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 1st August, 2023
Examiner:	Prof. Dr. Matthias Nießner	**Time:**	08:00 – 09:30

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

Working instructions

- This exam consists of **24 pages** with a total of **7 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.
- **Answers are only accepted if the solution approach is documented.** Give a reason for each answer unless explicitly stated otherwise in the respective subproblem.
- Do not write with red or green colors nor use pencils.
- Do not write outside of the answer boxes, because this area might get cut off during scanning.
- In case you are running out of space for a question, use the additional pages at the end of the exam. Make sure to indicate that you used the additional pages and to which question it belongs.
- Physically turn off all electronic devices, put them into your bag and close the bag.

Left room from _____ to _____ / Early submission at _____

Problem 1 Multiple Choice (18 credits)

Mark correct answers with a cross ☒
To undo a cross, completely fill out the answer option ■
To re-mark an option, use a human-readable marking ×■

Please note:

- For all multiple choice questions any number of answers, i.e. either zero (!), one 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.**

1.1 Which of the following statements are true about Autoencoders?

- ☐ Autoencoders are only used for image-based applications and cannot be applied to other data types.
- ☒ Autoencoders can be used for dimensionality reduction.
- ☐ Autoencoders require labeled training data to learn meaningful representations.
- ☐ Autoencoders can be trained without a loss function.

Important!

This is NOT a vector, but the samples in the datasets are scalars!

1.2 In the context of recurrent neural networks (RNNs), which of the following statements are true? Select all that apply.

- ☒ RNNs can process sequential data of variable length.
- ☒ RNNs have a hidden state that allows information to persist across time steps.
- ☒ Long Short-Term Memory (LSTM) is an RNN variant that addresses the vanishing gradient problem.
- ☐ RNNs use different weight matrices for each cell.

1.3 In the context of Transformers, which of the following statements are true? Select all that apply.

- ☒ Transformers utilize self-attention mechanisms to capture relationships between different tokens in the input sequence.
- ☐ Transformers rely on recurrent neural networks (RNNs) to model sequential dependencies.
- ☒ The encoder-decoder architecture in Transformers is commonly used for machine translation tasks.
- ☐ Transformers do not require any positional encoding as they inherently understand the order of tokens in a sequence.

1.4 Which of the following statements are true about Batch Normalization?

- ☐ Batch Normalization speeds up the processing time of a single batch.
- ☒ Batch Normalization accelerates the training process in most cases.
- ☐ Batch Normalization layers are always skipped at test time.
- ☒ Batch Normalization reduces the impact of poor weight initialization.

1.5 Which of the following statements are true about loss function and activation in a neural network?

- ☐ The activation function is responsible for handling class imbalances in the dataset.
- ☒ Activations introduce non-linearity to the neural network, enabling it to learn complex patterns.
- ☐ The primary purpose of a loss function is to calculate the accuracy of the model's predictions.

It's to guide the weights through backprop

- ☐ Removed option

1.6 Which of the following statements are true about Semantic Segmentation?

- ☐ Semantic segmentation is an unsupervised learning technique that does not require labeled training data.
- ☐ Semantic segmentation is the process of classifying images into broad categories, such as "dog," "cat," or "car," without pixel-level precision.
- ☐ A single semantic segmentation architecture can be trained for different multi-class segmentation tasks, each with an arbitrary number of classes, without any change to the architecture.
- ☒ The precision of an image semantic segmentation model can be influenced by variations in lighting conditions and image quality.

1.7 In the context of the input data, check all that is true:

- ☒ Random rotations with an angle in $[-30°, 30°]$ are appropriate data augmentation techniques for the MNIST digit dataset.
- ☐ CNNs are not suitable for inputs with high-dimensional input channels.
- ☐ Training on image inputs with values in the range of $[0, 255]$, compared to $[0, 1]$, enhances classification performance as the input data spans a higher value range.
- ☐ Re-scaling the input images from the range of $[0, 255]$ to $[0, 1]$ could mitigate overfitting issues.

1.8 Which of the following statements are true about data augmentation in a supervised learning setup?

- ☒ Data augmentation is a technique used to generate additional data for training.
- ☒ Data augmentation can help reduce overfitting by introducing variations in the training data.
- ☐ Data augmentation is only applicable to image data and not relevant for other types of data.
- ☐ Data augmentation should only be applied to the testing set to evaluate model generalization.

1.9 Which of the following statements are true about dropout regularization in neural networks?

- ☐ Dropout regularization is a technique used to add noise to the input data during training.
- ☒ Dropout regularization helps prevent overfitting by randomly setting a fraction of the neurons in the output of a layer to zero during training.
- ☐ Dropout regularization is only applicable to convolutional neural networks (CNNs) and not relevant to other types of neural networks.
- ☒ Dropout regularization helps prevent neural networks from memorizing specific sample(s) and promotes the learning of more generic features.

