Chair of Visual Computing & Artificial Intelligence
Department of Informatics
Technical University of Munich

Esolution
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Introduction to Deep Learning

Exam:	IN2346 / Endterm	**Date:**	Tuesday 11th August, 2020
Examiner:	Prof. Leal-Taixé and Prof. Nießner	**Time:**	08:00 – 09:30

	P 1	P 2	P 3	P 4	P 5	P 6	P 7	P 8
I								

Left room from ______________ to ______________

from ______________ to ______________

Early submission at ______________

Notes ____

Chair of Visual Computing & Artificial Intelligence
Department of Informatics
Technical University of Munich

Endterm

Introduction to Deep Learning

Prof. Leal-Taixé and Prof. Nießner
Chair of Visual Computing & Artificial Intelligence
Department of Informatics
Technical University of Munich

Tuesday 11th August, 2020
08:00 – 09:30

Working instructions

- This exam consists of **20 pages** with a total of **8 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 90 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources: **none**
- Do not write with red or green colors nor use pencils.
- Physically turn off all electronic devices, put them into your bag and close the bag.
- If you need additional space for a question, use the additional pages in the back and properly note that you are using additional space in the question's solution box.

Problem 1 Multiple Choice Questions: (18 credits)

- For all multiple choice questions any number of answers, i.e. either zero (!), one, all or multiple answers can be correct.
- For each question, you'll receive 2 points if all boxes are answered correctly (i.e. correct answers are checked, wrong answers are not checked) and 0 otherwise.

How to Check a Box:

- Please **cross** the respective box: ☒ (interpreted as **checked**)
- If you change your mind, please **fill** the box: ■ (interpreted as **not checked**)
- If you change your mind again, please place a cross to the left side of the box: ✗ ■ (interpreted as **checked**)

a) Which of the following statements regarding successful ImageNet-classification architectures are correct?

- ☐ ResNet18 has more parameters than VGG16.
- ☒ AlexNet uses filters of different kernel sizes.
- ☒ InceptionV3 uses filters of different kernel sizes.
- ☐ VGG16 only uses convolutional layers.

b) You train a neural network and the loss diverges. What are reasonable things to do? (check all that apply)

- ☒ Decrease the learning rate.
- ☐ Add dropout.
- ☐ Increase the number of parameters.
- ☒ Try a different optimizer.

c) What is the correct order of operations for an optimization with gradient descent?

(a) Update the network weights to minimize the loss.
(b) Calculate the difference between the predicted and target value.
(c) Iteratively repeat the procedure until convergence.
(d) Compute a forward pass.
(e) Initialize the neural network weights.

- ☐ bcdea
- ☐ ebadc
- ☐ eadbc
- ☒ edbac

d) Consider a simple convolutional neural network with a single convolutional layer. Which of the following statements is true about this network?

- ☐ It is rotation invariant.
- ☒ It is translation invariant.
- ☐ All input nodes are connected to all output nodes.
- ☐ It is scale-invariant.

e) Which of the following activation functions can lead to vanishing gradients?

- [x] Tanh.
- [ ] ReLU.
- [x] Sigmoid.
- [ ] Leaky Relu.

f) Logistic regression (check all that apply).

- [ ] Has a discrete output space.
- [x] Can be seen as a 1-layer neural network.
- [x] Uses cross-entropy loss.
- [x] Allows to perform binary classification.

g) A sigmoid layer

- [ ] cannot be used during backpropagation.
- [ ] has a learnable parameter.
- [ ] maps surjectively to values in (-1, 1), i.e., hits all values in that interval.
- [x] is continuous and differentiable everywhere.

h) Your training error does not decrease. What could be wrong?

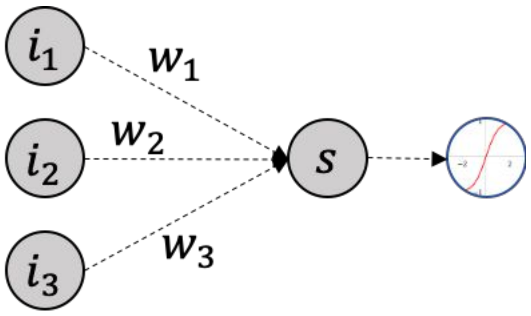
- [x] Learning rate is too high.
- [x] Too much regularization.
- [ ] Dropout probability not high enough.
- [x] Bad initialization.

i) Which of the following have trainable parameters? (check all that apply)

- [ ] Leaky ReLU
- [x] Batch normalization
- [ ] Dropout
- [ ] Max pooling

Problem 2 Activation Functions and Weight Initialization (8 credits)

