DL Solution SS22

I can neither confirm nor deny details of any operation without the Secretary's approval.

1 Single Choice Questions (10P)

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 techniques is NOT a common practice to tackle class imbalance?

☐ Weighting loss with inverse class frequency

☐ Data augmentation

☐ Oversampling

☒ Batch normalization

Question 1.2 1 P.

What is ensembling?

☐ A visualization technique where different areas in the input image are occluded

☐ A network with multiple outputs for different tasks

☐ Parallel paths in a network each including a different kernel size respectively to let the network decide on the best kernel parameters

☒ Multiple networks trained on the same task where the prediction is a majority vote

Question 1.3 1 P.

Which option can be used to create synthetic samples?

☐ The discriminator part of a Generative Adverserial Network

☐ The encoder part of a Variational Autoencoder

☒ The decoder part of a Variational Autoencoder

☐ The encoder part of a Generative Adverserial Network

Question 1.4 1 P.

Imagine that you want to generate a sequence of words with your RNN after you finished the training. Which of the following sampling strategies does not exist for that purpose?

☐ Greedy sampling

☐ Beam sampling

☒ Recognition sampling

☐ Random sampling

Question 1.5 1 P.

Which statement is false when implementing a Convolutional layer?

☐ The number of kernels determines the number of output channels

☐ Stride bigger than one reduces the image size

☒ A valid convolution outputs the image of the same size as its input

☐ The number of parameters/weights inside the kernel is independent from the image size

Question 1.6 1 P.

Why does initialization matter for the optimization of deep neural networks?

☒ Because the problem is often non-convex

☐ To avoid too many parameters

☐ So we always end up in the same local minimum

☐ Because the problem is convex

Question 1.7 1 P.

What's the main tasks of Object Detection

☒ Finding Bounding boxes and Classification

☐ Search for boundaries between different objects

☐ Increase the image resolution of the regions of interest

☐ Fill out the missing part of the image

Question 1.8 1 P.

Which statement about YOLO (You Only Look Once) and Fast R-CNN (Regional Convolutional Neural Network) is True?

- ☐ YOLO produces many object candidate suggestions that are passed to the CNN
- ☒ YOLO combines bounding box prediction and classification in one network
- ☐ Fast R-CNN is in general faster
- ☐ Fast R-CNN can do real time detection while YOLO can not

Question 1.9 1 P.

It is important to create your data set properly to avoid misclassifications during the testing. Which of the following examples is not a confound?

- ☒ Food classification task: The food images were taken with two different cameras regardless of their label
- ☐ Traffic object detection task: training images of cars were taken in daylight, while images of pedestrians were taken on rainy days
- ☐ Clothing classification task: T-shirts are worn only be females while hoodies are only worn by males
- ☐ Language classification: spanish voice samples were recorded with a different microphone than the italian samples

Question 1.10 1 P.

When handling an image classification task, you might only have a few options to tackle the problem but luckily, you have a model that was already trained on similar task. Which of the following methods could be used to make use of this trained model?

- ☒ Freeze all feature extraction layers and retrain the classification layers at the end
- ☐ Use the whole trained model and only retrain the input layer
- ☐ Only retrain the feature extraction layers and use the original classification layers
- ☐ Assess each layers and pick some layers to form a new model

2 Short Answers (9P)

Question 2.1 1 P.

Solution 2.1
cross-correlation is a convolution with flipped kernel and vice-versa.

Question 2.2 1 P.

Solution 2.2
variety of functions it can approximate / number of parameters

Question 2.3 2 P.

Solution 2.3

- ReLU is not zero-centered
- Initialization and input distribution might not be normalized
- Deeper nets $-->$ amplified effect

Question 2.4 1 P.

Solution 2.4
It contains time-dependency / hidden states as time-dependent state between each time / it contains hidden states as "short memories" to connect between each time

Question 2.5 1 P.

Solution 2.5
it contains long term memory / It has cell state that serves as long term memory so we can model longer time series / Easier to detect long-term dependencies by using cell states

Question 2.6 1 P.

Solution 2.6
Pixel Accuracy / Mean Pixel Accuracy / Mean Intersection over Union / Frequency Weighted Intersection over Union (FWIoU)

Question 2.7 1 P.

Solution 2.7
activations generated by kernels

Question 2.8 1 P.

Solution 2.8
(each could give one point, in total one point)

- 1x1 convolutions simply calculate inner products at each position
- Simple and efficient method to decrease the size of a network
- Learns dimensionality reduction, e.g., can reduce redundancy in your feature maps

3 Feed Forward Network (9P)

Question 3.1 1 P.

Why do you use ReLU activations instead of Sigmoid activations after each fully connected layer?

Solution 3.1

because Sigmoid function saturates / ReLU does not saturate in positive area. (1)

Question 3.2 2 P.

Imagine an image size of (X, X).

Solution 3.2

$(X^2) * Z$ (1 P.)
$(K^2) * N$ (1 P.)

Question 3.3 2 P.

Solution 3.3

Disadvantage fully connected:

- Highly correlated / not independent from each other
- Scale dependent / different size
- Intensity variations / color variation

Advantages convolutional:

- Great feature extraction
- Local connectivity -¿ filters
- Translational equivariance

Question 3.4 1 P.

Solution 3.4

Flatten, pooling etc. 1 P.

Question 3.5 1 P.

Solution 3.5

A simpler way to understand what the bias is: it is somehow similar to the constant b of a linear function y = ax + b
It allows you to move the line up and down to fit the prediction with the data better.
Without b, the line always goes through the origin (0, 0) and you may get a poorer fit.

Question 3.6 2 P.

Solution 3.6

- Sigmoid function (1).
- Each output entry is between 0 and 1. Softmax function will make all entries sum up to 1. (1)

4 Recurrent Networks and Backpropagation (9P)

Question 4.1 3 P.