Problem 2 Short Questions (15 credits)

0 1 2 3

2.1 You are training a network to classify between [dog, cat, monkey] with a multiclass cross-entropy loss. Consider the two proposed outputs of the network, after the application of the softmax function:

$$\hat{y}_1 = [0.4, 0.3, 0.3]^\top, \hat{y}_2 = [0.4, 0.5, 0.1]^\top$$

- What is the predicted animal class for $\hat{y}_1$ (0.5p) and for $\hat{y}_2$ (0.5p)?
- Assume both inputs are assigned a ground-truth label 'dog', i.e., $y = [1, 0, 0]$. How would the value of the loss function differ between $CE(\hat{y}_1, y)$ and $CE(\hat{y}_2, y)$? Explain your answer. (2p)

- argmax($\hat{y}_1$) = dog (0.5)
- argmax($\hat{y}_2$) = cat (0.5)
- Mentioning that both losses would be the same (1p)
- CE only depends on probability that model gave to GT class/ correct calculation/ correct loss values (1p)

0 1 2

2.2 In the context of the previous question, you decide to drop the softmax activation before you apply the Cross-Entropy loss, to save some calculations. The resulting logits and ground truth one-hot-encoding vector are then:

$$\hat{y}_1 = [-1.0, -1.3, -1.3] \quad (2.1)$$
$$y = [1, 0, 0]$$

Explain why we must apply a softmax-like activation before the usage of the Cross-Entropy loss function, and what problem will occur in this specific case.

- Mentioning that Soft Max pushes logits into the range of [0, 1] or creates probability like representation (1p)
  - Only 0.5 p for Soft Max makes values add up to one/ makes sure that values are positive
- mentioning problem that log(-1) is not defined/ numerical issue (1p)
  - Only 0.5 p for problems with negative values without mentioning why

0 1 2

2.3 Explain the purpose and benefits of performing the "bias correction" step in the Adam optimizer algorithm. Consider the equations:

$$m_{k+1} = \beta_1 \cdot m_k + (1 - \beta_1) \cdot \nabla \mathrm{L}(x, \theta)$$

$$v_{k+1} = \beta_2 \cdot v_k + (1 - \beta_2) \cdot [\nabla \mathrm{L}(x, \theta) \circ \nabla \mathrm{L}(x, \theta)]$$

- issue of initialization bias in the running averages (0.5p)
- they are initialized to 0 (0.5p)
- bias correction step leads to unbiased estimation (0.5p)
- benefits: More significant updates or accelerating training (0.5p)

2.4 Given the following grayscale image (left), a 3x3 convolutional filter produces the right output image. Write down the 3x3 filter values and precisely name the operation the filter performs.

0
1
2

The proposed kernels were verified with: https://setosa.io/ev/image-kernels/
1) **(1p)**: (0.5p) Horizontal (0.5p) edge detector.
2) **(1p)** Accepted kernels:

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

(1p) For one of the given above, or their vertical flip.

Note:

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

Can also be accepted. If you didn't get the points for that, please write in the review.

(0.5p) If their transposed version was given.
(0.5p) For any other kernel that had some horizontal logic in it.

(0p) For the **Laplacian** kernel, as it also extracts the vertical edges.
Laplacian:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

(0p) For any other matrix.

0 1 2

2.5 Deep Learning has huge potential. However, there are also potential risks. Name and explain 2 risks associated with deep learning specifically **in the context of autonomous driving** and propose solutions to mitigate these risks.

Full points: **Distinct** 2 risks and 2 solutions (0.5p each). Also, the solutions had to be relevant to the given proposed risk. Note: DL is a tool, not an AI agent. It cannot drive or run over grandmas.

- – Risk - "Black box" problem: It is not trivial to determine the NN's prediction making process, making it less reliable.
  – Solutions - Use the NN as a component in a bigger software product. \ Apply hand-crafted tests before relying on the model's output, also during inference.
- – Risk - Ethics: When applying AI models to real-world applications, they oppose risks upon decision-making. Not relevant for DL, but accepted, as it is the most common answer. However, only if properly explained.
  – Solution - Introduce ethical constraints to the bigger software component.
- – Risk - DL models are probabilistic models, and therefore are prone to make errors.
  – Solutions: Use the NN as a component in a bigger software product. \ Introduce more and more diverse training data, for better generalization.
- – Risk - The DL model performs poorly on unseen data / environments / mismatch of distribution: The DL is limited by its finite training data.
  – Solution - Collect data from many more environments / countries / weather or lighting conditions. Also data that contains more objects falls into this category.
- – Risk - Limited by hardware: DL models are expansive, and run relatively slow.
  – Solutions - Optimize the model / Install better hardware on endpoint machine.
- – Adversarial attack: Models are prone to slight input changes, that can harm the performance severely.
  – Solution: Train the model to be robust with adversarial training
- Overfitting / Underfitting: These are very general optimization problems of DL. If it is not clear from the answer that the model "overfits" to the training data in the sense that it knows only it, but lacking many other conditions, then no points are given. Underfitting is a known problem, but deploying an underfitted model to a car is not a DL problem, but an engineering problem. In that sense, also deploying an untrained model will introduce the same risk.

2.6 During training, we usually feed the network with batches of inputs rather than a single input at a time. Give two advantages of using a batch of inputs during training over a single input at a time. Explain your statements.

