Exercises

1	2	3	4	5	6	7

Surname, First name

1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9
0	0	0	0	0	0	0	0

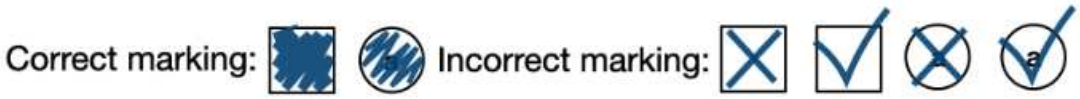
Deep Learning 1 (52041DEL6Y)
Final Exam

_____Final exam for Deep Learning 1 - 23 December 2021 _____

Pay attention to the following:
• Write your name and student number on the front page (don't forget to mark the digits)
• Write all your answers on this exam booklet; it will be scanned and digitally graded.
• Write your answers inside the boxes (it is ok, to slightly go over the margins though).
• The answer boxes should be large enough for your answer
• If you need to empty the answer box in order to start over, ask the invigilator for a blank sticker

During the exam you are allowed to use:
• The provided printed cheat sheet from us.
• A calculator (though not strictly necessary)

About the multiple-choice questions:
• You should fill the boxes and not just check them. E.g.



• In case you want to correct your answer, clearly indicate this(e.g. by filling all boxes and using another indicator such as an arrow to indicate your choice). We can then resolve this during grading.
• The grading of the multiple-choice questions is based on all or nothing: you get full points for ticking off the correct boxes and leaving the others blank, but get zero points if you didn't check all correct answers, or ticked off an incorrect answer.

About the open questions:
• We work with a partial grading scheme: you can get points even if you don't manage to solve the full question.

Please scan over the questions before you start to get an impression of the content
• In total 100 points can be earned.
• You have 2 hours for the exam (except for those with pre-approved extensions)

Multiple choice

4p **1a** Consider the following statements regarding Optimization and Regularization of neural networks. Mark each of the statements which are true.

- [ ] In Layer Normalization, the mean and standard deviation statistics are independent of the batch dimension.
- [ ] During training, Dropout replaces the value of the 'dropped' neurons with new, random samples from a unit Gaussian.
- [ ] The Adam optimizer uses both momentum and an adaptive learning rate.
- [ ] Applying an L1 regularization on the weight parameters is identical to multiplying the parameters by a factor close to 1 at each step (weight decay).

4p **1b** Consider the following statements regarding Convolutional Neural Networks (CNNs). Mark each of the statements which are true.

- [ ] If you apply a convolutional layer with kernel size 3 on an image of size 12x12 (no padding, stride 1), the output of this operation will be the size 9x9.
- [ ] Standard CNNs like the original AlexNet architecture are rotation invariant.
- [ ] Applying 2 3x3 convolutions have fewer parameters than one 5x5 convolution, when the input and output channel size is the same.
- [ ] An inception module applies a 1x1, 3x3, 5x5 and a max pooling operation on the same input feature map.

4p **1c** Consider the following statements regarding Transformers and Multi-Head attention. Mark each of the statements which is true.

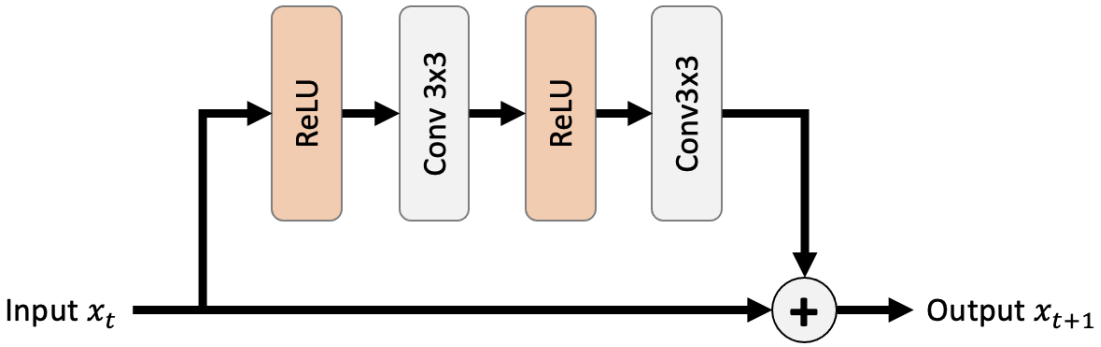
- [ ] Self-attention has a linear complexity over the sequence length.
- [ ] Transformers for language tasks such as machine translation are permutation equivariant.
- [ ] The BERT model is pretrained by autoregressive language modeling.
- [ ] In multi-head attention as applied in Transformers, each head uses a different set of key, query and value vectors.

