DL Exam

Full Name*	
Matriculation Number*	
Course of Studies*	

- You have 90 minutes to finish the exam.
- You are not allowed to use any electronic auxiliaries including calculators. If you have complex mathematical expressions, you may leave the fractions, logarithms, exponentials, etc. as is without having to calculate the exact numerical value.
- You are allowed to use exactly one DinA4 sheet of notes. (Back and front handwritten)
- The space below each question should be sufficient to write down your answer (more paper is available on demand).
- Please keep your handwriting legible and stick to the number of answers asked for. Illegible, ambiguous and multiple answers will be not graded.
  Use a permanent marker!
- Students who registered with "MeinCampus" can check their results after grading there. All others will be notified by the e-mail address linked with the StudOn course access.

☐ You can send me e-mails for upcoming events and open positions to the following e-mail address**:

I have visited the Deep Learning exercise in the following semester ***:	
I have read all the information above and entered required data truthfully:	
Signature	

*This data is required to identify you for the grading process.

**This entry is optional and has no effect on the exam whatsoever. Only fill it in if you want to be put on a mailing list from our lab.

*** This addresses in particular students who did the exercise in a previous semester. We want to ensure bonus points are transferred correctly from previous semesters.

Question	1	2	3	4	5	6	Exercise Bonus	Total	
Points	12	8	10	9	15	6	(6)	60	
Achieved									

1 Single Choice Questions (12P)

For each of the following questions, mark the **one correct choice**. Each question has **only one** correct option. No explanation is required.

Question 1.1 1 P.

Which of the following is a commonly used convolutional neural network architecture for image classification?

- ☐ ResNet
- ☐ LSTM
- ☐ Autoencoder
- ☐ GAN

Question 1.2 1 P.

Which of the following operations cannot introduce non-linearity in a neural network?

- ☐ Max pooling
- ☐ Rectified linear unit
- ☐ Convolution operation
- ☐ Softmax function

Question 1.3 1 P.

Which general statement about neural networks is not correct?

- ☐ Neural networks using only linear activations can be represented by a single matrix
- ☐ A single hidden layer can already be a universal function approximator
- ☐ The chain rule is necessary during inference
- ☐ A single perceptron cannot handle the XOR problem

Question 1.4 1 P.

Which statement about Sparse Autoencoders is correct?

- ☐ The encoder part is not trainable
- ☐ Sparsity is introduced by penalizing the weights
- ☐ The sparsity can be enforced by using an L1-norm
- ☐ It must contain fewer hidden units than input units

Question 1.5 1 P.

Which step is not correct when training Generative Adversarial Networks?

- ☐ The generator uses noise as an input
- ☐ The discriminator needs to be optimized before the generator
- ☐ The discriminator is trained in a supervised fashion
- ☐ GANS are trained using the Minimax Game

Question 1.6 1 P.

Which of the following statements about model capacity is correct? (Referring to the ability of neural network models to fit complex functions)

- ☐ The model capacity increases with an increased number of hidden layers
- ☐ During inference the model capacity decreases with the increase of dropout rate
- ☐ The larger the learning rate, the larger the model capacity
- ☐ The capacity of a model increases with the increase of the complexity of the non-linearity functions

Question 1.7 1 P.

What is the purpose of the Dropout regularization technique in deep learning?

- ☐ To prevent overfitting by randomly setting a percentage of activations to zero during training
- ☐ To prevent overfitting by strictly setting a fixed set of activations to zero during training to introduce sparsity
- ☐ To reduce the number of hidden neurons in layers
- ☐ To create small weights to stabilize the training process

Question 1.8 1 P.

How does Adam optimizer differ from other optimizers like SGD and RMSProp?

- ☐ Adam uses momentum while other optimizers do not
- ☐ Adam adapts the learning rate while other optimizers use fixed learning rates
- ☐ Adam uses both momentum and adaptive learning rate
- ☐ Adam uses neither momentum nor adaptive learning rate

Question 1.9 1 P.

What is the purpose of using data augmentation in deep learning?

- ☐ To increase the number of independent samples in the training set
- ☐ To improve the generalization performance of the network
- ☐ To reduce the number of input units in the network
- ☐ To balance the number of samples in different classes

Question 1.10 1 P.

What is the main disadvantage of using a large learning rate during training?