0 1 2

1. **(0.5p)** Better hardware utilization: By processing multiple inputs in parallel, utilizing the computational capabilities of modern hardware such as GPUs, training can be significantly accelerated. This parallel processing enables faster computations, leading to shorter training times. **(0.5p)**
2. **(0.5p)** Improved optimization: Using a batch of inputs allows the model to estimate the gradient of the loss function more accurately. By considering multiple inputs simultaneously, the model can make more informed updates to its parameters during the optimization process. This leads to more stable and efficient convergence, reducing the risk of getting stuck in suboptimal solutions. **(0.5p)**
3. **(0.5p)** Memory Efficiency: Operations can be batched, reducing overhead associated with frequent data transfer. **(0.5p)**

2.7 Find the derivative of the $tanh(x)$ function and express its derivative with the $tanh(x)$ itself:

0 1 2

$$tanh(x) = 2\sigma(2x) - 1$$

where σ is the sigmoid function:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Answer: **(2p)** $1 - tanh(x)^2$. Only **(1p)** point for: $-4\sigma(2x)^2 + 4\sigma(2x)$

$$\begin{aligned}
\frac{\partial 2\sigma(2x) - 1}{\partial x} &= \frac{\partial(2\sigma(2x) - 1)}{\partial x} \cdot \frac{\partial 2x}{\partial x} \\
&= 2 \cdot \left(\frac{\partial(2x)}{\partial x} \cdot (1 - \sigma(2x)) \right) \cdot \frac{\partial 2x}{\partial x} \\
&= 2 \cdot (\sigma(2x) \cdot (1 - \sigma(2x))) \cdot 2 \qquad \textbf{(0.5p)} \\
&= 4\sigma(2x) \cdot (1 - \sigma(2x)) \\
&= -4\sigma(2x)^2 + 4\sigma(2x) \qquad \textbf{(1p)}
\end{aligned}$$

$$\begin{aligned}
-4\sigma(2x)^2 + 4\sigma(2x) &= -4 \cdot \left(\frac{\tanh(x) + 1}{2} \right)^2 + 4 \cdot \left(\frac{\tanh(x) + 1}{2} \right) \qquad \textbf{(0.5p)} \\
&= -4 \cdot \left(\frac{\tanh(x) + 1}{2} \right)^2 + 2 \cdot (\tanh(x) + 1) \\
&= -4 \cdot \left(\frac{\tanh^2(x) + 2 \cdot \tanh(x) + 1}{4} \right) + 2 \cdot \tanh(x) + 2 \\
&= -\tanh(x)^2 - 2\tanh(x) - 1 + 2 \cdot \tanh(x) + 2 \\
&= 1 - \tanh(x)^2 \qquad \textbf{(1p)}
\end{aligned}$$

Problem 3 Backpropagation (15 credits)

You are training a 1-layer neural network to fit the following dataset:

Input $x \in \mathbb{R}$	2	3	4
GT Value $y \in \mathbb{R}$	0.5	0.75	1.0

Important!

This is NOT a vector, but the samples in the datasets are scalars!

You have chosen the following network architecture:

$$\hat{y} = ReLU(Wx + b)$$

$\hat{y}$ is the output of the network, x is the input, W and b are the trainable network parameters.
In the following, for W and b '=' denotes that all entries of the tensors are assigned the same value.

0 1 2

3.1 What is the dimensionality of the tensor W? How many trainable parameters do we have in total?

- W is a scalar; 1x1; dim = 1 **(1p)**
- 2 trainable parameters (1 weight, 1 bias) **(1p)**

0 1 2

3.2 You plan to minimize the following Loss function:

$$\mathcal{L}(y, \hat{y}) = (y - \hat{y})^2$$

Can a loss value of zero be achieved on your dataset? Explain why not, or provide parameters that would achieve that.

- $W = 0.25$ **(1p)**, $b = 0$ **(1p)**
- Matching W and *bias* to one sample input x: each **(0.5p)**. Max **(1.5p)**
- Referencing $\hat{y}$ needing to be zero, no parameters provided: **(0p)**

0 1

3.3 You go ahead with the above loss function $\mathcal{L}$, and start training using SGD, a mini-batch size of 1. Consider the following initialization:

$$W_0 = 1, b_0 = 1$$

Perform a forward pass for the third datapoint, $x = 4$. What is the model's predicted value?

- $\hat{y} = ReLU(1 \cdot 4 + 1) = 5$
- Correct final result with correct solution approach (1p)
- If the student had correct intermediate result, e.g., Wx+b = 5 or **ReLU is missing** (0.5p)
- For correct solution without any intermediate steps (0.5p)
- If the solutions involved all data points and treated W as a vector, like $\hat{y} = ReLU(1 \cdot 2 + 1 \cdot 3 + 1 \cdot 4 + 1) = 10$ (0p)

0 1

3.4 What is the value of the loss function for this datapoint?

- $Loss = (5 - 1)^2 = 4^2 = 16$ (1p)
- Correct final result **without** appropriate solution approach (0.5p)
- If the student had the correct final result even based on the wrong result from previous task (3.3) and have the calculation provided (1p)
- Points are given appropriately in various cases, e.g.
  - If y=4 is used instead of y=1 (considered as misread) and the calculation is correct. (0.5p)
  - If a placeholder like $\hat{y}$ for prediction is used and the final formular is correct. (0.5p)

3.5 We will now perform the backward step. Use the chain rule and the above calculations to calculate the value of the derivatives $\frac{\partial \mathcal{L}}{\partial W}$ and $\frac{\partial \mathcal{L}}{\partial b}$.
(**NOTE**: If you are unsure of your calculations up to this point, you may use $\hat{y} = 2$ to avoid chain errors.)