For your first job, you have to set up a neural network but you have some issue with its weight initialization. You remember from your I2DL lecture that you can sample the weights from a zero-centered normal distribution, but you can't remember which variance to use. Therefore, you set up a small network and try some numbers. You initialize the weights one time with Var(**w**) = 0.02 and one time with Var(**w**) = 1.0:



Inputs:

- $i_1 = 2, i_2 = -4, i_3 = 1$

Var(**w**) = 0.02:

- $w_1 = 0.05, w_2 = 0.025, w_3 = -0.03$

Var(**w**) = 1.0:

- $w_1 = 1.0, w_2 = 0.5, w_3 = 1.5$

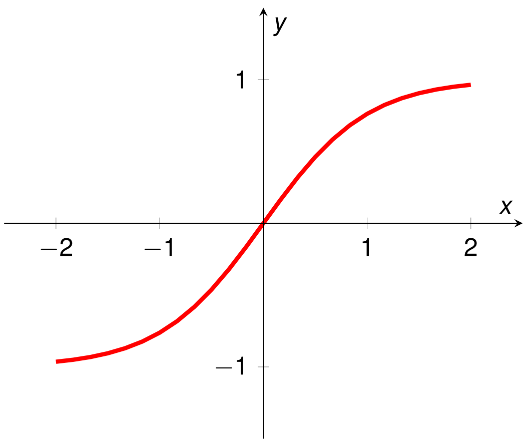
0 ☐
☐
1 ☐
☐
2 ☐

a) Compute a forward pass for each set of weights and draw the results of the linear layer in the Figure of the tanh plot. You don't need to compute the tanh.

Var(**w**) = 0.02: s = $(2 * 0.05) + (-4 * 0.025) + (1 * (-0.03)) = -0.03$ (0.5p)

Var(**w**) = 1.0: s = $(2 * 1.0) + (-4 * 0.5) + (1 * 1.5)) = 1.5$ (0.5p)

0.5p for each of the calculated numbers drawn correctly into the plot.
Note: Inaccurate drawing is tolerated, but positive instead of negative isn't



b) Using the results above, explain what problems can arise during backpropagation of deep neural networks when initializing the weights with too small and too large variance. Also, explain the root of these problems.

0 1 2

Too small variance: The output in deeper layers is close to zero. Therefore, the gradient w.r.t the weights also becomes very small (vanishing gradient).
(0.5p for explanation, 0.5p for small gradients, i.e. only "vanishing gradient is 0.5p. Needs to mention gradients)
Too large variance: The tanh activation function saturates. Therefore, the gradient w.r.t the weights becomes very small (vanishing gradient).
Note: again, only vanishing gradient is 0.5p

c) Which initialization scheme did you learn in the lecture that tackles these problems? What does this initialization try to achieve in the activations of deep layers of the neural network?

0 1 2

Xavier initialization (1p)
Note: 0.5p for only writing down the formula
The goal is to keep the variance of the output is the same as the input. (1p)

d) After switching from tanh to ReLU activation functions, one of your initial problems occurs again. Why does this happen? How can you modify the initialization scheme proposed in c) to adjust it for this new non-linearity?

0 1 2

ReLU "kills" half of the outputs. Therefore, the output variance of a layer is halved. (1p)
Note: only mentioning that $relu(x) = 0$ for $x < 0$ is only 0.5p
Solution: Multiply Xavier initialization with 2/He initialization (1p)
Note: only name gives 0.5p

Problem 3 Batch Normalization and Computation Graphs (6 credits)

For an input vector **x** as well as variables γ and β the general formula of batch normalization is given by

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - E[\mathbf{x}]}{\sqrt{Var[\mathbf{x}]}}$$
$$\mathbf{y} = \gamma\hat{\mathbf{x}} + \beta.$$

0 1

a) Why would one want to apply batch normalization in a neural network?

Prevent covariate shift/ Mimics the normalization of data for each layer to receive more stable inputs (1p)

Note: Only 0.5p for stable gradient, regularization, able to train deeper network, stabilize training, faster training ...

0 1

b) Why are γ and β needed in the batch normalization formula?