4p **1d** Consider the following statements regarding Generative Adversarial Networks (GANs). Mark each of the statements which is true.

- ☐ The vanilla Generative Adversarial Network incorporates a reconstruction loss in the generator G.
- ☐ In Virtual Batch Normalization, each data sample is normalized based on a fixed batch of data rather than within its minibatch.
- ☐ The Wasserstein GAN calculates the Wasserstein distance between the output of the generator and a sample from the dataset to reduce mode collapse.
- ☐ Mode collapse is a *notorious* problem in GANs and is related to Discriminator which during training collapses into making random decisions (giving real and fake a probability of 50% for any image).

Initialization in ResNets

Consider a simplified pre-activation ResNet block, similar to the one we have seen in the tutorials:



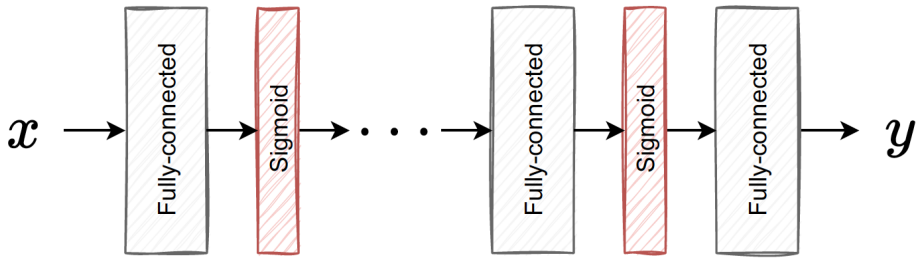
Suppose the convolutional layers have been initialized with the Kaiming initialization for a ReLU activation function.

5p **2a** If the input to this ResNet block (x_t) is distributed according to a Gaussian with zero mean and a variance of σ_t^2, what distribution does x_{t+1} follow?
Hint: you can assume that the input and output variables of a convolution are statistically independent. However, the input and output distributions might have dependencies with their mean and standard deviation.

6p **2b** What problem could this cause in a deep architecture? How does the ResNet architecture overcome it?

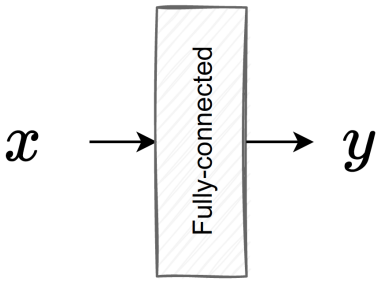
Activation Function and Initialization
Consider two following neural networks for a classification task:

Network 1: a sequence of 5 fully connected layers where except the last layer, each layer has an output dimension of 512, and is followed by a sigmoid activation function. The network's weights are initialized with the Kaiming approach $w_{ij} \sim \mathcal{N}(0, \frac{2}{d_x})$.



Network 1

Network 2: a one-layer neural network.



Network 2

6p **3a** We train both networks on the CIFAR10 dataset using an SGD optimizer. We observe that, despite network 1 having much more parameters, network 2 outperforms network 1. On the test set, network 1 achieves around 10% accuracy, while network 2 achieves 38%. Please elaborate on why network 2 outperforms network 1 in this setup, and what optimization issues network 1 has.