0 1 2 3 4

- For each correctly calculated $\frac{\partial \mathcal{L}}{\partial W} = 32$ or 8 for ($\hat{y} = 5$ or $\hat{y} = 2$) using chain rule (2p). Also for any other $\hat{y}$ or y, if calculation is correct.
- Correct chain rule equations (0.5p for each)
- Correct calculation of $\frac{\partial \mathcal{L}}{\partial \hat{y}} = -2(y - \hat{y})$ (0.5p) $= 8$ or 2 for $\hat{y} = 2$ for $\hat{y} = ReLU(Wx + b)$
- Correct calculation of $\frac{\partial \hat{y}}{\partial z} = 1$ for $z = Wx + b$ (i.e. derivative of ReLU) (0.5p)
- Correct calculation of $\frac{\partial z}{\partial W} = x$ (0.5p) $= 4$ and $\frac{\partial z}{\partial b} = 1$ (0.5p)
- Each of the final results of $\frac{\partial \mathcal{L}}{\partial W}$ and $\frac{\partial \mathcal{L}}{\partial b}$ (each 0.5p)
- 0p for the final results if any results from the previous sub-steps are wrong

3.6 We will now perform an SGD update steps to W and b, using a learning rate of $lr = 0.5$. Write the update equation (1p) and calculate the updated values for W_1 and b_1.
(**NOTE**: If you are unsure of your calculations up to this point, you may use $\frac{\partial \mathcal{L}}{\partial W} = 50$ and $\frac{\partial \mathcal{L}}{\partial b} = -10$ to avoid chain errors.)

0 1 2

$$W_1 = W_0 - lr \cdot \frac{\partial \mathcal{L}}{\partial W} = 1 - 0.5 \cdot 32 = -15 \quad \text{or} \quad 1 - 0.5 \cdot 50 = -24$$

(1p)

$$b_1 = b_0 - lr \cdot \frac{\partial \mathcal{L}}{\partial b} = 1 - 0.5 \cdot 8 = -3 \quad \text{or} \quad 1 - 0.5 \cdot -10 = 6$$

(1P)
Note: 0.5p for each general equation and 0.5p for each correct result. Points are also given if used $\frac{\partial \mathcal{L}}{\partial W}$ or $\frac{\partial \mathcal{L}}{\partial W}$ is wrong.

3.7 In the next iteration of the training loop, the first datapoint $x = 2$ is chosen. What is the value of the gradient? Using the above values for W_1, b_1, perform the forward pass and calculate the model's **prediction** and the value of the **loss** function. (**NOTE**: If you are unsure of your calculations up to this point, you may use $W_1 = -5, b = 3$ to avoid chain errors.)

0 1

$$pred = ReLu(-15 * 2 - 3) = 0 \quad \text{or} \quad ReLu(-24 * 2 + 6) = 0 \quad \text{or} \quad ReLu(-5 * 2 + 3) = 0$$

(0.5p)

$$loss = (0.5 - 0)^2 = 0.25$$

(0.5p)
Note: Full points are given if results are incorrect because of wrong results in previous questions.

0 ☐
1 ☐

3.8 You let the training loop keep running from this point on for 3 more iterations. You can assume training samples will be selected by their order of appearance in the table (i.e, 2,3,4,2,3,4,...). After the last gradient descent step, write down W_4?

(1p) Gradients are dead, weights same as before: $W_1 = W_4 = -15$ or -24 or -5

(0.5p) just mentioning dead gradients, but with no or wrong conclusion

Note: Full points given if result is incorrect because of wrong results in previous questions.

0 ☐
1 ☐

3.9 What minimal modification could you make to your **network** to improve the training, and what issue would this change address and how? (Assume the same learning rate, initialization and training scheme as previously).

(0.5p) changing to another activation function, that doesn't kill gradients (e.g. Leaky ReLu)
(0.5p) correct explanation (dead gradients, dead ReLu problem etc.)

(0p) for changes regarding learning rate, initialization or training scheme (see task description)

Note: Full points given for changes to the network that aren't related to dying gradients, but would help training in general.

Problem 4 Optimization (14 credits)

You work for a gambling company, and with the Boxing World Cup coming up, your job is to create a deep learning model that predicts the outcome of a match between two boxers. You have access to a lot of data collected over the past 10 years from different boxing matches around the world. The model needs to take two vectors as input, one for each boxer, containing information like their age, number of matches, number of wins, weight, height, and more. The goal is to predict which boxer is going to win. You have decided to use the Kaiming initialization for the weights, the ReLU activation, and the SGD optimizer with a learning rate of $1e^{-4}$. Additionally, you use L_2 regularization with $\lambda = 1$.

4.1 How would you construct the input to the model (1p)? What type of architecture fits the task more appropriately - An Autoencoder, a Fully Connected network, or the VGG16 architecture (0.5p)? Explain (1p).