- ☐ The model takes longer to converge
- ☐ The model may get stuck in local minima
- ☐ The model may overfit to the training data
- ☐ The model may oscillate and fail to converge

Question 1.11 1 P.

Suppose you have built a neural network and decided to initialize the weights and biases to be zero. We can do this since the first hidden layer's neurons will always perform different computations from each other. Hence, their parameters will evolve independently, ensuring that the network can learn complex relationships no matter the initialization.

- ☐ True
- ☐ False

Question 1.12 1 P.

Self-supervised learning requires large amounts of annotated data to allow algorithms to achieve proper performance.

- ☐ True
- ☐ False

2 Short Answers (8P)

For each of the following questions, answer briefly in 1-2 sentences.

Question 2.1 2 P.

What are two advantages of using convolutional kernels rather than learning on the flattened image? Name two points.

Question 2.2 2 P.

How can the concept of Occlusion help when investigating important features/areas of your input data in a classification task. Briefly explain its main idea and how its output can be interpreted.

Question 2.3 2 P.

Explain the idea behind the Max-Pooling Layer and how the error is backpropagated during training.

Question 2.4 2 P.

(Mini-Batch) Stochastic Gradient Descent is an iterative method for optimizing an objective function. What problem might occur when using it? What other optimization tricks could you think of that might help to solve this problem?

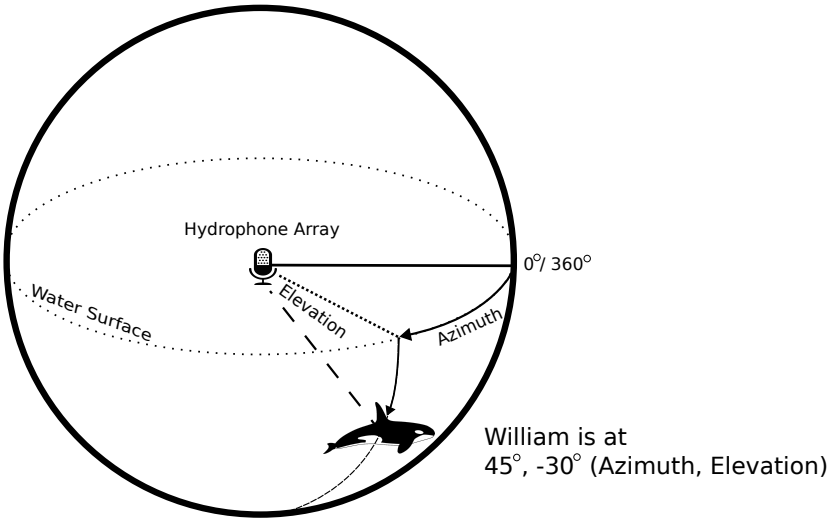


Figure 1: Explanation of Azimuth and Elevation of William the Whale

3 Regression, Loss, and Optimization - 10 Points

William the Whale, having recently been released from captivity, has been spotted at sea and, because you want to monitor his well-being and communication with other individuals, you want to devise a neural network to locate William using his calls and a hydrophone array (at least 2 hydrophones relatively close together) placed at the surface of the water. William, being a whale, is never above the hydrophone array but could be at any position around the array. You also know that you need to be as exact as possible and you therefore choose to tackle this situation as a regression problem instead of a classification problem and you need to find both the azimuth and elevation (see Figure 1) of William with respect to the hydrophone array. Hint: $\theta = \arctan\left(\frac{\sin(\theta)}{\cos(\theta)}\right)$

Question 3.1 1 P.

What is a common loss function that could be used in this instance for both azimuth and elevation prediction?

Question 3.2 1 P.

Conceptually, how can you alter the loss when performing the **elevation prediction** to account for the physical limitations of William?

Question 3.3 2 P.

When designing the network for **azimuth prediction**, which only has one **single output node**, the entire range of a circle is possible for your output. At certain positions, the loss calculated using the traditional loss function may be very large, even though the actual error between the ground truth and prediction is quite small. What is / are these positions and how would implement your **loss function** to account for this discrepancy?

Question 3.4 1 P.

How would you change azimuth label values to fit within a useful range?

Question 3.5 2 P.

To determine whether we are recording an orca or boat noises, we are training a classification network to distinguish these two classes. However boat noises are much more common and appear more often in the data set. Which strategies during training could you apply during training without changing the dataset. Name and briefly explain two strategies.

