReLU is also discriminatory.