5p **3b** Propose one solution for how you could achieve a higher accuracy with network 1 than network 2, while keeping 5 linear layers and Sigmoid as an activation function. Explain how this method overcomes the previous issues of network 1.

Long Short-Term Memory (LSTM)

Recall the equations that are used to calculate the cell and hidden states of an LSTM cell:

$$\mathbf{c}^{(t)} = \mathbf{g}^{(t)} \circ \mathbf{i}^{(t)} + \mathbf{c}^{(t-1)} \circ \mathbf{f}^{(t)}$$
$$\mathbf{h}^{(t)} = \tanh(\mathbf{c}^{(t)}) \circ \mathbf{o}^{(t)}$$

4p **4a** Give the partial derivative $\frac{\partial \mathbf{c}^{(t)}}{\partial \mathbf{c}^{(t-1)}}$. Describe what elements of the LSTM influence the derivative.
Hint: you can assume that $\frac{\partial \mathbf{h}^{(t-1)}}{\partial \mathbf{c}^{(t-1)}} = \mathbf{0}$*, i.e., the input to the gates does not give any gradients back to* $\mathbf{c}^{(t-1)}$.

Now consider an adapted version of the LSTM-cell, where the cell and hidden states are calculated as follows:

$$\mathbf{c}^{(t)} = \tanh(\mathbf{g}^{(t)} \circ \mathbf{i}^{(t)} + \mathbf{c}^{(t-1)} \circ \mathbf{f}^{(t)})$$
$$\mathbf{h}^{(t)} = \mathbf{c}^{(t)} \circ \mathbf{o}^{(t)}$$

5p **4b** Again, give the partial derivative $\frac{\partial \mathbf{c}^{(t)}}{\partial \mathbf{c}^{(t-1)}}$.
Hint: You can take the same assumption as in part a). Check the equations in the cheat sheet if needed.

6p **4c** Consider now again the two LSTM variants. How does the difference between the position of the non-linearity affect training? For example, would the adapted model in part b) be suitable for learning long-term dependencies in data? Explain your answer.

↳

VAE - ELBO and β-VAE

We have seen that when training a VAE we optimize the ELBO:
$\mathcal{L}_{ELBO} = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})\right] - KL\left(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \| p(\boldsymbol{z})\right)$

3p **5a** As seen above, $\mathcal{L}_{ELBO}$ consists out of two terms. What are these two terms typically called?

4p **5b** Briefly explain why the names of the two terms (max 5 sentences per term) provided in part a are appropriate.

Another variation of VAEs is the β-VAE in which the Kullback-Leibler divergence term is weighted with a parameter β. So that

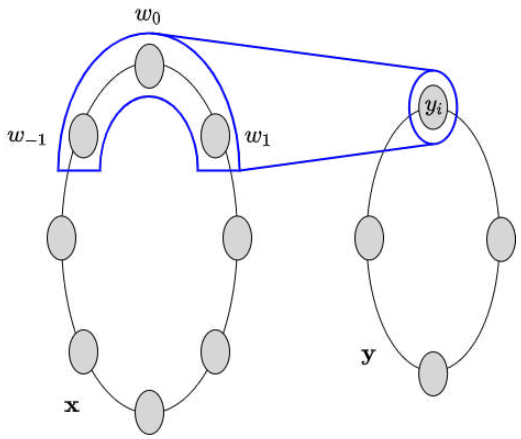
$\mathcal{L}_\beta = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})] - \beta KL\left(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \| p(\boldsymbol{z})\right)$

Note that when $\beta = 1$ is it is identical to the original VAE formulation.

5p **5c** Given that we choose the following prior: $p(\boldsymbol{z}) = \mathcal{N}(0, I_D)$
Briefly explain (max. 10 sentences) what kind of effect a high value of β would have on the trained model?