Question 3.6 3 P.

Based on your answers to the previous questions and assuming the **labels and predictions have already been normalized to be between 0 and 1**, implement the forward pass of the **azimuth loss** function.

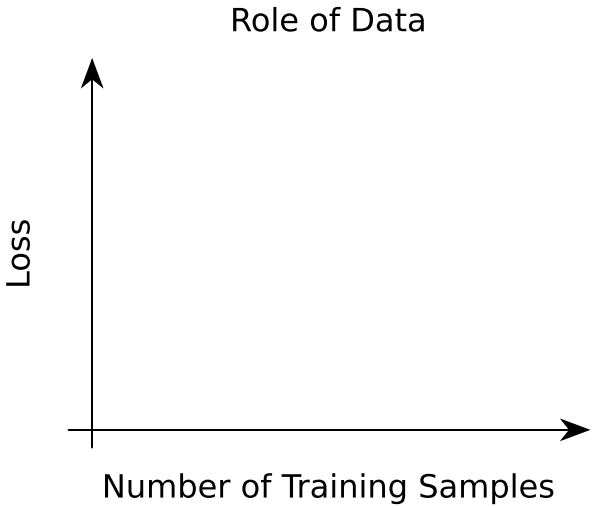
```
import numpy as np
class AzimuthLoss:

    def encode(self, tensor):
        """
        Encode a single tensor as described in the questions above
        (1 Point)
        """

    def forward(self, prediction_tensor, label_tensor):
        """
        Calculate the loss between prediction and ground truth
        (B x 1)-shaped tensors, where each value is between 0 and 1
        (2 Points)
        """

        #TODO
        loss =
        return loss
```

4 Regularization and Common Practices - 9 Points

Consider a binary image classification task with a convolutional neural network (CNN). Our first baseline model has a (test) accuracy of 72 %. The goal is to find a better model for this task.

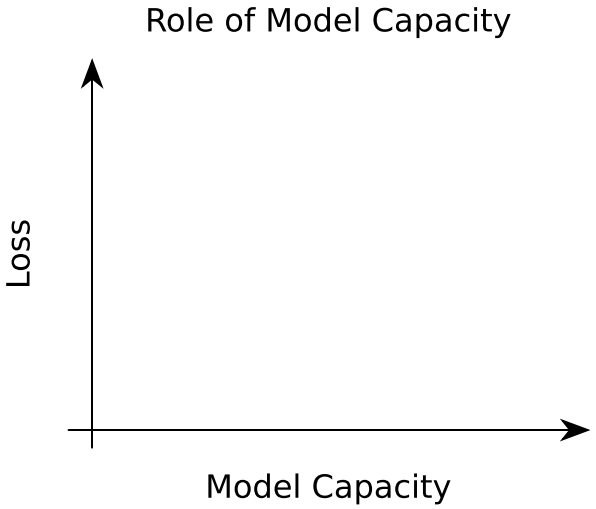
Question 4.1 1 P.

How does the number of independent training samples affect the loss on the training and test set? Please complement the graph with both loss curves and label them.



Question 4.2 1 P.

Consider a finite data set for now and complement the graph with the curves for training and test loss curves and label them.



Question 4.3 2 P.

Our current dataset is split into two sets: training and testing. We want to prevent overfitting on the training data. Additionally, we want to determine the best model for our test data. What steps should we take to achieve these goals?

Question 4.4 2 P.

In order to prevent overfitting, we experiment with regularization on our loss function. Our initial model had an accuracy of 72 %, but with L2 regularization, the accuracy increases to 81 %. Your original loss function is defined as $L(w, x, y)$, where w are the weights of the model, $x \in \mathbb{R}^n$ is the prediction vector of the model and $y \in \mathbb{R}^n$ are the corresponding labels. How do you alter the loss function $L(w, x, y)$ when introducing L2-Regularization? How does this affect the weight update during training, given a learning rate η? Complete the formulas below.

$L_2(w, x, y) =$

$w^{k+1} =$

Question 4.5 1 P.

In our current dataset, all images were taken using only one camera. However, we have now obtained additional data for our project, where all of the images were captured using different cameras, and our current model is not performing well on this new data. Discuss how this situation relates to the bias-variance trade-off.

Question 4.6 2 P.

