Deep Learning

MSc in Computer Science and Engineering

MSc in Electrical and Computer Engineering

Final exam — February 26, 2022

Version A

Instructions

- You have 120 minutes to complete the exam.
- Make sure that your test has a total of 9 pages and is not missing any sheets, then write your full name and student n. on this page (and your number in all others).
- The test has a total of 19 questions, with a maximum score of 100 points. The questions have different levels of difficulty. The point value of each question is provided next to the question number.
- Please provide your answer in the space below each question. If you make a mess, clearly indicate your answer.
- The exam is open book and open notes. You may use a calculator, but any other type of electronic or communication equipment is not allowed.
- Good luck.

Part 1	**Part 2**	**Part 3, Pr. 1**	**Part 3, Pr. 2**	**Total**
32 points	18 points	25 points	25 points	100 points

Part 1: Multiple Choice Questions (32 points)

In each of the following questions, indicate your answer by *checking a single option.*

1. (4 points) Which of the following is the derivative of the tanh activation function?

   ■ $1 - \tanh(s)^2$

   □ $\tanh(s)(1 - \tanh(s))$

   □ 1 if $s > 0$, 0 otherwise.

   □ None of the above.

   Solution: The correct option is $1 - \tanh(s)^2$.

2. (4 points) Your model is overfitting. Which of these strategies could mitigate the problem?

   ■ **Augment your training set with more labeled data.**

   □ Decrease regularization.

   □ Increase the learning rate.

   □ All of the above.

   Solution: Annotate more data. Regularization should increase, not decrease.

3. (4 points) A **transformer-based** sequence-to-sequence model translates an input sentence of M words into an output sentence with N words. How does the number of computational operations (algorithmic complexity) increase as a function of M and N?

   □ $O(M + N)$

   □ $O(MN)$

   ■ $O(\max(M, N)^2)$

   □ $O(M^N)$

   Solution: The correct option is $O(\max(M, N)^2)$, since self-attention has a quadratic cost.

4. (4 points) Which of the following sentences is **true**?

   □ Neural networks work well in practice because their loss function is convex.

   ■ **Variational auto-encoders are an instance of a latent variable model.**

   □ Generative adversarial networks maximize a lower bound of the data log-likelihood.

   □ Convolutional neural networks are equivariant to rotations and scalings.

   Solution: Variational auto-encoders are an instance of a latent variable model. They maximize a lower bound of the data log-likelihood, GANs do not.

5. (4 points) Consider a linear model used in a binary classification task, where the output corresponds to the probability of $y = +1$ given the input, and trained with a binary cross-entropy loss. Which of these statements is **false**?

   - □ The model corresponds to a logistic regression classifier.
   - ■ **The model is trained with the perceptron algorithm.**
   - □ The decision boundary for the network is a hyperplane in feature space.
   - □ The network can be trained using stochastic gradient descent.

   Solution: The binary cross-entropy loss indicates that the model is trained as a logistic regression classifier, which differs from the perceptron algorithm.

6. (4 points) Which of these statements is **false**?

   - □ A common used objective in GANs is

     $$\min_G \max_D \mathbb{E}_{\mathbb{P}_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{\mathbb{P}_{\boldsymbol{\theta}}(h)}[\log(1 - D(G(h)))],$$

     where G is the generator and D is the discriminator.
   - ■ **GPT-3 is an encoder-only model trained on a masked language modeling objective.**
   - □ VAEs often suffer from posterior collapse.
   - □ A common ethical concern with existing deep learning systems is their bias against minority groups not well represented in their training data.

   Solution: GPT-3 is a decoder-only model. **BERT** is an encoder-only model trained on a masked language modeling objective.

7. (4 points) The first layer in AlexNet can be specified by the following code line in Pytorch:

   ```
   nn.Conv2d(3, 96, kernel_size=11, stride=4)
   ```

   where the first two parameters correspond, respectively, to the number of input and output channels, and no padding is used. Suppose that the input images are 223 × 223 × 3. What is the size of the output after the above convolutional layer?

