Exercises
Introduction to Machine Learning
Spring 2023

Institute for Machine learning
Dept. of Computer Science, ETH Zürich
Prof. Dr. Andreas Krause, Prof. Dr. Fanny Yang

For questions, please refer to Moodle.
Released on **28 March 2023**

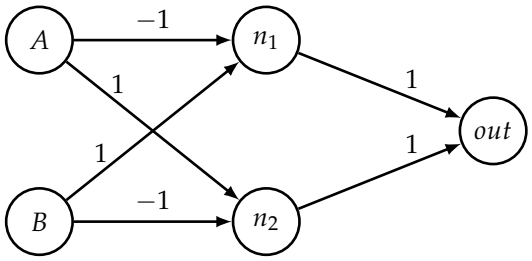
Homework 3
(Neural Networks)

General Instructions

- Submission of solutions is not mandatory but solving the exercises are highly recommended. The master solution will be released next week.
- Part of the exercises are available on Moodle as a quiz. These problems are marked with [☑].

Exercise 1: Neural Network Learns Boolean Functions

Consider the following neural network with one hidden layer and no bias. The neural network uses the ReLU activation function for the hidden layer, and mean squared error loss. The weights are initialized as follows:



We denote $\boldsymbol{x} = (A, B)^T$, $\boldsymbol{z} = W^{(1)}\boldsymbol{x}$, $\boldsymbol{v} = (n_1, n_2)^T = \sigma(\boldsymbol{z})$, $out = W^{(2)}\boldsymbol{v}$, where σ is the ReLU activation function.

(a) [☑]What are weight matrices of layers 1 ($W^{(1)}$) and 2 ($W^{(2)}$)?

1. $$W^{(1)} = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix},\ W^{(2)} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$
2. $$W^{(1)} = \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix},\ W^{(2)} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$
3. $$W^{(1)} = \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix},\ W^{(2)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$
4. $$W^{(1)} = \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix},\ W^{(2)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

(b) [☑]Calculate the forward propagation for the inputs:

- $\boldsymbol{x_1} = (A, B)^T = (0, 0)^T$
- $\boldsymbol{x_2} = (A, B)^T = (0, 1)^T$
- $\boldsymbol{x_3} = (A, B)^T = (1, 0)^T$
- $\boldsymbol{x_4} = (A, B)^T = (1, 1)^T$

What Boolean function does this neural network compute?

1. AND
2. OR
3. XOR
4. NAND

(c) [☑]What is $\frac{\partial L}{\partial W_{11}^{(2)}}$?

1. $2(y - out) \cdot n_1$
2. $2(out - y) \cdot n_1$
3. $2(y - out) \cdot n_2$
4. $2(out - y) \cdot n_2$

(d) [☑]What is $\frac{\partial L}{\partial W_{12}^{(1)}}$?

1. $2(out - y) \cdot W_{11}^{(2)} \cdot \sigma'(W_{11}^{(1)} A + W_{12}^{(1)} B) * B$
2. $2(out - y) \cdot W_{12}^{(2)} \cdot \sigma'(W_{11}^{(1)} A + W_{12}^{(1)} B) * B$
3. $2(out - y) \cdot W_{11}^{(2)} \cdot \sigma'(W_{11}^{(1)} A + W_{12}^{(1)} B) * A$
4. $2(out - y) \cdot W_{12}^{(2)} \cdot \sigma'(W_{11}^{(1)} A + W_{12}^{(1)} B) * A$

(e) [☑]Now, provide weights that implements the logical OR function $Y = x_1 \vee x_2$. Assume the weights can only take values -1, 0, or 1.

1. $$W^{(1)} \in \mathbb{R}^{2\times 2} = \begin{pmatrix} 0 & -1 \\ -1 & 1 \end{pmatrix},\ W^{(2)} \in \mathbb{R}^{1\times 2} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$
2. $$W^{(1)} \in \mathbb{R}^{2\times 2} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix},\ W^{(2)} \in \mathbb{R}^{1\times 2} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$
3. $$W^{(1)} \in \mathbb{R}^{2\times 2} = \begin{pmatrix} -1 & -1 \\ 0 & 1 \end{pmatrix},\ W^{(2)} \in \mathbb{R}^{1\times 2} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$
4. $$W^{(1)} \in \mathbb{R}^{2\times 2} = \begin{pmatrix} -1 & 1 \\ -1 & 0 \end{pmatrix},\ W^{(2)} \in \mathbb{R}^{1\times 2} = \begin{pmatrix} 1 & 1 \end{pmatrix}$$

Exercise 2: MLP vs CNN

You just took a picture which has $1920 * 1080$ pixels. Each pixel has a red, blue, and green value. You would like to design a neural network for this image.

(a) [☑]Calculate the number of parameters needed for a multilayer perceptron (MLP) that has only one hidden layer of 256 nodes and an output layer of 10 nodes.

(b) [☑]Now you would like to use a convolutional neural network (CNN). The network has 10 layers, each layer has 64 4×4 filters for EACH channel. Calculate the number of parameters needed for this CNN.