Up until now, we have only assessed the performance of the model in terms of accuracy, and the final model achieved an accuracy of 84%. Our test dataset contained T=1000 samples, consisting of 450 true positives (TP) and 50 false negatives (FN). Calculate:

- True negatives
- Recall
- Specificity
- F1-score

5 LSTM and Backpropagation - 15 Points

Given is the following LSTM cell which receives an input x_t to predict the output $\hat{y}_t(x)$ using the two states c_{t-1} and h_{t-1}. It only contains the weight w_{1-4} and the activation functions σ and tanh. $\times$ is a multiplication, $+$ a summation and concat concatenates the h_{t-1} and x_t For a better overview, we defined multiple intermediate results f_t, i_t, $\tilde{c}_t$, o_t, a, b, d and e. The LSTM cell is visualized in Figure 2 on the next page.

Question 5.1 1 P.

Name an advantage of the LSTM cell compared to the Elman cell.

Question 5.2 2 P.

Describe what an element of a batch would be for a recurrent network (e.g. by using an example). Why does the required memory space increase with higher batch sizes during training? (Neglecting the costs for storing the initial batch.)

Question 5.3 4 P.

Define the LSTM outputs and states c_t, h_t and $\hat{y}_t$ as functions depending on the values w_i, b and activation functions σ and tanh (or by reusing already defined variables). Furthermore, define the derivative $\sigma'(x)$ and $\tanh'(x)$ depending on the original functions $\sigma(x)$ and $\tanh(x)$ respectively.

$$
\begin{aligned}
c_t &= \\
h_t &= \\
\hat{y}_t &= \\
\tanh'(x) &= \\
\sigma'(x) &=
\end{aligned}
$$

Question 5.4 8 P.

Derive the partial derivatives (on the next page) for the network listed below as general formulas depending on the above-defined variables. You may substitute already computed derivatives in the following derivations. Furthermore, you can assume that the gradients $\frac{\partial L}{\partial \hat{y}_t}$, $\frac{\partial L}{\partial h_t}$ and $\frac{\partial L}{\partial c_t}$ are given.

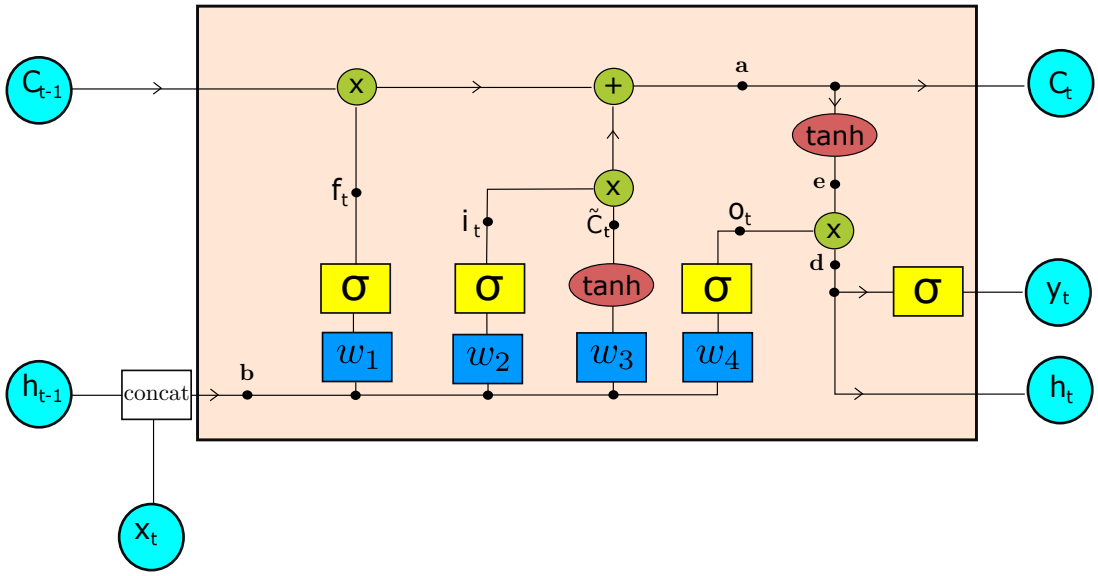


Figure 2: LSTM cell with intermediate steps.