0 1 2

- non symmetric function ($f(x_1, x_2) \neq f(x_2, x_1)$) of both vectors (e.g. concatenate both vectors, difference between vectors etc.) (1p) (Addition is symmetric, since $x_1 + x_2 = x_2 + x_1$, so the network can not possibly say which input is the winner!)
- fully connected (0.5) + points only if correct network was chosen
  - reason against VGG16 (+0.5p) (e.g. need global context)
  - reason against Autoencoder (+0.5p) (e.g. classification task)

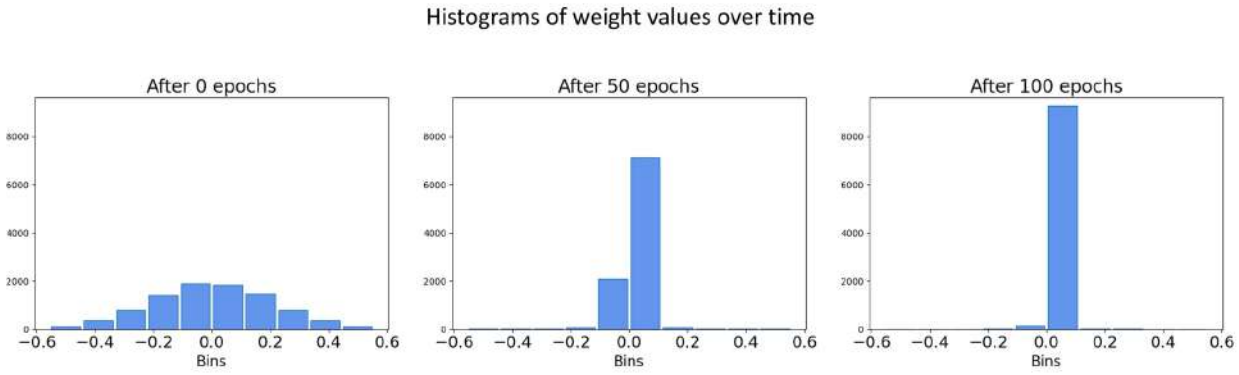
4.2 What is the definition of L_2 regularization and why is it used in general?

0 1 2

- correct formula $reg = \lambda \sum_{i=1}^{n} \theta_i^2$ (1p)
- reason why it is used (1p) e.g.
  - L2 regularization enforces that the weights have similar, small values -> all features get taken into account similarly
  - prevent overfitting / improve generalization by reducing the model's reliance on individual training examples

4.3 After a few epochs of training, you decide to visualize the values of the weight matrices as a histogram. It seems the values of your weights are not very well distributed and all close to 0. Explain why?

0 1



- The L2 regularization coefficient λ is too high. The byproduct of this regularization term is that it makes the weights **very small**, hence its name (1p).
- vanishing gradients/dead Relu would **not** lead to this distribution. Vanishing gradients wouldn't lead to weights to be close to zero, they would prevent the weights from changing at all.
- wrong/bad initialization method is also **wrong**. These are not the outputs from our layers, these are the weight distribution at different training stages!

0 1

4.4 Since it is a binary classification task, you remember that logistic regression utilizes a Sigmoid function as activation. You decide to change all the ReLU activations with Sigmoid activation layers. Which problem does this change to the nextwork introduce, given that nothing else has been changed? How can the problem be mitigated?

The Kamining initialization is specifically designed for layers that utilize the ReLU activation function. However, when applied to layers that use the sigmoid activation function, it violates the principle of maintaining equal variances between the input and output of each layer, as expressed by the equation $\text{Var}(x_{l-1}) = \text{Var}(x_l)$ A solution would be to use the Xavier initialization instead.
(1.5p) if the student explained that kaiming initialization is not suitable for sigmoid activations (1p) and we should use xavier initialization instead (0.5p) also using batch normalization and skip connections are valid answers (0.5p). (1.5p) if the student explained that using sigmoid activation in middle layers will cause vanishing gradients (1p) and using xavier initialization or to replace it back with ReLUs or using batch normalization will solve the problem (0.5p) (1.5p) if the student explained that sigmoid is not zero-centered and we should use zero centered activation function instead

Common mistakes: Centering the data at the beginning
Exploding gradients

0 1 2 3

4.5 Explain the vanishing gradient problem (2p). Name 2 activation functions that are known to suffer from this problem (0.5p each).

The gradients lose their magnitude quite substantially as they flow up the stream, from the loss function to the "shallower" layers. This creates a situation where deeper layers train faster, while shallower layers train much slower, as their gradient magnitude is much smaller, or even not at all, at some exteme cases. (2p)
Activation functions: Sigmoid, tanH (0.5p each).
Saying "kills" the neurons is the wrong answer.

(-0.5p) if the student wrote additional incorrect explanation (-0.5p) if the student didn't express that the vanishing happening due to the multiplication of small gradient values in backpropagation and (-0.5p) for each additional missing part of the definition.

0 1 2

4.6 What is the main difference between the vanishing gradient and the "Dead ReLU"? Explain why the latter happens, and state the core difference between the two.

