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	7	Exercise Bonus	Total
Points	6	8	10	14	9	6	7	(6)	60
Achieved									

1 Single Choice Questions (6P)

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 P.

Which statement about the Softmax function as the last layer in a neural network for a classification task is **false**?

☐ It assigns a probability to each class

☐ The individual outputs for one sample are dependent on each other

☒ It cannot be used during testing

☐ Generally produces a vector per sample for multi-class problems

Question 2. 1 P.

Which statement about initialization is **false**?

☒ Initialization is important for convex problems

☐ Using a small positive constant for the bias is helpful against the dying ReLU problem

☐ The gradient with respect to the weights is zero when the weights are initialized with zero

☐ Xavier initialization takes the number of input features into account

Question 3. 1 P.

In which scenario would you use under- or oversampling?

☐ When evaluating a segmentation task

☐ In the dropout layer to boost the influence of certain weights

☒ To counter class imbalance during training

☐ To create artificial data with variational autoencoders

Question 4. 1 P.

Which of the following techniques **cannot** be used to visualize important image areas for a neural network with a classification task?

- ☐ Saliency maps
- ☐ Occlusion
- ☐ Guided backpropagation
- ☒ Inception

Question 5. 1 P.

Which statement about reinforcement learning is **correct**?

- ☐ During training it is sufficient to exploit a known good action
- ☐ Following the exploitation principle we enforce new action sequences during the whole training process
- ☒ Exploration should be used during training to try out new action sequences
- ☐ Exploration should be used during testing

Question 6. 1 P.

Which statement about segmentation is correct?

- ☐ Instance segmentation can only find one instance of an object in the image
- ☒ Semantic segmentation can differentiate between different classes in an image
- ☐ Segmentation tries to find bounding boxes around the object of interest
- ☐ Semantic segmentation treats multiple objects of the same class as multiple entities

2 Multiple Choice Questions (6P)

For each of the following questions, mark **all choices** that apply. Each question has **at least** one correct option unless otherwise stated. No explanation is required. You will only get points if all correct options are marked.

Question 1. 2 P.

What is a suitable measure to evaluate the final performance of your fine-tuned neural network?

- ☐ Accuracy on the validation set
- ☐ Mean precision on a combination of training, validation and test set
- ☒ Receiver operating characteristics curve on the test set
- ☐ Recall on the training set
- ☒ F1-Score on the validation set

Question 2. 2 P.

What are advantages of making a neural network deeper?

- ☒ Exponential feature reuse
- ☐ Less memory consumption
- ☒ Increasingly abstract features
- ☐ Enable training with larger batch size
- ☒ Ability to approximate increasingly complex functions

Question 3. 2 P.

Which network architectures contain convolutional layers and skip connections?

- ☒ U-Net
- ☐ LeNet
- ☐ VGG16
- ☐ LSTM
- ☒ ResNet

Question 4. 2 P.

What are problems associated with the simple Elman cell?

- ☒ Small weights are multiplied multiple times and may lead to a vanishing gradient
- ☒ Long term memory loss
- ☐ Many to many mapping is not possible
- ☐ Skip connections lead to short- and long-term memory loss
- ☐ Bounded non-linear activation functions

3 Short Answers (10 points)

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

Question 1. 1 P.

The model only needs to classify everything to the dog to achieve 80%. (1P for explanation)

Question 2. 3 P.

the following are possible answers

- the dataset is unbalanced (0.5p)
  - use equal numbers of images for both classes (1p)
  - (or) use a weighting loss function to compensate for cat (1p)
- fully connected network assume pixels to be independent (0.5p)
  - use convolutional neural network (or other suitable network architectures) (1p)

Question 3. 2 P.

- use validation accuracy to train the network and define when to stop training instead of training accuracy (1p)
- use regularization methods to prevent overfitting (1p)

Question 4. 2 P.

- To introduce non-linearity such that the network can model complex non-linear functions instead of just a linear function (1p)
- Saturation/vanishing gradient. Gradient only exists between -1 and 1. Gradient approximates 0 for larger input values. (1p)

Question 5. 2 P.