   - □ 213 × 213 × 96
   - □ 54 × 54 × 11
   - ■ 54 × 54 × 96
   - □ None of the above.

   Solution: The final dimension is $M' \times M' \times F$, where $M' = \lfloor (M - K)/S \rfloor + 1$ and F is the number of filters. In our case, $M = 223$, $K = 11$, $S = 4$ and $F = 96$, yielding $M' = (223 - 11)/4 + 1 = 54$.

8. (4 points) Suppose that a **max pooling** layer with a 2 × 2 kernel and stride of 1 receives the following input:

   $$\boldsymbol{x}_{\text{in}} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}.$$

   What is the output of the pooling layer?

□ $\boldsymbol{x}_{\text{out}} = \begin{bmatrix} 3 & 4 \end{bmatrix}$.

□ $\boldsymbol{x}_{\text{out}} = \begin{bmatrix} 1.5 & 2.5 \\ 4.5 & 5.5 \end{bmatrix}$.

■ $\boldsymbol{x}_{\text{out}} = \begin{bmatrix} 5 & 6 \end{bmatrix}$.

□ $\boldsymbol{x}_{\text{out}} = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix}$.

Part 2: Short Answer Questions (18 points)

Please provide **brief** answers (1-2 sentences) to the following questions.

1. (6 points) Explain which problem long-short term memories (LSTMs) try to solve and how they do it.

   Solution: LSTMs solve the vanishing gradient problem of RNNs. They do it by using memory cells (propagated additively) and gating functions that control how much information is propagated from the previous state to the current and how much input influences the curreent state.

2. (6 points) What is an auto-encoder and why it can be useful?

   Solution: Auto-encoders are networks that are trained to learn the identify function, i.e., to reconstruct in the output what they see in the input. This is done by imposing some form of constraint in the hidden representations (e.g. lower dimensional or sparse). They are useful to learn good representations of data in an unsupervised manner, for example to capture a lower-dimensional manifold that approximately contains the data.

3. (6 points) Explain why transformers need causal masking at training time.

   Solution: The self-attention in transformers allows any word to attend to any other word, both in the source and on the target. When the model is generating a sequence left-to-right it cannot attend at future words, which have not been generated yet. At training time, causal masking is needed in the decoder self-attention to mask future words, to reproduce test time conditions.

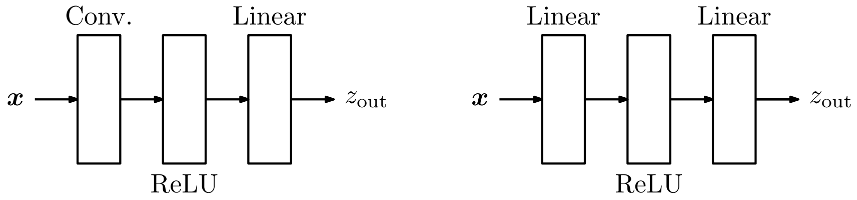
Part 3: Problems (50 points)

Problem 1: Convolutional Neural Networks (25 points)

Consider the two networks depicted in Fig. 1. In the network of Fig. 1a the first block corresponds to a convolutional layer with a single 2×2 filter, no padding and stride $s = 1$. In the network of Fig. 1b the first block corresponds to a hidden layer with 4 units and ReLU activation.

In both networks we denote by $\boldsymbol{z}_1$ the input to the ReLU, by $\boldsymbol{h}_1$ the input to the rightmost linear layer, which in both cases comprises a single unit. We denote by z_{out} the scalar output, such that $\sigma(z_{\text{out}}) = \mathbb{P}[y = +1 \mid \boldsymbol{x}]$, where σ is the sigmoid function.

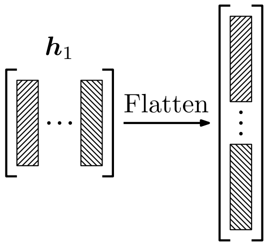
Note: Assume that, before the linear layer, $\boldsymbol{h}_1$ is flattened into a single column vector by stacking all columns of $\boldsymbol{h}_1$ together, as in the following diagram:



(a) Convolutional network.