The "Dead ReLU" is a "neuron-wise" problem of the ReLU activation layer. It occurs when the weights of a ReLU neuron are adjusted such that an intermediate linear layer consistently outputs a value that is smaller than zero, causing the neuron to become permanently inactive due to the saturation. This can happen due to poor weight initialization or high learning rates, preventing the neuron from being activated, contributing to the learning process, and its relevant weights to be modified. This reduces the capacity of the model. If not initialized carefully, this would affect many weights in the network.
The main difference is that the vanishing gradient problem affects the training process by making gradients gradually smaller as they propagate up-the-stream, making an unbalanced training process for shallower and deeper layers of the network, while the "Dead ReLU" problem occurs when ReLU neurons become permanently inactive. **The vanishing gradient problem slows down learning, while the "Dead ReLU" problem limits the network's capacity to learn.**
(1p) if the student explained why dead relu happens and that the gradient will be 0. (1p) for the difference between dead relu and vanishing gradients.
(-0.5p) if the student didn't explain what happens regarding the gradient in dead relu.

4.7 Out of a large dataset containing hundreds of thousands of matches, you have selected the last 50,000 matches as your training set. You realize that the entire training set can be loaded into (V-)RAM and that you can perform mathematical operations with it. Explain how to leverage this fact, to optimize the training process.

0 1 2

- if the student mentions usage of all data at the same time without splitting batches. It is OK not to explicitly point out the name of Gradient Descent. (2p)
- if the student mentions using (Gauss-)Newton method and “to utilize the curvature to converge much faster”. (2p)
- if the student mentions “using a larger batch size”. (1p)

Problem 5 Autoencoder (9.5 credits)

With the knowledge of this course, you are planning to found an AI startup to revolutionize data encryption and compression focusing on company-specific image data. Your idea is to use an Autoencoder-like network architecture, which is trained on the company's data. After the model is converged, only the decoder is distributed once among its clients, and the company can send the encoded, low-dimensional latent vector of the respective data to its clients.

0 1

5.1 Which type of Autoencoder-like architecture is better suited for the task: a vanilla fully-convolutional Autoencoder or a fully-convolutional U-Net architecture? Explain.

0.5p for picking vanilla fully-convolutional Autoencoder.

1p for correct explanation: cannot use U-net since it has skip connections from encoder to decoder, and the user has access only to the decoder.

0 1 2 3

5.2 A company is interested in your technology and wants to give it a try. You train a model on their data, i.e. images of their products. However, they notice that the decoded images are blurry. Name and explain two possible reasons that could cause this problem.

0.5p for stating a valid reason, and 1p for correctly explaining it.

1. Use of MSE/L2 loss: optima/minima is the mean/average of all residuals.
2. Latent space dimension too low: insufficient capacity to represent the data -> loss of information.

0 1 2 3

5.3 After several improvements, the company is convinced of your idea. Also, the company is developing a new product. To make sure employees do not accidentally send encoded images of their new prototype via email to clients, they want to implement an extra feature to detect and prevent these mistakes.
Since there are not yet a lot of images of their prototype to train an image classifier, explain how you might use your previous model to get a usable model given the huge amount of other product images available. Also, include in your answer the loss function and the number of neurons in the output.

The task is to classify whether a product is a "prototype" or not.

(1p) Use the pretrained encoder part as a feature extractor.
(1p) Replace decoder with a classifier and finetune on the small "prototype" dataset.

Just 1p if the student mentions "Transfer Learning", or any other valid short answer.

0.5p + 0.5p for a correct loss and no. of output neuron pair:

- Binary Cross Entropy, 1
- Cross Entropy/Categorical Cross Entropy, 2

5.4 The Lead Product Designer of the company approaches you and would like to know if the approach can also be used to produce images of new product ideas which are similar to the company's previous products. She heard of an architecture called "Variational Autoencoder". Explain why a VAE is better suited for such task (1p). What changes to the architecture are needed (0.5p), and what is the loss function (0.5p)?

0 1 2

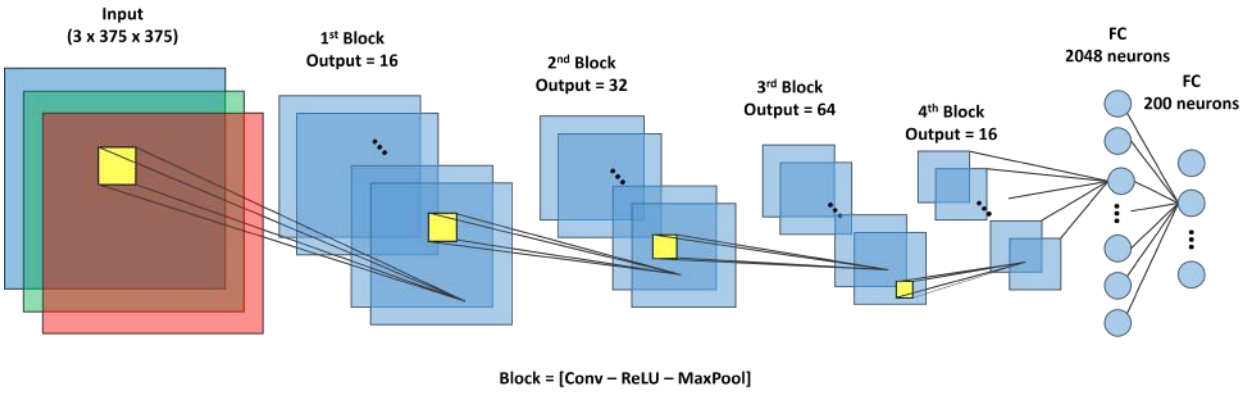
(1p) VAE enforces a probability distribution (gaussian/normal) on the latent space.