Allows the network to "undo"/scale+shift the normalization. (1p)
Note: 0.5p if only one of scaling and shift are mentioned
Stating that gamma and beta are hyperparameters does not get any points

0 1 2

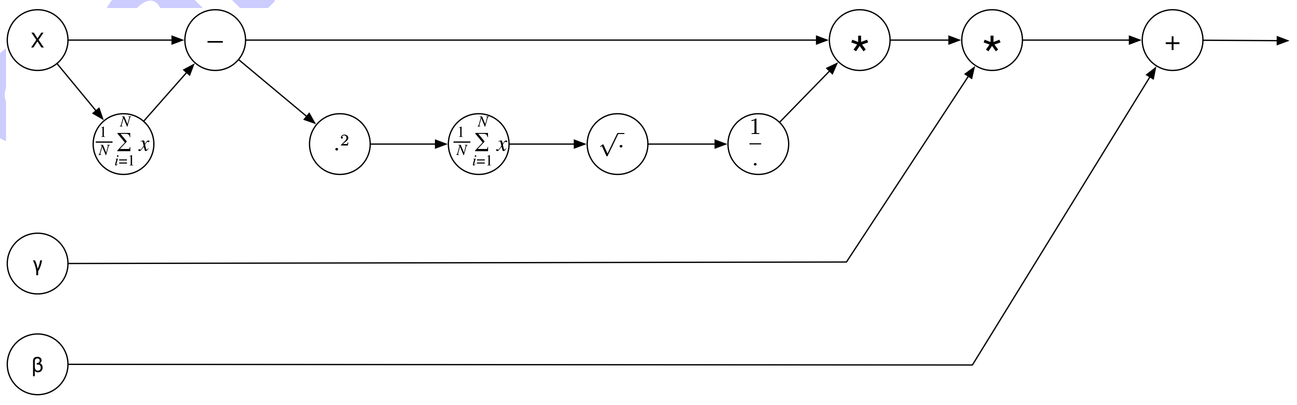
c) How is a batch normalization layer applied at training (1p) and at test (1p) time?

Train time: **Calculate the mean and variance computed from the mini-batch** (0.5p) and **store a weighted average** across training mini-batches for the update at test time (0.5p)

Test time: Use exponentially weighted / moving / running average mean and variance that was computed at training time on the test samples (1p)
Note: 0.5p only if just simply mentioned acquiring statistics from the training set, but failed to illustrate which method is needed to acquire mean and var, i.e. exponentially weighted average

0 1 2

d) Computational graph of a batch normalization layer. Fill out the nodes (circles) of the following computational graph. Each node can consist of one of the following operations $+$, $-$, $*$, ^{2}, $\sqrt{\cdot}$, $\frac{1}{\cdot}$.



Note: -1p for each wrong operator

Problem 4 Convolutional Neural Networks and Receptive Field (12 credits)

A friend of yours asked for a quick review of convolutional neural networks. As he has some background in computer graphics, you start by explaining previous uses of convolutional layers.

a) You are given a two dimensional input (e.g., a grayscale image). Consider the following convolutional kernels

0 1 2

$$C_1 = \frac{1}{9} \cdot \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \qquad C_2 = \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix}.$$

What are the effects of the filter kernels C_1 and C_2 when applied to the image?

C_1: Local/box (0.5p) blur/smoothing (0.5p) kernel. Note: If only mean/arg is mentioned instead of blur then 0.5p
C_2: vertical (0.5p) edge detector (0.5p)

After showing him some results of a trained network, he immediately wants to use them and starts building a model in Pytorch. However, he is unsure about the layer sizes so you quickly help him out.

b) Given a Convolution Layer in a network with 5 filters, filter size of 7, a stride of 3, and a padding of 1. For an input feature map of $26 \times 26 \times 26$, what is the output dimensionality after applying the Convolution Layer to the input?

0 1 2

$8 \times 8 \times 5$ (2p)   1p for only kernel size computation $\frac{26-7+2\cdot1}{3} + 1 = 7 + 1 = 8$

c) You are given a convolutional layer with 4 filters, kernel size 5, stride 1, and no padding that operates on an RGB image.

0 1 2

1. What is the shape of its weight tensor?
2. Name all dimensions of your weight tensor.

Shape: (3, 4, 5, 5) (1p)
Reasoning: input size/rgb channels, output size/channels, width, height (1p)

Note: -1p if only 3 dimensions are mentioned, -1p if 5's are simply described as filter size

Now that he knows how to combine convolutional layers, he wonders how deep his network should be. After some thinking, you illustrate the concept of receptive field to him by these two examples. For the following two questions, consider a grayscale 224x224 image as network input.

d) A convolutional neural network consists of 3 consecutive 3×3 convolutional layers with stride 1 and no padding. How large is the receptive field of a feature in the last layer of this network?