(b) Fully connected network.

Figure 1: Two network architectures to process the input $\boldsymbol{x}$.



1. (5 points) Suppose that both networks expect as input a 3 × 3 matrix, which in the case of the network in Fig. 1b has been previously flattened. Fill in the table below. When counting the number of parameters in each network, make sure to consider the bias terms.

	Conv.	**Fully conn.**
Dimensions of $\boldsymbol{x}$	3×3	9×1
Dimensions of $\boldsymbol{z}_1$	2×2	4×1
Dimensions of $\boldsymbol{h}_1$	2×2	4×1
Dimensions of z_{out}	1	1
N. param. first layer	5	40
N. param. last layer	5	5

2. (8 points) Suppose that, after training, the parameters of the convolutional network in Fig. 1a are

$$\boldsymbol{K}_1 = \begin{bmatrix} -1 & -2 \\ 1 & 2 \end{bmatrix}, \qquad b_1 = 0 \qquad \boldsymbol{w}_{\text{out}} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}, \qquad b_{\text{out}} = -5,$$

where $\boldsymbol{K}_1$ and b_1 are the parameters of the convolutional layer, and $\boldsymbol{w}_{\text{out}}$ and b_{out} the parameters of the linear layer. Compute the output z_{out} of the network for the input

$$\boldsymbol{x} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Solution: Computing the output of the convolutional layer for the provided input, we get

$$\boldsymbol{z}_1 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + 0 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \qquad \boldsymbol{h}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Then,

$$z_{\text{out}} = \begin{bmatrix}1 & 2 & 3 & 4\end{bmatrix}\begin{bmatrix}1\\0\\0\\1\end{bmatrix} - 5 = 5 - 5 = 0.$$

3. (8 points) Suppose that, after some training iterations, the parameters for the rightmost linear layer of the fully connected network in Fig. 1b are

$$\boldsymbol{w}_{\text{out}} = \begin{bmatrix}1\\2\\3\\4\end{bmatrix}, \qquad\qquad b_{\text{out}} = -5.$$

Let $(\boldsymbol{x}, y)$ be a sample in the dataset for which $y = +1$, and suppose that, when the input is $\boldsymbol{x}$, we have $\boldsymbol{h}_1 = \begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix}^{\top}$. Perform one update of stochastic gradient descent to $\boldsymbol{w}_{\text{out}}$ and b_{out}, using a stepsize $\eta = 1$ and knowing that the loss considered is the negative log likelihood, i.e.,

$$L(z_{\text{out}}; y) = \begin{cases}-\log\sigma(z_{\text{out}}) & \text{if } y = +1;\\ -\log(1 - \sigma(z_{\text{out}})) & \text{if } y = -1.\end{cases}$$

Recall that $\sigma(z) = 1/(1 + \exp(-z))$ and $\sigma'(z) = \sigma(z)(1 - \sigma(z))$.

Solution: We have that

$$z_{\text{out}} = \begin{bmatrix}1 & 2 & 3 & 4\end{bmatrix}\begin{bmatrix}1\\0\\0\\1\end{bmatrix} - 5 = 5 - 5 = 0$$

Also,

$$\begin{aligned}\nabla_{\boldsymbol{w}_{\text{out}}} L(z_{\text{out}}; y = +1) &= (\sigma(z_{\text{out}}) - 1)\nabla_{\boldsymbol{w}_{\text{out}}} z_{\text{out}} = (\sigma(z_{\text{out}}) - 1)\boldsymbol{h}_1,\\ \nabla_{b_{\text{out}}} L(z_{\text{out}}; y = +1) &= (\sigma(z_{\text{out}}) - 1)\nabla_{b_{\text{out}}} z_{\text{out}} = (\sigma(z_{\text{out}}) - 1).\end{aligned}$$