- momentum. (1p)
- use previous gradient directions to accelerate the training and become more robust against local minima. (1p)

4 Recurrent Networks and Backpropagation (14 P)

Question 1. 2 P.

$$
\begin{aligned}
f(x_t, h_{t-1}) &= (h_{t-1} + x_t) \cdot w + h_{t-1} && (1P) \\
s_t &= h_{t-1} + x_t && (0.5P) \\
o_t &= (h_{t-1} + x_t) \cdot w && (0.5P)
\end{aligned}
$$

Question 2. 8 P.

$$
\begin{aligned}
\frac{\partial L}{\partial \hat{y}_t} &= 2(\hat{y}_t - y_t) \\
\frac{\partial L}{\partial o_t} &= \frac{\partial \hat{y}_t}{\partial o_t} \cdot \frac{\partial L}{\partial \hat{y}_t} + \frac{\partial h_t}{\partial o_t} \cdot \frac{\partial L}{\partial h_t} = \frac{\partial L}{\partial \hat{y}_t} + \frac{\partial L}{\partial h_t} \\
\frac{\partial L}{\partial w_t} &= \frac{\partial o_t}{\partial w_t} \cdot \frac{\partial L}{\partial o_t} = (h_{t-1} + x_t) \cdot \frac{\partial L}{\partial o_t} (= s_t \cdot \frac{\partial L}{\partial o_t}) \\
\frac{\partial L}{\partial w} &= \sum_t^T \frac{\partial L}{\partial w_t} \\
\frac{\partial L}{\partial s_t} &= \frac{\partial L}{\partial o_t} \cdot \frac{\partial o_t}{\partial s_t} = \frac{\partial L}{\partial o_t} \cdot w \\
\frac{\partial L}{\partial x_t} &= \frac{\partial L}{\partial s_t} \cdot \frac{\partial s_t}{\partial x_t} = \frac{\partial L}{\partial s_t} \\
\frac{\partial L}{\partial h_{t-1}} &= \frac{\partial L}{\partial o_t} + \frac{\partial L}{\partial s_t}
\end{aligned}
$$

Question 3. 1 P.

It would change the function while deriving it, leading to incorrect gradients for the following time steps.

Question 4. 3 P.

Problem: Single parameter update is very expensive (1P)
Solution: Truncated backpropagation through time. (1p)
Explanation: Split long sequences into smaller batches (0.5P) and let those batches overlap (0.5P)
(Writing down the TBPTT Algorithm is also fine as an explanation).

5 Unsupervised learning (9P)

Question 1. 3 P.

1. Draft of an AE. (0.5P)
2. Encoder - Decoder (0.5P each)
3. Input = x, Output = x'
4. $L_2(x, x') = \|\mathbf{x} - \mathbf{x}'\|_2^2$ (0.5p for loss function 0.5p input output where output is reconstruction of input)
5. Vector of reduced dimensionality in the middle, y = f(x), where f is the encoder (0.5p)

Question 2. 2 P.

1. The non-linearities in the autoencoder.
2. An autoencoder network trained without non-linearities and with L2-Loss learns a generalization of the PCA.

Question 3. 2 P.

Corrupt the input depending on a noise model (1p), e.g., Gaussian, and learn the original image (1p).

Question 4. 2 P.

Variational autoencoders compress the input information into a constrained multivariate latent distribution from which the output can be reconstructed. (1p)
Therefore you have to know the distribution in advance to model the distribution (e.g. Gaussian). In autoencoders you do not assume a distribution in the latent space. (1p)

6 Coding: Framework (6P)

Question 1. 6 P.

Solution: (multiple solutions are possible)

```
def forward(self, prediction_tensor, label_tensor):
    # +1.5 applying correct formula
    # + 0.5 extracting batch-size
    # +1.5 storing information required in backward
    # and using it in backward (1/0.5)

    batch_size = prediction_tensor.shape[0]
    factor = 1 / np.sqrt(batch_size+1)
    self.prediction_tensor = prediction_tensor
    self.label_tensor = label_tensor
    loss = np.square(prediction_tensor - label_tensor)
    loss = np.sum(loss)
    loss *= factor
    return loss

def backward(self):
    # +1.5 applying correct derivative and implementation

    batch_size = self.prediction_tensor.shape[0]
    factor = 1/np.sqrt(batch_size+1)
    error = np.subtract(self.prediction_tensor, self.label_tensor)
    return 2*factor * error
```

7 Coding: Pytorch (7P)

Question 1. 7 P.

```
# A. __3__.   B. __1__.   C. _128_.   D. __3__.   E. __2__.
# F. __3__.   G. __64_.   H. __5__.   I. __2__.   J. __1__.
# each 0.5 p, in total 5 p

def forward(self, x):
    convB1 = F.relu(self.convB1(x))         # (0.5 p)
    convB2 = F.relu(self.convB2(convB1))    # (0.5 p)
    convC1 = F.relu(self.convC1(x))         # (0.5 p)
    maxPool = self.maxPool(x)               # (0.5 p)

    output = torch.cat([convA2, convB3, convC2, convD], dim=1)
    return output
```