0 1

1x1 → 3x3 → 5x5 → 7x7 (1p)
Note: -0.5p if no tuple

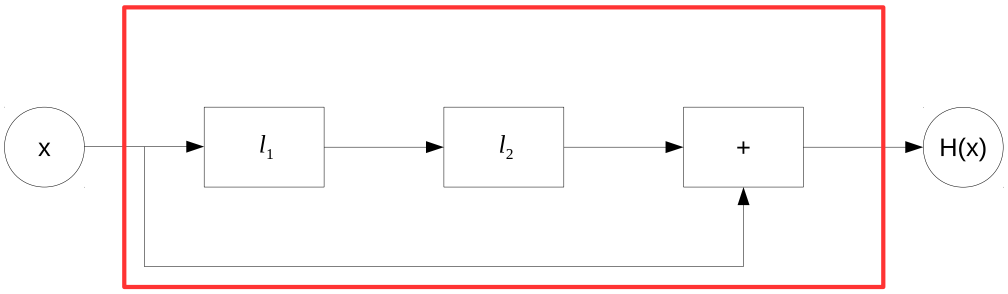
0 1 2

e) Consider a network consisting of a single layer.

1. What layer choice has a receptive field of 1?
2. What layer has a receptive field of the full image input?

1x1 convolution or identity (1p)
fully connected or conv/pooling layer with kernel size equals full input size (224x224) (1p)
If the answer is reasonable but incomplete: -0.5p

Blindly, he stacks 10 convolutional layers together to solve his task. However, the gradients seem to vanish and he can't seem to be able to train the network. You remember from your lecture that ResNet blocks were designed for these purposes.



0 1 2

f) Draw a ResNet block in the image above (1p) containing two linear layers, which you can represent by l_1 and l_2. For simplicity, you don't need to draw any non-linearities. Why does such a block improve the vanishing gradient problem in deep neural networks (1p)?

1p for correct drawing
Note: if image strucutre is correct but: i) arrow is missing or ii) "+" symbol is missing or not drawn correctly 0.5p
1p for highway of gradients
Note: if only forward pass is mentioned then no points

0 1

g) For your above drawing, given the partial derivative of the residual block $R(x) = l_2(l_1(x))$ as $\frac{\partial R(x)}{\partial x} = r$, calculate $\frac{\partial H(x)}{\partial x}$.

$\frac{\partial H(x)}{\partial x} = \frac{\partial x + R(x)}{\partial x} = \frac{\partial x}{\partial x} + \frac{\partial R(x)}{\partial x} = 1 + r$
1p for $\frac{\partial H(x)}{\partial x} = 1 + r$

Problem 5 Training a Neural Network (15 credits)

A team of architects approaches you for your deep learning expertise. They have collected nearly 5,000 hand-labeled RGB images and want to build a model to classify the buildings into their different architectural styles. Now they want to classify images of architectures into 3 classes depending on their style:

Islamic

Baroque

Soochow

a) How and in what parts would you split your dataset to build your deep learning classifier? Formulate your answer in percentages.

0.5 pt: Train + Validation + Test (must mention all 3)
0.5 pt: meaningful percentages (training split must be $>$ 50%)

b) After visually inspecting the different splits in the dataset, you realize that the training set only contains pictures taken during the day, whereas the validation set only has pictures taken at night. Explain what is the issue and how you would correct it.

Issue (1p):
0.5 pt: data from different distributions / mismatch and therefore
0.5 pt: bad generalization / high val error
How to correct (1p): mix training + validation data, shuffle, split again => data in all sets from same distribution
Note:
- zero points for “cross validation” (without mentioning re-shuffling first)
- zero points for data augmentation (simply changing brightness will not lead to realistic night images!)

c) As you train your model, you realize that you do not have enough data. Unfortunately, the architects are unable to collect more data so you have to temper the data. Provide 4 data augmentation techniques that can be used to overcome the shortage of data.

Rotation, cropping, flipping, adding noise, mirroring, ... (0.5p each, max 2p)
Note: if they named more than 4 actions: only grade first 4 ones

d) You now start training your network using Gradient Descent (GD) on the entire training data. What might be a problem of Gradient Descent concerning saddle points or local minima of the cost function?

Problem: Gradient is zero at saddle point/ local minimum (0.5p) $\rightarrow$ **Optimization** can get stuck (0.5)

e) While training your classifier you experience that loss only slowly converges and always plateaus independent of the used learning rate. Now you want to use Stochastic Grading Descent (SGD) instead of Gradient Descent (GD). What is an advantage of SGD compared to GD in dealing with saddle points?

SGD has noisier updates (0.5p)$\rightarrow$ can help escape from a saddle point or local minima (0.5p)

f) Explain the concept behind momentum in SGD

Avoid getting stuck in local minima or accelerate optimization (1p) or exponentially weighted average of past gradients.
Note: informal explanations: 0.5p if somewhat correct, 0p if too imprecise

g) Why would one want to use larger mini-batches in SGD?

1p: make the gradients less noisy
only “less noisy” (not mentioning gradients) $->$ half a point

h) Why do we usually use small mini-batches in practice?

