Universität des Saarlandes
Prof. Dr. Dietrich Klakow
Lehrstuhl für Signalverarbeitung
NNTI Winter Term 2023/2024

Exercise Sheet 5

Neural Networks Basics

Deadline: 07.12.2023 23:59

Guidelines: You are expected to work in a group of 2-3 students. While submitting the assignments, please make sure to include the following information for all our teammates in your PDF/python script:

Name:

Student ID (matriculation number):

Email:

Your submissions should be zipped as **Name1_id1_Name2_id2_Name3_id3.zip** when you have multiple files. For assignments where you are submitting a single file, use the **same naming convention** without creating a zip. For any clarification, please reach out to us on the **CMS Forum**.
Note that the above instructions are mandatory. If you are not following them, tutors can decide not to correct your exercise.

Exercise 5.1 - Gradient Descent (1+0.5 points)

a) Let $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ be the following real function, where $a = 1, b = 100$:

$$f(\mathbf{x}) = (a - x_1)^2 + b(x_2 - x_1^2)^2$$

Perform gradient descent on f in order to minimize it. Use the following value as x_0

$$\mathbf{x_0} = \begin{bmatrix} 0.9 \\ 1.12 \end{bmatrix} \tag{1}$$

Use a learning rate $\epsilon = 0.0001$. Do 3 iterations or until the gradient becomes zero, whichever happens first. On each iteration make sure to show the following:

- The value of $\mathbf{x}$
- The value of $\nabla f(\mathbf{x})$
- The value of $f(\mathbf{x})$

Once finished, compare the values $f(\mathbf{x})$ and $f(\mathbf{x}_0)$.

What would happen if a learning rate of ϵ_1 or ϵ_2 were used? Where

$$\epsilon_1 < \epsilon \lll \epsilon_2 \,.$$

b) Finally, find an $\hat{\mathbf{x}}$ that minimizes f and discuss the speed of convergence towards that point. You are free to choose your own learning rate; justify your decision. Is this $\hat{\mathbf{x}}$ guaranteed to be a global minimum given the nature of gradient descent and the function? (Hint: discuss the convexity) (*max 3-4 sentences*)

Exercise 5.2 - Weight Space Symmetry (0.5 + 0.5 points)

Consider a Neural Network with a single hidden layer consisting of M neurons and *tanh* activation function.For any neuron in the hidden layer, simultaneous change of sign of input and output weights from the neuron leads to no change in the output layer therefore producing an equivalent transformation.Similarly, for any pair of neurons interchange of input weights between the neurons and simultaneous interchange of output weights produces an equivalent transformation:

(a) Find the total number of equivalent transformation for the hidden layer.

(b) Consider a deep neural network with N hidden layers. Each hidden layer consists of M_i neurons where $i \in 1, 2, ..., N$ and *tanh* activation function. Find the total number of equivalent transformations for the network

Exercise 5.3 - SVM and Kernels (1.5+0.5+0.5 points)

Let us consider a classification problem with target class $y_i \in \{+1, -1\}$ which we want to solve using SVMs (see the book of Pattern Recognition and Machine Learning in the Sparse Kernel Machines chapter to know more). For any input $\vec{x}$, consider the vector $\vec{\mathbf{w}}$ (perpendicular to the classifier) such that $\vec{\mathbf{w}}.\vec{x} + b \geq 0$ determines the class of the input.

The fundamental idea of SVM is to find the hyperplane that maximizes the margin between the two classes. This is formulated as an optimization problem where the objective is to maximize the distance between the nearest points of each class to the hyperplane.

SVMs are known as maximal-margin classifiers, this margin is calculated as $\frac{2}{\|\mathbf{w}\|}$, where $\mathbf{w}$ is the weight vector normal to the hyperplane. The optimization problem in SVMs aims to minimize $\|\mathbf{w}\|^2$ which is equivalent to maximizing this margin.

(a) Solve the following constraint-based optimization problem using the *Lagrangian* to find the *extremum* (see here for more information):

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\mathbf{w}^T \cdot \mathbf{x}_i + b) - 1 \right]$$

where α_i is the Lagrange multiplier.
For this you have to first find $\mathbf{w}$ and b by setting the derivatives of $L(\mathbf{w}, b, \alpha)$ w.r.t $\mathbf{w}$ and b to zero and then replace them in $L(\mathbf{w}, b, \alpha)$.

(b) What is the significance of the term α_i in SVMs? (*max 2 sentences*)

(c) (optional) (0 Points) Until now we were dealing with linear data. Is it possible to change the maximized margin $\|\mathbf{w}\|^2$ so that it works for non-linear data as well? Justify your answer. There is no need for mathematical proofs or derivations, but you can support your answer with relevant equations. (*max 4 sentences*)

In the lecture, in *Chapter 4* while discussing the Support Vector Machines, we came across the idea of *kernelization* or the so-called kernel trick. This trick consists of mapping a feature vector in low dimensional space to a higher dimensional space. This allows to model highly non-linear decision boundaries for non-linear data and thus achieve a more flexible classifier while retaining computational simplicity.

Formally *kernelization* involves finding a mapping $\phi : \mathcal{X} \rightarrow \mathcal{Z}$ such that, $\mathcal{Z}$ has a higher dimension than $\mathcal{X}$ and the computation in $\mathcal{Z}$ only uses inner product.
There is a function K called kernel such that the inner product of $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ is $K(\mathbf{x}_i, \mathbf{x}_j)$.

(d) Using the resulting Lagrangian in b) kernalize the SVM formulation and find the value of $\vec{\mathbf{w}}$.

Exercise 5.4 - SVM (5 points)

See the accompanying Jupyter notebook.