(0.5p) Encoder outputs mean and variance of latent space distribution (instead of fixed latent vector).

(0.5p) Reconstruction loss + KL divergence / distribution loss.

Problem 6 CNNs (8.5 credits)

In the following, we assume that the input of our network is a $(3 \times 375 \times 375)$ RGB image. The task is to perform image classification on 200 classes. You design a network with the following structure [CONV - ReLU - MaxPool] x 4 - FC - ReLU - FC. Each convolutional layer has a fixed kernel size 3×3, stride of 1 and no padding. The output channels of these four convolutional layers are 16, 32, 64, and 16. The max-pooling layers have a window size of 2×2 and a stride of 2.



0 1 2

6.1 What is the number of parameters in the **first** Convolutional layer of the network (Consider bias term)?

$(3 \times 3) \times 3 \times 16 + 16 = 448$
1p for the bias term, i.e. 16
1p for the weights, e.g. 432, 3*3*3*16

2p if only the final number given and correct
2p if the complete formula given and correct but without final result, e.g. 28*16, (3*3*3+1)*16
-0.5p if calculation mistakes in the final result, e.g. $16 + 432 \neq 448$, (3*3*3+1)*16 = 160
0p If the final number is incorrect, unless there is an obvious correct subpart for the bias term or the weights.

0 1 2

6.2 Write down the receptive field of a neuron after the **3rd** convolutional layer with respect to the input image. Explain the individual steps graphically or by calculation.

Conv1: 3
Maxpool1: 3 + 1*(2-1) = 4
Conv2: 4 + (1*2)(3-1) = 8
Maxpool2: 8 + (1*2*1)(2-1) = 10
Conv3: 10 + (1*2*1*2)(3-1) = 18

1.5p for correct final receptive field, i.e. 18 or 18*18, without any explanation
0.5p for every correct intermediate result when the final result is incorrect (forwards: 3-4-8-10-18, backwards: 3-6-8-16-18)

6.3 What is the output size of the network after the **4th** Pooling layer? Write down all steps.

0
1
2

Solution: $16 \times 21 \times 21$ Calculation:
General formula for both conv and maxpool: $\lfloor \frac{X_{in}+2p-k}{s} \rfloor + 1$

- $\lfloor \frac{375+2\cdot 0-3}{1} \rfloor + 1 = 373$
  - $\lfloor \frac{373+2\cdot 0-2}{2} \rfloor + 1 = 186$
- $\lfloor \frac{186+2\cdot 0-3}{1} \rfloor + 1 = 184$
  - $\lfloor \frac{184+2\cdot 0-2}{2} \rfloor + 1 = 92$
- $\lfloor \frac{92+2\cdot 0-3}{1} \rfloor + 1 = 90$
  - $\lfloor \frac{90+2\cdot 0-2}{2} \rfloor + 1 = 45$
- $\lfloor \frac{45+2\cdot 0-3}{1} \rfloor + 1 = 43$
  - $\lfloor \frac{43+2\cdot 0-2}{2} \rfloor + 1 = 21$

6.4 We want to modify the network to be fully convolutional. State the shapes of the new weight matrices.

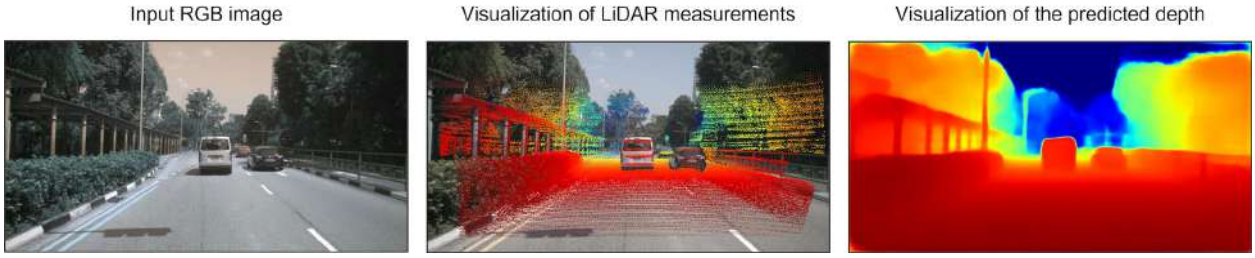
0
1
2

1st FC layer $\rightarrow$ ($2048 \times 16 \times 21 \times 21$).
2nd FC layer $\rightarrow$ ($200 \times 2048 \times 1 \times 1$).

1p for each layer
1p if any reasonable idea stated without giving the concrete solution in the context of this problem, e.g. apply multiple additional conv layers after the 4th layer so that the output size of the final layer is 200*1*1
2p 16*21*21*200

Considering that this sub-problem depends on the last one, the solution will also be accepted in case that the student uses his/her incorrect result from 6.3 but with a correct approach for this sub-problem.