Since $\sigma(z_{\text{out}}) = 0.5$,

$$\begin{aligned}\nabla_{\boldsymbol{w}_{\text{out}}} L(z_{\text{out}}; y = +1) &= \begin{bmatrix}-0.5\\0\\0\\-0.5\end{bmatrix},\\ \nabla_{b_{\text{out}}} L(z_{\text{out}}; y = +1) &= -0.5,\end{aligned}$$

yielding

$$\begin{aligned}\boldsymbol{w}_{\text{out}} &= \begin{bmatrix}1\\2\\3\\4\end{bmatrix} - \begin{bmatrix}-0.5\\0\\0\\-0.5\end{bmatrix} = \begin{bmatrix}1.5\\2\\3\\4.5\end{bmatrix},\\ b_{\text{out}} &= -5 - (-0.5) = 5.5.\end{aligned}$$

4. (4 points) Briefly explain why using a ReLU activation may be preferable over a sigmoid activation in very deep networks.

   Solution: ReLUs are significantly simpler and have a much simpler derivative than the sigmoid, leading to faster computation times. Also, sigmoids are easy to saturate and, when that happens, the corresponding gradients are only residual, making learning slower. ReLUs saturate only for negative inputs, and have constant gradient for positive inputs, often exhibiting faster learning.

Problem 2: Sequence Classification (25 points)

Ada is developing a system for quote attribution, where the input is a sentence (a quote) and the output should be the author of that sentence, among three possibilities: Mark Twain (class 1), Albert Einstein (class 2), and GPT-3 (class 3). Ada trains an RNN-based classifier on a corpus of quotes, obtaining the following model parameters:

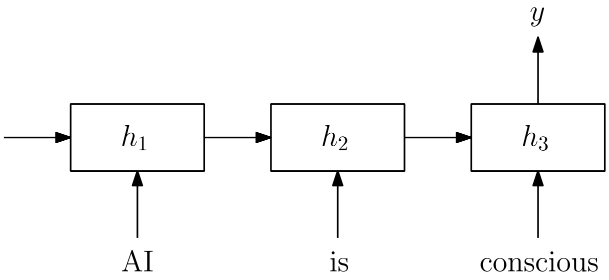
- The initial hidden state of the RNN, $\boldsymbol{h}_0$, is all-zeros.
- The input-to-hidden matrix and the hidden-to-output matrix are respectively:

$$\boldsymbol{W}_{hx} = \begin{bmatrix} 0 & -1 & 2 \\ 1 & -2 & 0 \end{bmatrix}, \quad \boldsymbol{W}_{yh} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 2 & -1 \end{bmatrix}.$$

- The recurrent matrix $\boldsymbol{W}_{hh}$ is the identity matrix.
- All biases are vectors of zeros.
- The RNN uses `relu` activations.

She now wants to use her model to predict the author of the quote "AI is conscious". The relevant word embeddings are:

$$\boldsymbol{x}_{\text{AI}} = [0, 1, 1]^\top, \quad \boldsymbol{x}_{\text{is}} = [1, 0, -1]^\top, \quad \boldsymbol{x}_{\text{conscious}} = [0, -1, 0]^\top.$$



1. (8 points) Assume Ada's model uses the last state of the RNN ($\boldsymbol{h}_3$) to make the prediction. Who is the predicted author of the quote? Show all your calculations.

Solution: We have:

$$
\begin{aligned}
\boldsymbol{h}_1 &= \mathrm{relu}(\boldsymbol{W}_{hx}\boldsymbol{x}_{\mathrm{AI}} + \boldsymbol{W}_{hh}\boldsymbol{h}_0)\\
&= \mathrm{relu}([1,-2]^\top + [0,0]^\top)\\
&= [1,0]^\top.\\
\boldsymbol{h}_2 &= \mathrm{relu}(\boldsymbol{W}_{hx}\boldsymbol{x}_{\mathrm{is}} + \boldsymbol{W}_{hh}\boldsymbol{h}_1)\\
&= \mathrm{relu}([-2,1]^\top + [1,0]^\top)\\
&= [0,1]^\top.\\
\boldsymbol{h}_3 &= \mathrm{relu}(\boldsymbol{W}_{hx}\boldsymbol{x}_{\mathrm{conscious}} + \boldsymbol{W}_{hh}\boldsymbol{h}_2)\\
&= \mathrm{relu}([1,2]^\top + [0,1]^\top)\\
&= [1,3]^\top.\\
\hat{\boldsymbol{y}} &= \mathrm{argmax}(\boldsymbol{W}_{yh}\boldsymbol{h}_3)\\
&= \mathrm{argmax}([-3,1,-1]) = 2 \quad \Rightarrow \quad \text{Albert Einstein.}
\end{aligned}
$$