Limited GPU memory (1p)
“faster computation” -> 0 points

i) There exists a whole zoo of different optimizers. Name an optimizer that uses both first and second order momentum

Adam (1p)

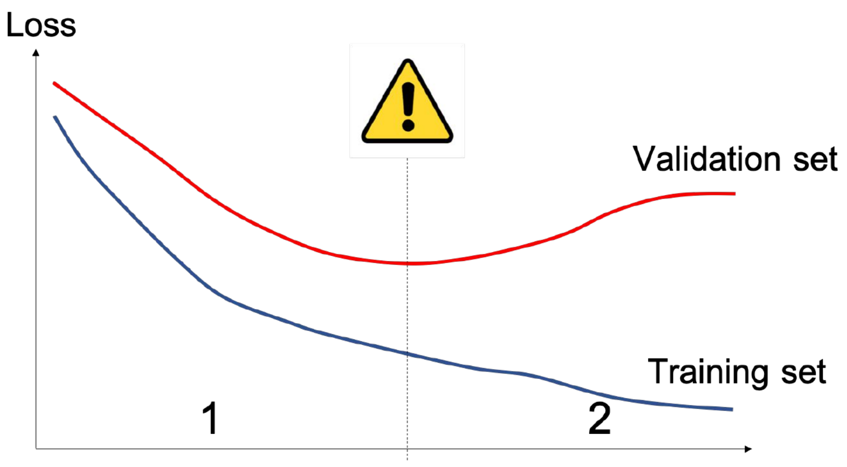
j) Choosing a reasonable learning rate is not easy.

1. Name a problem that will result from using a learning rate that is too high (1p).
2. Name a problem that will arise from using a learning rate that is too low (1p)?

Too high: cost function does not converge to an optimal solution and can even diverge. (1p)
Note: if divergence (or no convergence) not mentioned: -0.5p
Too low: cost function may not converge to an optimal solution, or will converge after a very long time. (1p)

k) Finally you plot the loss curves with a suitable learning rate for both training data and validation data. What's the issue of period 2 called? Name a possible actions that you could do without changing the number of parameters in your network to counteract this problem.

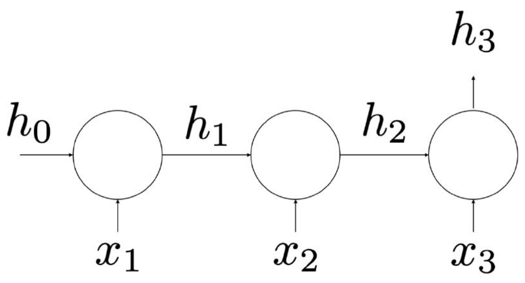
☐ 0 ☐ ☐ 1 ☐ ☐ 2



Issue: Model is overfitting (Note: generalization gap only is not enough) (1p).
Action: weight decay or drop out, data augmentation (1p)
Note: if only regularization 0.5p

Problem 6 Recurrent Neural Networks and Backpropagation (9 credits)

Consider a vanilla RNN cell of the form $h_t = \tanh(V \cdot h_{t-1} + W \cdot x_t + b)$. The figure below shows the input sequence x_1, x_2, and x_3.



0 1

a) Given the dimensions $x_t \in \mathbb{R}^3$ and $h_t \in \mathbb{R}^5$, what is the number of parameters in the RNN cell? (Calculate final number)

$3 \times 5 + 5 \times 5 + 5(bias) = 15 + 25 + 5 = 45$ (1p for 45, else 0)

0 1

b) If x_t and b are the 0 vector, then $h_t = h_{t-1}$ for any value of h_t. Discuss whether this statement is correct.

False: (0.5p)
After transformation with V and non-linearity $x_t = 0$ does not lead to $h_t = h_{t-1}$ (0.5p) , i.e. h_t can be changed.
Note: simply repeating the formula $h_t = \tanh(V \cdot h_{t-1}))$ does not give any points. If you only mention V or *tanh* then this is also correct, though giving an incorrect formula invalidates that half point.