5p **5d** One way to optimize the ELBO is to find the 'true' posterior and likelihood (or at least a very good approximation of them).
However, there is a common failure mode for this loss in which this does not happen. Starting from the ELBO given above, explain how a model could optimize the ELBO without learning a good encoding model $q_\theta(z|x)$.

Circular Convolutional Layer



You wish to analyze a large amount of traffic data along ring roads encircling various cities, such as the A10 around the center of Amsterdam. This will allow you to investigate pollution, predict traffic jams, or improve travel time estimation in and around urban centers. For this task, you decide to use a convolutional architecture that takes advantage of the circular nature of your dataset. Only odd-sized filters are considered, with components w_k labeled by integer indices and the middle component represented by w_0. The filter size is denoted by F. In the **figure** above, such a circular convolutional layer with $F = 3$ is shown as an example. Biases are omitted for simplicity.

The components of the input feature vector $\mathbf{x} \in \mathbb{R}^M$ are arranged in a circle starting at x_0 and ending at x_{M-1}. For a circular convolution with stride $S = 1$, the components of the output vector $\mathbf{y} \in \mathbb{R}^N$ are then formally calculated as follows:

$$y_i = \sum_{k \in \mathbb{Z}} w_k x_{i+k}$$

Note that for index values beyond the filter size the weights are considered to be set to zero. On the other hand, indices of the feature vectors are defined modulo the size of the feature vector.

4p **6a** How does the above expression for y_i change for a general stride $S \geqslant 1$? (You may assume that the number of features in the input M is divisible by the stride S.)

6p **6b** You find expressions for the gradients required during backpropagation in an old textbook. Unfortunately, some of the indices are illegible (represented below by Greek letters):

$$\frac{\partial L}{\partial x_i} = \sum_m \frac{\partial L}{\partial y_m} w_\alpha$$
$$\frac{\partial L}{\partial w_i} = \sum_m \frac{\partial L}{\partial y_m} x_\beta$$

Find expressions for α and β for general stride S using the chain rule and your solution for (a).

4p **6c** A fellow student in your Deep Learning class makes the following statement: *"During backpropagation, the filter from a convolutional layer is spatially flipped compared to the filter used in the forward pass."* Demonstrate what they mean by this using your results from the previous question, or otherwise.

↳

We wish to extend the above in order to allow for mini-batches of arbitrary size B. Input and output features will therefore be collected in feature matrices $\mathbf{X} \in \mathbb{R}^{B \times M}$ and $\mathbf{Y} \in \mathbb{R}^{B \times N}$, respectively.

4p **6d** Using the chain rule, explain how the gradient formulas above need to change to allow for batch-size 1"> $B > 1$.

4p **6e** Consider a circular convolutional layer with filter size $F = 3$ and stride $S = 2$. The number of input and output features are as shown in the **figure** above. The forward pass can then be written succinctly as $\mathbf{Y} = \mathbf{X}\mathbf{W}^\top$, with a weight matrix $\mathbf{W} \in \mathbb{R}^{N \times M}$. Write down all the elements of the weight matrix explicitly.

Circular Convolutional Layer + GNNs

Consider a function f which is applied on a graph with adjacency matrix A and node features X. Remember that we can write this function also more generally as:

$$f(X, A) = \begin{bmatrix} g(x_1, \chi_1) \\ g(x_2, \chi_2) \\ \vdots \\ g(x_M, \chi_M) \end{bmatrix}$$

where x_i are the rows of X, and χ_i is the set of node features of the nodes in the neighborhood of node i.

4p **7a** Write down what it means for the function f to be permutation equivariant, and what is the condition on the function g that ensures permutation equivariance of f. Write one example of such g that can be used in the equation.

4p **7b** Suppose you are performing classification with a GNN. Write an example of a layer that you can apply at the end of a GNN that reduces the input X to a single feature vector, before eventual linear layers for classification. Describe how this layer ensures the GNN's predictions are permutation invariant.

This page is left blank intentionally