2. (8 points) Ada realized that her model is much better if she uses a simple attention mechanism (instead of using the last state of the RNN) as the pooling strategy. She adds a query vector $\boldsymbol{q}$ as an extra model parameter, and trains the entire model, obtaining the same parameters as above and $\boldsymbol{q} = [2,-1]^\top$. This model uses as keys and values the RNN states $\boldsymbol{h}_1 = [1,0]^\top$, $\boldsymbol{h}_2 = [0,1]^\top$, $\boldsymbol{h}_3 = [1,3]^\top$. Using **scaled dot product attention**, what is the new prediction for the same quote and which word receives the largest attention probability? Show all your calculations.

Solution: The states $\boldsymbol{h}_1$, $\boldsymbol{h}_2$, $\boldsymbol{h}_3$ are the same as in the previous exercise. The attention scores are:

$$
\begin{aligned}
s_1 &= \frac{1}{\sqrt{2}}\boldsymbol{q}^\top\boldsymbol{h}_1 = \frac{1}{\sqrt{2}}[2,-1]^\top[1,0] = \frac{2}{\sqrt{2}}\\
s_2 &= \frac{1}{\sqrt{2}}\boldsymbol{q}^\top\boldsymbol{h}_2 = \frac{1}{\sqrt{2}}[2,-1]^\top[0,1] = \frac{-1}{\sqrt{2}}\\
s_3 &= \frac{1}{\sqrt{2}}\boldsymbol{q}^\top\boldsymbol{h}_3 = \frac{1}{\sqrt{2}}[2,-1]^\top[1,3] = \frac{-1}{\sqrt{2}}\\
\boldsymbol{p} &= \mathrm{softmax}([s_1,s_2,s_3]) = [0.807, 0.097, 0.097]\\
\boldsymbol{c} &= p_1\boldsymbol{h}_1 + p_2\boldsymbol{h}_2 + p_3\boldsymbol{h}_3 = [0.903, 0.387]\\
\hat{\boldsymbol{y}} &= \mathrm{argmax}(\boldsymbol{W}_{yh}\boldsymbol{c})\\
&= \mathrm{argmax}([-0.387, 0.903, 1.420]) = 3 \quad \Rightarrow \quad \text{GPT-3.}
\end{aligned}
$$

3. (4 points) Is there any possible input for which any of the two models above (with the given parameters) can assign probability higher than $\frac{1}{3}$ to Mark Twain? What if we replace relu by tanh activations? Justify your answer.

Solution: There is no such input. Since the output of relu is non-negative, given any $\boldsymbol{c} = [c_1, c_2]^\top$ with $c_1, c_2 \geq 0$, we have the logits $\boldsymbol{W}_{yh}\boldsymbol{c} = [-c_2, c_1, 2c_1 - c_2]^\top$, therefore the probability of the first class (Mark Twain) is always below or equal to the probability of the first class and below or equal to the probability of the second class, which implies it is less than or equal to $\frac{1}{3}$. This does not happen (necessarily) if we replace relu by tanh, since in that case c_1 and/or c_2 can be negative.

4. (5 points) Give one example of a transformer-based pretrained model that Ada could use for this task and the necessary steps to use it.

   Solution: Ada could use a pretrained encoder-only (e.g., BERT) or encoder-decoder model (e.g., T5, BART) and fine-tune it on the data she has available. Note: a decoder-only model (e.g. GPT-3) would not be the best choice, since this a classification task.