Now consider the following **one-dimensional** ReLU-RNN cell without bias b.

$$h_t = \text{ReLU}(V \cdot h_{t-1} + W \cdot x_t)$$

(Hidden state, input, and weights are scalars)

0 1 2

c) Calculate h_2 and h_3 where

$$V = -3, \quad W = 3, \quad h_0 = 0, \quad x_1 = 2, \quad x_2 = 3 \quad \text{and } x_3 = 1.$$

$h_0 = 0$
$h_1 = \text{relu}\,(-3 \cdot 0 + 3 \cdot 2) = 6$
$h_2 = \text{relu}\,(-3 \cdot 6 + 3 \cdot 3) = 0$ (1 p)
$h_3 = \text{relu}\,(-3 \cdot 0 + 3 \cdot 1) = 3$ (1 p)

Note: Only points for correct solutions, no points for intermediate steps (even if you have an incorrect h_1)

0 1 2 3

d) Calculate the derivatives $\frac{\partial h_3}{\partial V}$, $\frac{\partial h_3}{\partial W}$, and $\frac{\partial h_3}{\partial x_1}$ for the forward pass of the ReLU-RNN where

$$V = -2, \quad W = 1, \quad h_0 = 2, \quad x_1 = 2, \quad x_2 = \frac{3}{2} \quad \text{and } x_3 = 4.$$

for the forward outputs

$$h_1 = 0, \quad h_2 = \frac{2}{3}, \quad h_3 = 1.$$

Use that $\frac{\partial}{\partial x}\text{ReLU}(x)\Big|_{x=0} = 0$.

Generally:

$$\frac{\partial h_t}{\partial V} = h_{t-1} + V \cdot \frac{\partial h_{t-1}}{\partial V}$$
$$\frac{\partial h_t}{\partial W} = \frac{\partial \text{ReLU}(z_t)}{\partial z_t} \cdot \left(V \cdot \frac{\partial h_{t-1}}{\partial W} + x_t\right)$$
$$\frac{\partial h_t}{\partial x_\tau} = \frac{\partial \text{ReLU}(z_t)}{\partial z_t} \cdot \left(V \cdot \frac{\partial h_t}{\partial x_\tau} + W \cdot \delta_{t\tau}\right)$$

$$\frac{\partial h_3}{\partial V} = h_2 + V \cdot h_1 = \frac{2}{3} + 0 = \frac{2}{3} \qquad (1\text{p})$$
$$\frac{\partial h_3}{\partial W} = V \cdot x_2 + x_3 = -2 \cdot \frac{3}{2} + 4 = 1 \qquad (1\text{p})$$
$$\frac{\partial h_3}{\partial x_1} = 0 \text{ (dead ReLU)} \qquad (1\text{p})$$

Note: alternatively $\frac{\partial h_3}{\partial V} = \frac{3}{2}$ if student correctly identified that h_2 should have been flipped to be a correct forward pass.
For $\frac{\partial h_3}{\partial x_1}$, it's okay even if no formula, but some explanation is given (dead relu after first layer)

0 ☐
☐
1 ☐
☐
2 ☐

e) A Long-Short Term Memory (LSTM) unit is defined as

$$\begin{aligned} g_1 &= \sigma\left(W_1 \cdot x_t + U_1 \cdot h_{t-1}\right), \\ g_2 &= \sigma\left(W_2 \cdot x_t + U_2 \cdot h_{t-1}\right), \\ g_3 &= \sigma\left(W_3 \cdot x_t + U_3 \cdot h_{t-1}\right), \\ \tilde{c}_t &= \tanh\left(W_c \cdot x_t + u_c \cdot h_{t-1}\right), \\ c_t &= g_2 \circ c_{t-1} + g_3 \circ \tilde{c}_t, \\ h_t &= g_1 \circ c_t, \end{aligned}$$

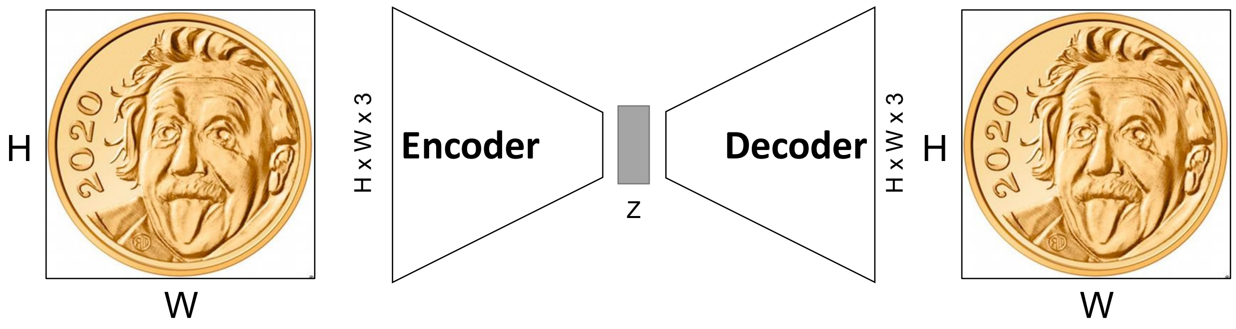
where g_1, g_2, and g_3 are the gates of the LSTM cell.
1) Assign these gates correctly to the **forget** f, **update** u, and **output** o gates. (1p)
2) What does the value c_t represent in a LSTM? (1p)

g_1 = output gate
g_2 = forget gate
g_3 = update gate/input gate
(1p for all three, zero otherwise)
c_t: cell state/memory (1p)
Note: if students interpreted c_t as "what does it do?" half a point was awarded. Possible half point: “Intermediate value, check what to forget and what to add from input”

Problem 7 Autoencoder and Network Transfer (11 credits)

You are given a dataset containing 10,000 RGB images with height H and width W of single coins without any labels or additional information.