Solution 4.1
Half point each:

$$
\begin{aligned}
l &= x \cdot w + b \\
m &= x \cdot w + b + x \\
\sigma(x) &= \frac{1}{1 + \exp(x)} \\
\sigma'(x) &= \sigma(x) \cdot (1 - \sigma(x)) \\
f_1(x) &= \sigma(x + wx + b) \\
f_2(x) &= l = w \cdot x + b
\end{aligned}
$$

Question 4.2 6 P.

Solution 4.2
One point each:

$$
\begin{aligned}
\frac{\partial L}{\partial m} &= \frac{\partial L}{\partial \hat{y}_1}\frac{\partial \hat{y}_1}{\partial m} = \frac{\partial L}{\partial \hat{y}_1}\sigma'(x + wx + b) \\
\frac{\partial L}{\partial l} &= \frac{\partial L}{\partial \hat{y}_2} + \frac{\partial L}{\partial m} \\
\frac{\partial L}{\partial b} &= \frac{\partial L}{\partial l}\frac{\partial l}{\partial b} = \frac{\partial L}{\partial l} \cdot 1 \\
\frac{\partial L}{\partial w} &= \frac{\partial L}{\partial l} \cdot x \\
\frac{\partial L}{\partial k} &= \frac{\partial L}{\partial l}\frac{\partial l}{\partial k} = \frac{\partial L}{\partial l} \cdot w \\
\frac{\partial L}{\partial x} &= \frac{\partial L}{\partial k} + \frac{\partial L}{\partial m}
\end{aligned}
$$

5 Reinforcement learning (9P)

Question 5.1 1 P.

Solution 5.1

action: going from on student to the next. (0.5p)
reward: points for finishing a request. (0.5p)

Question 5.2 2 P.

Solution 5.2

Y maximum points (1p), X greedy algorithm (1p).

0	1	0	0	2
Y	Y	1	0	0
0	Y	0	X	0
1	F	X	X	1
0	0	0	1	0

Question 5.3 3 P.

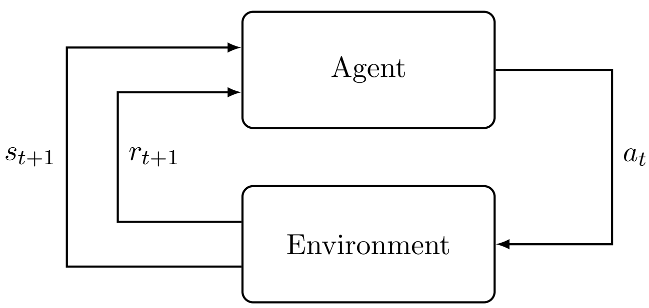
Solution 5.3

Only exploitation of the next action is used to get maximum reward which does not lead to a maximum reward over multiple steps. (Cannot look ahead, no state transition included) (1P)
Exploitation: use the known good action (0.5 P)
Exploration: try out new actions (0.5 P)
Epsilon greedy (1.P):

$$\pi(a) = \begin{cases} 1-\epsilon & \text{if}\, a = \max\limits_{a} Q_t(a) \\ \epsilon/(n-1) & \text{else} \end{cases}$$

Question 5.4 3 P.

Solution 5.4



6 Coding: Optimization (7P)

$$v^{(k)} = \mu v^{(k-1)} - \eta \nabla L(w^{(k)}) \tag{1}$$

$$w^{(k+1)} = w^{(k)} - \mu v^{(k-1)} + (1+\mu) v^{(k)} \tag{2}$$

Question 6.1 7 P.

```
import numpy as np

class NAG():

    def __init__(self, learning_rate, momentum=0.9):
        self.learning_rate = learning_rate  (0.5p)
        self.momentum_gradient = None (0.5p)
        self.momentum = momentum    (0.5p)
        # Initialize momentum gradient with None or zero_like tensor.


    def calculate_update(self, weight_tensor, gradient_tensor):

        if self.momentum_gradient is None:
            self.momentum_gradient = np.zeros_like(weight_tensor)   (1p
              )

        previous_direction = self.momentum_gradient     (1p)
        self.momentum_gradient = self.momentum * previous_direction -
           learning_rate  * gradient_tensor          (1.5p, decide by
           yourself)
        # Compute the current momentum gradient -> 2 P.
        return weight_tensor - self.momentum * previous_direction + (1+
           self.momentum)*self.momentum_gradient    (2p)
```

7 Coding: Pytorch (7P)

Question 7.1

7 P.

```
class Inception(nn.Module):
    def __init__(self):
        self.convA2 = nn.Conv2d(64, 64, kernel_size=3, padding=1,
           stride=1)
        self.down1 = nn.Conv2d(64, 128, kernel_size=3, padding=1,
           stride=2)

        self.convC = nn.Conv2d(256, 128, kernel_size=3, padding=1)

    def forward(self, x):

        convA1 = F.relu(self.convA1(x))

        # TODO
        convA2 = F.relu(self.convA2(convA1))    (0.5p)
        down1 = F.relu(self.down1(convV2))      (0.5p)

        convB = F.relu(self.convB(down1))       (0.5p)
        down2 = F.relu(self.down2(convB))        (0.5p)

        conv_up1 = F.relu(self.conv_up1(down2))  (0.5p)
        conv_all = torch.cat([_convB_, _conv_up1_], dim=1)   (0.5p)

        convC = F.relu(self.convC(conv_all))        (0.5p)

        conv_up2 = F.relu(self.conv_up2(convC))      (0.5p)
        convD1 = F.relu(self.convD1(conv_up2))       (0.5p)
        convD2 = F.relu(self.convD2(convD1))         (0.5p)
```