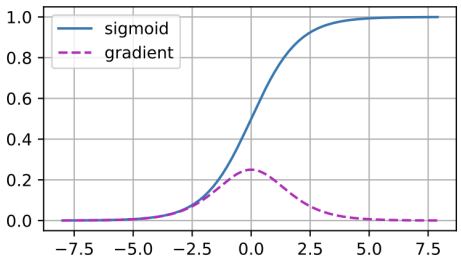
(c) [☑]Now consider a simple CNN with only one layer and one 4×4 filter per channel. What dimensions do the outputs of this layer have, if we choose a stride of 2 and apply 2-pixel padding to the input?

(d) Prove that there exists a fully connected linear layer of input size and output size that is functionally equivalent to the described convolutional network.

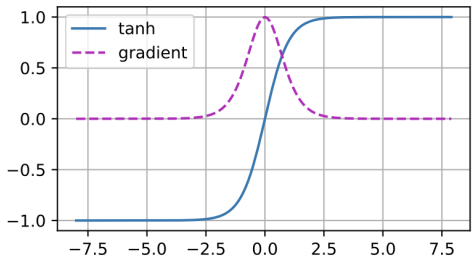
(e) Deduce that the family of functions written as convolutional layers is a subset of those written as fully connected linear layers.

Exercise 3: Activation Functions

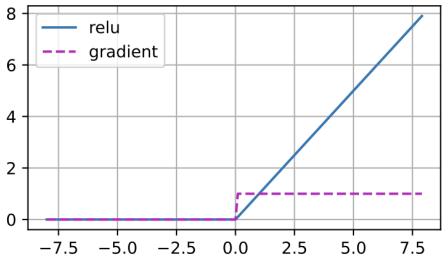
Take a look at the graphs of 4 activation functions and their derivatives:



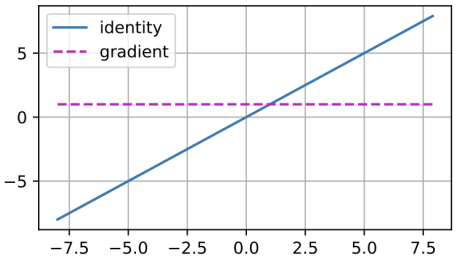
(a) sigmoid



(b) tanh



(c) relu



(d) identity

(a) [☑]Which activation function(s) are more prone to vanishing gradients?

1. sigmoid
2. tanh
3. ReLU
4. identity

(b) [☑]Which activation function(s) are differentiable?

1. sigmoid
2. tanh
3. ReLU
4. identity

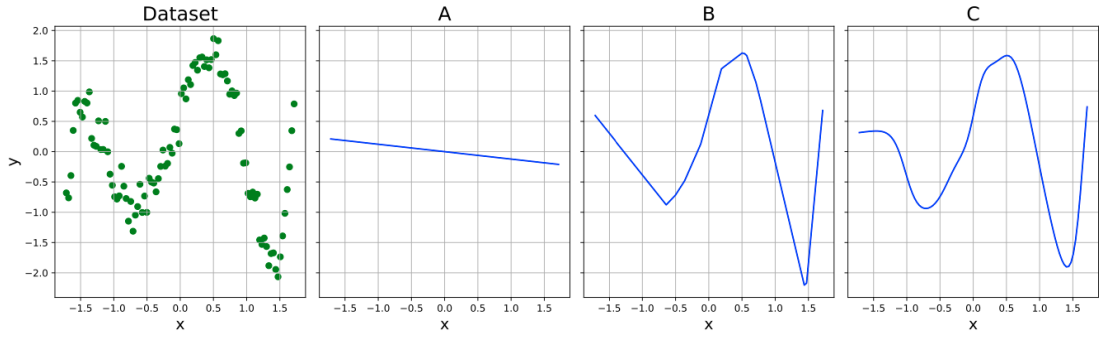
(c) [☑]Which activation function(s) are non-linear?

1. sigmoid
2. tanh
3. ReLU
4. identity

(d) [☑]Which activation function(s) are zero-centered?

1. sigmoid
2. tanh
3. ReLU
4. identity

(e) [☑]The figure below displays a training dataset and the learned function of 3 different neural networks that are trained on the dataset. The dataset consists of 100 scalar input-output pairs. All neural networks have one hidden layer with 20 units but differ in the choice of the activation function that are either the sigmoid, ReLu or identity function. Match the learned output function with the activation function that is used in the corresponding neural network.



1. (A, B, C) = (sigmoid, ReLU, identity)
2. (A, B, C) = (ReLU, identity, sigmoid)
3. (A, B, C) = (identity, ReLU, sigmoid)
4. (A, B, C) = (identity, sigmoid, ReLU)

Exercise 4: Gradient Descent

Recall the following Gradient Descent Methods:

(Batch) Gradient Descent: uses all examples in the training data.

Stochastic Gradient Descent: uses one randomly chosen example.

Mark the following as (T)rue or (F)alse:

(a) [☑] Batch Gradient Descent is suited for large number of training data. ☐ True ☐ False

(b) [☑] Batch Gradient Descent gives the global optimal solution given sufficient time. ☐ True ☐ False

(c) [☑] Stochastic Gradient Descent tends to escape local minima. ☐ True ☐ False

(d) [☑] Stochastic Gradient Descent reaches convergence faster than Batch Gradient Descent. ☐ True ☐ False

(e) [☑] Batch Gradient Descent is faster and less computationally expensive. ☐ True ☐ False