To work with the image dataset you build an autoencoder as depicted in the figure below:



The input of the encoder is the images of dimension $(H \times W \times 3)$ which are transformed into a one-dimensional real vector with z entries. The latent code is used to decode the input image with the same dimension $(H \times W \times 3)$. Both encoder and decoder are neural networks and the combined network is trainable and uses the L_2 loss as its optimization function.

a) Is an autoencoder an example of unsupervised learning or supervised learning?

0
1

unsupervised learning (1p) if you give the wrong reasoning after correct choice then only 0.5p

b) As the data gets scaled down from the original dimension to a lower-dimensional bottleneck, an autoencoder can be used for data compression. How does an autoencoder as described above differ from linear methods to reduce the dimensionality of the data such as PCA (principal component analysis)?

0
1

An autoencoder built of neural networks (containing non-linearities) is a non-linear function (1p)
Note: Won't receive point if only mentioned "parameters" ,"conv layers"

c) For an autoencoder we can vary the size of the bottleneck. Discuss briefly what may happen if

i) the latent space is *too small* (1p).

ii) the latent space is *too big* (1pt)

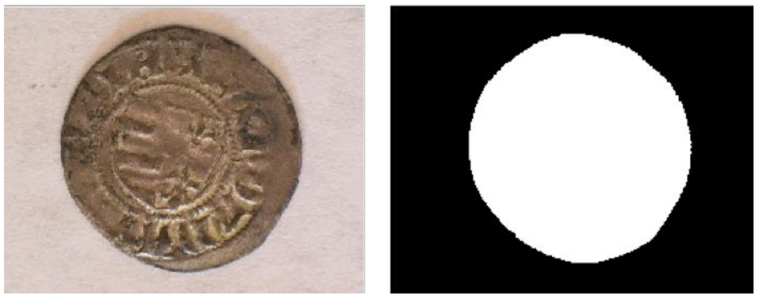
0
1
2

(i) too small: loss of information/bad reconstruction quality/too much compression/extraction/blurry/underfitting... (1p)
(ii) too big: too little compression/extraction/overfitting/poor generalization/learned identity mapping/memorizes the inputs... (1p)

0 ☐ ☐ 1 ☐

d) Now, you want to generate a random image of a coin. To do so, can you just randomly sample a vector from the latent space to generate a new coin image?

No (0.5p), network does not project input distribution surjectively into latent space $\rightarrow$ decoder will not project all latent space vectors to coin images (0.5)
Note: mentioning variational autoencoder does not give points unless it is specified what is different there -> “yes, variational autoencoder can do it” is 0 p, “no, you would need a variational autoencoder” is 0.5p, “no, you would need a variational autoencoder where you can sample from the gaussian latent space” is 1p



0 ☐ ☐ 1 ☐

e) Now, someone gives you 1,000 images that are annotated for semantic segmentation of coin and background as shown in the image above. How would you change the architecture of the discussed autoencoder network to perform semantic segmentation?

Replace the last layer of the of the autoencoder/ decoder (0.5p) to output 1 or 2 channels (0.5p)
Note: “change output layer” is 1p with dimension otherwise 0.5p
“add conv layer” is 1p with dimensions, 0.5p for “add 1x1 conv” else 0p
“add/change/... FC layers” is 0p
mentioning optional improvements like “change to U-Net” does not lead to deduction

0 ☐ ☐ 1 ☐ ☐ 2 ☐

f) If you wanted to train the new semantic segmentation network what loss function would you use and how?

(Binary) Cross Entropy loss/hinge loss (1p) on pixel level/ over channels/ depth-wise (1p) for the two classes (coin and background)
Notes: “dice loss” 1p without explanation, 2p with explanation
“softmax loss” 0p
multiple losses listed: wrong loss + correct loss -> 50% deduction

0 ☐ ☐ 1 ☐ ☐ 2 ☐

g) How would you leverage your pretrained autoencoder for training a new segmentation network efficiently?