$$\frac{\partial L}{\partial c_{t-1}} =$$

$$\frac{\partial L}{\partial d} =$$

$$\frac{\partial L}{\partial e} =$$

$$\frac{\partial L}{\partial o_t} =$$

$$\frac{\partial L}{\partial a} =$$

$$\frac{\partial L}{\partial \tilde{c}_t} =$$

$$\frac{\partial L}{\partial i_t} =$$

$$\frac{\partial L}{\partial f_t} =$$

$$\frac{\partial L}{\partial b} =$$

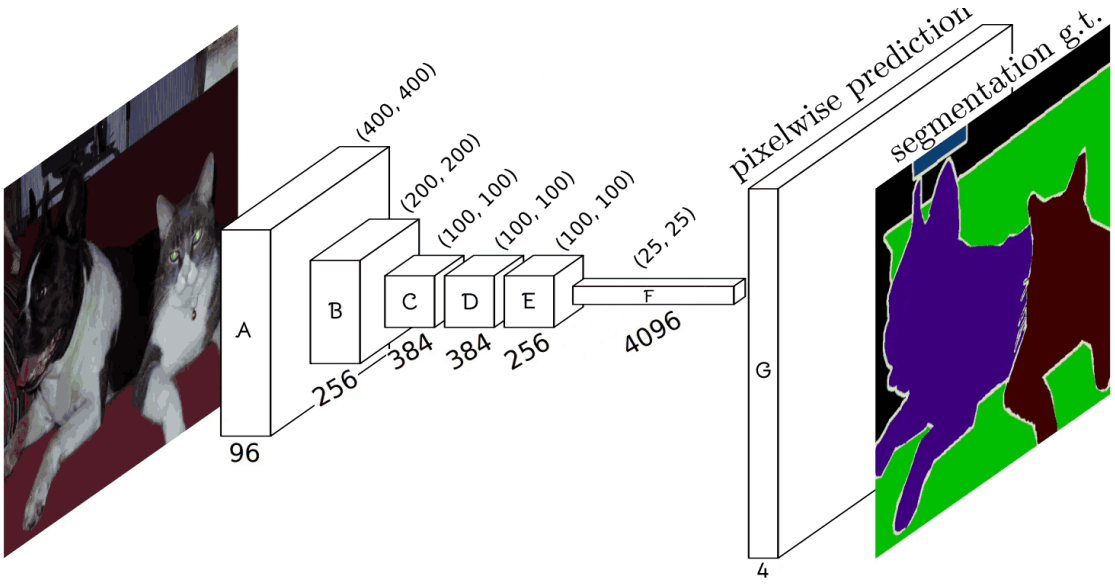


Figure 3: A Segmentation Network

6 Coding: Pytorch - 6 Points

You want to implement the following architecture in PyTorch. It consists of multiple convolutional layers with different strides and padding (specified in the Model constructor below), each followed by a ReLU activation.
Remark: All convolutional layers and max pooling layer in PyTorch are per default in "valid" mode, that means no padding at the borders is applied (padding value of 0). If the padding value is larger than 0, padding is applied before the convolution in height and width dimension on both sides of the image. The padding value (in the constructor) defines the padding width of one side of the image.

Question 6.1 4.5 P.

Implement the constructor for the given architecture in python using the PyTorch library.

```
import torch
from torch import nn
from torch.nn.functional import relu
from util import upsample

class Model(nn.Module):
    def __init__(self, input_channels, hidden_channels, num_classes):
        # 4.5 P
        super(Model, self).__init__()

        self.convA = nn.Conv2d(3, ___, kernel_size=3, padding=1, stride
            =2)
        self.convB = nn.Conv2d(___, ___, kernel_size=3, padding=1,
            stride=2)
        self.convC = nn.Conv2d(256, 384, kernel_size=5, padding=___,
```

```
        stride=___)
    self.convD = nn.Conv2d(384, 384, kernel_size=___, padding=3,
        stride=___)
    self.convE = nn.Conv2d(384, 256, kernel_size=1, padding=___,
        stride=1)
    self.convF = nn.Conv2d(256, 4096, kernel_size=3, padding=1,
        stride=___)
    self.upsample = upsample(4096, 4)
```

Question 6.2 1.5 P.

What is the correct order of the following training steps using pytorch?

A. loss.backward()
B. x = dataset()
C. loss = self._criterion.forward(y_hat, y)
D. self._optim.step()
E. y_hat = self._model.forward(x)
F. self._optim.zero_grad()

7 Notes