Problem 7 Data Loading (10 credits)



For autonomous driving, cars are often equipped with expensive LiDAR sensors which capture the distance to its surrounding. One essential task in computer vision is depth prediction from a single RGB image. The ground truth for this supervised task can be acquired from an RGB camera and a LiDAR sensor attached to the car, which is driving around streets.
The recorded data consists of scenes captured in both daylight and nighttime. The dataset is sorted in such a way that all the daytime scenes are stored first, and so the nighttime ones are stored last. Consecutive data samples (⟨RGB image, depth map⟩ pair) may come from the same scene, captured with a gap of 20 milliseconds.

0 1 7.1 While loading your training data, you notice the following batch:

$$\begin{bmatrix} \langle \text{sun-0001.jpg, sun-0001-gt.png,} \rangle \\ \langle \text{sun-0002.jpg, sun-0002-gt.png,} \rangle \\ \langle \text{sun-0003.jpg, sun-0003-gt.png,} \rangle \\ \langle \text{sun-0004.jpg, sun-0004-gt.png} \rangle \end{bmatrix}$$

Where "sun" indicates that the images were taken during daytime. State one problem with this batch, given the proposed dataset, and explain it.

(0.5p for naming the problem and 1p for **fully** explaining it):

(0.5p) Homogeneous features/images / all sunny, consecutive, sequential images / not shuffled properly / domain gap / batch doesn't represent the whole dataset / if student wrote "not equally distributed" / no (well) generalization without stating to which kind of data
(1p) Results in biased gradient updates towards daytime images / overfits to the daytime images / doesn't generalize well to the night time images because of noisy updates as all images in batch are sunny

0 1 7.2 Explain why it is important to shuffle the dataset before splitting it into the splits (train / val / test)?

(0.5p) Splits have similar/same distribution / To avoid bias in training/evaluation / to solve the problem in 7.1 / explaining data mismatch / domain gap

(1p) Making sure the splits represent as much as possible the general distribution of the dataset / To have our distributions in splits being representative of the overall/general distribution / to represent all possible variance/cases in the overall dataset.

No points for: Data has to be **from** the same distribution in all datasets

Note that before shuffling, everything that is in your dataset - that is the distribution. When splitting it - there might pop up problems of mistmach distributions like day and night time images.

7.3 Let's assume the dataset consists of very high-resolution images. You set the batch size to 16, but after starting your training script, you get the following error message "RuntimeError: CUDA Out of memory". Name two possible solutions to mitigate memory usage.

0 1 2

1. Downsample Images: Reduces memory requirements and allows for more images to fit into each batch.
2. Use Image Crops: use crops of the raw image.
3. Reduce Batch Size: Decrease the number of images loaded into memory at once. Reduces the memory needed to process each batch.
4. Upgrade Hardware: Upgrade the hardware, such as using a GPU with a larger VRAM capacitys. This allows for larger batch sizes and improves training performance.

1P if students mention downsample, rescale, resize, compression or similar expression, **0.5P** for mentioning conv layer or pooling (won't help here, still need to load the high-resolution images first before feeding them into model), **0P** for encode.

1P if students mention reduce batch size or similar expression.

1P if students mention use image crops or similar expression, **0.5P** for only mentioning transformation or data augmentation.

1P if students mention upgrade hardware or change hardware with more memory or similar expression.

1P if students mention smaller model, model with fewer parameters, or similar expression, **0P** for dropout.

Maximum **2P**

7.4 While loading the data, you desire to apply various transformations: data augmentation and input normalization operation. To which of the splits (train, val, test) do we apply the mentioned transformations? Explain the difference.

0 1 2

- Data augmentation is applied **only** to the training split. The purpose of data augmentation is to increase the diversity of the training set, which helps the model generalize better and reduces overfitting. Augmentations are not applied to the validation and test splits to ensure that the evaluation is performed on the original, unmodified data.
- Normalization operation is applied to all sets. Normalization ensures that the input data has a specific range and distribution that the network expects. This preprocessing step helps in improving the convergence and performance of the model during training and evaluation.

0.5P if students only mention: Data augmentation is applied only to the training split.

0.5P if students add explanation, like the goal of data augmentation or similar expression, **0P** if the former conclusion is wrong.

0.5P if students only mention: Normalization operation is applied to all sets.

0.5P if students add explanation, like the goal of normalization or similar expression, **0P** if the former conclusion is wrong.

0 1 2

7.5 In common deep learning libraries (e.g. pytorch), the Dataset and Dataloader classes are commonly used for efficient data handling. One of the essential methods of the Dataset class is getitem(). State the input and output of the getitem() method for a supervised learning scenario. Explain the role of the Dataloader class.

1. **(+0.5p)** Input: The input to the getitem() method is an index/key representing a specific example or instance from the dataset.
2. **(+0.5p)** Output: The output of the getitem() method is a tuple or a dictionary containing the input instance and its corresponding label or ground-truth information.
3. Dataloader: Batching

   0.5P for mentioning index/key

   0.5P for mentioning both input and label (or ground-truth information)

   1P for mentioning batching, **0.5P** for only mentioning shuffle, data augmentation or load data into memory

0 1

7.6 You are adding random horizontal flip as data augmentation to your input images. You observe that your model trains much worse than before, i.e. both the training and validation loss remain high. What could be the reason for this behavior?

- Forget to add flip to ground truth as well.
- Other options

  1P forget to adjust (add flip to) ground truth accordingly

  1P flipping will affect the depth/target value/label

  0.5P if students mention position/depth invariant stuff

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