- use pretrained encoder of autoencoder and discard decoder (1p)
- (i) freeze weights of encoder or (ii) use very small learning rate for encoder during training for segmentation (1p)

Note: “use transfer learning” - only 1p, you have to mention “weight transfer” and “freeze” or “different learning rate” for encoder/first layers
“freeze all weights of autoencoder and train last layer only” - would not be efficient for segmentation -0.5p

0 ☐ ☐ 1 ☐

h) Why do you expect the pretrained autoencoder variant to generalize more than a randomly initialized network?

access to much more data 1p
Note: can already detect features / features translate to other tasks 0.5p -> 1p for which features (encoder / low-level / shapes &edges / ...)
has prior knowledge of coins 0.5p

Problem 8 Unsorted Short Questions (11 credits)

a) Why do we need activation functions in our neural networks?

0 1

Introduce non-linearity (1p), other wise the network can be reduced to a single linear layer. (alternatively: to model non-linear functions).

b) You are solving the binary classification task of classifying images as cars vs. persons. You design a CNN with a single output neuron. Let the output of this neuron be z. The final output of your network, $\hat{y}$ is given by:

0 1

$$\hat{y} = \sigma\left(ReLU(z)\right),$$

where σ denotes the sigmoid function. You classify all inputs with a final value $\hat{y} \geq 0.5$ as car images. What problem are you going to encounter?

As $ReLU(x) \geq 0$ we get $\sigma(ReLU(z)) \geq 0.5 \quad \forall z$ (0.5p)
$\rightarrow$ classifier only predicts one class (1p)

c) Suggest a method to solve exploding gradients when training fully-connected neural networks.

0 1

Gradient clipping/ better weight initialization (e.g. Xavier initialization)/ Batch normalization

d) Why is it a problem to use the classification accuracy as a loss to train a neural network?

0 1

Accuracy is a discrete function (0.5p) $\rightarrow$ not differentiable (1p)

e) Why do we often refer to $L2$-regularization as “weight decay”? Derive a the mathematical expression that includes the weights W, the learning rate η, and the $L2$-regularization hyperparameter λ to explain your point.

0 1 2 3

Weight update with objective function J incl. weight decay:
$W = W - \eta \nabla_W \left(J + \frac{1}{2}\lambda \sum_i W_i^2\right)$
$W = W(1 - \eta\lambda) - \eta \nabla_W J,$
where η = learning rate and λ = regularization parameter (1p - an error in the equation or not taking the derivate of the L2 term subtracts 0.5 points.) With $\eta \cdot \lambda << 1$. (1p)
Value of W is pushed towards zero in each iteration since a constant amount is subtracted at each iteration. (1p - explanation gives one point if the derivation is correct or partially correct, but without derivation this does not give a point.)

0 ☐ ☐ 1 ☐ ☐ 2 ☐ ☐ 3 ☐ ☐ 4 ☐

f) You are given input samples $\mathbf{x} = (x_1, \dots, x_n)$ for which each component x_j is drawn from a distribution with zero mean. For an input vector $\mathbf{x}$ the output $\mathbf{s} = (s_1, \dots, s_n)$ is given by

$$s_i = \sum_{j=1}^{n} w_{ij} \cdot x_j,$$

where your weights $\mathbf{w}$ are inititalized by a uniform random distribution $U(-\alpha, \alpha)$.

How do you have to choose α such that the variance of the input data and the output is identical, hence $\text{Var}(s) = \text{Var}(x)$?

Hints: For two statistically independent variables X and Y holds:

$$\text{Var}(X \cdot Y) = \left[\text{E}(X)\right]^2 \text{Var}(Y) + \left[\text{E}(Y)\right]^2 \text{Var}(X) + \text{Var}(X)\text{Var}(Y)$$

Furthermore the PDF of an uniform distribution $U(a, b)$ is

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{for } x \in [a, b] \\ 0 & \text{otherwise.} \end{cases}$$

The variance of a continuous distribution is calculated as

$$\text{Var}(X) = \int_R x^2 f(x)\, dx - \mu^2,$$

where μ is the expected value of X.

$\text{Var}(s) = \text{Var}\left(\sum_{j=0}^{n} w_{ij} \cdot x_j,\right) = \sum_{j=0}^{n} \text{Var}\left(w_{ij} \cdot x_j\right)$ (1p)

$\sum_{j=0}^{n} \text{Var}\left(w_{ij} \cdot x_j\right) = n \cdot \text{Var}(w \cdot x) = n \cdot \text{Var}(w) \cdot \text{Var}(x)$ (independence)(1p)

$\text{Var}(w) = \frac{1}{3}\alpha^2 (1p)$

$n \cdot \frac{1}{3}\alpha^2 = 1 \rightarrow \alpha = \sqrt{\frac{3}{n}}$ (1p)

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.